Pre Laboratory 2:

ADDER AND FLIP-FLOP

OBJECTIVES

- The purpose of this lab is to learn how to connect simple input (switches) and output devices (LEDs and 7-segment) to an FPGA chip and implement a circuit that uses these devices.
- Design some combinational circuits that can perform addition and binary-coded-decimal (BCD) addition.
- Besides, investigate latches, flip-flops, and registers.

PREPARATION FOR LAB 2

- Students have to simulate all the exercises in Pre Lab 2 at home. All results (codes, waveform, RTL viewer, …) have to be captured and submitted to instructors prior to the lab session. *If not, students will not participate in the lab and be considered absent this session.*

REFERENCE

1. Intel FPGA training

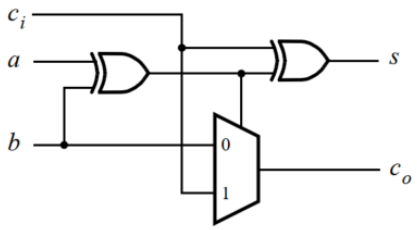
Pre Laboratory 2:

ADDER AND FLIP-FLOP

EXERCISE 1:

Objective: Known how to program full adder.

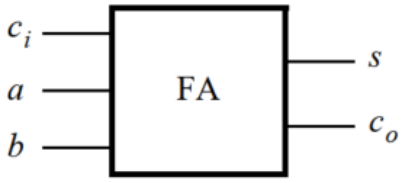
Requirement: Write a VHDL entity to describe the full adder. Figure 1a shows a circuit for a *full adder (FA)*, which has the inputs *a*, *b*, and c_i, and produces the outputs *S* and *co*. Parts *b* and *c* of the figure show truth table and a circuit symbol for the full adder, which produces the two-bit binary sum $c_oS = a + b + c_i$.



(a) Full adder circuit

b	a	c_i	c_o	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

(b) Full adder truth table



(c) Full adder symbol

Figure 2: Instruction to design full adder.

Instruction: Use Karnaugh to minimize the function of s and C_o, then write the VHDL code entity for the full adder.

Minimize function:

$$S = a \oplus b \oplus c_i$$

$$c_o = ab + ac_o + bc_o = ab + c_o(a \oplus b)$$

Pre Laboratory 2:

ADDER AND FLIP-FLOP

Check: Your report has to show two results:

- The waveform to prove the circuit works correctly.
- The result of RTL viewer.

Pre Laboratory 2:

ADDER AND FLIP-FLOP

EXERCISE 2

Objective: Understand RS-latched circuit.

Requirement: Figure 2 depicts a gated RS latch circuit. A style of VHDL code that uses logic expressions to describe this circuit is given below.

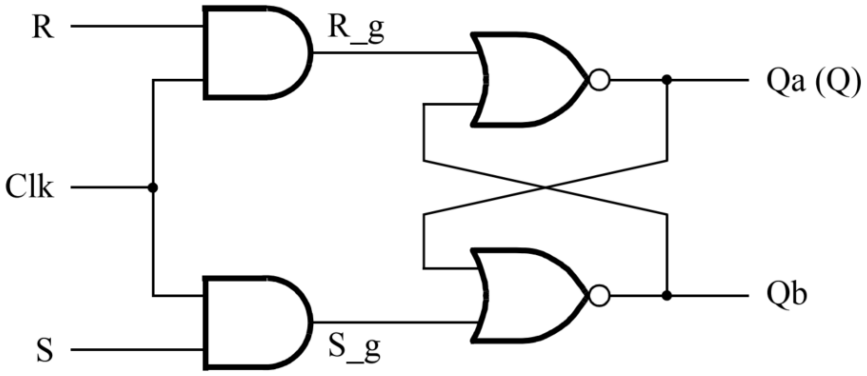


Figure 2: A gated RS latch circuit.

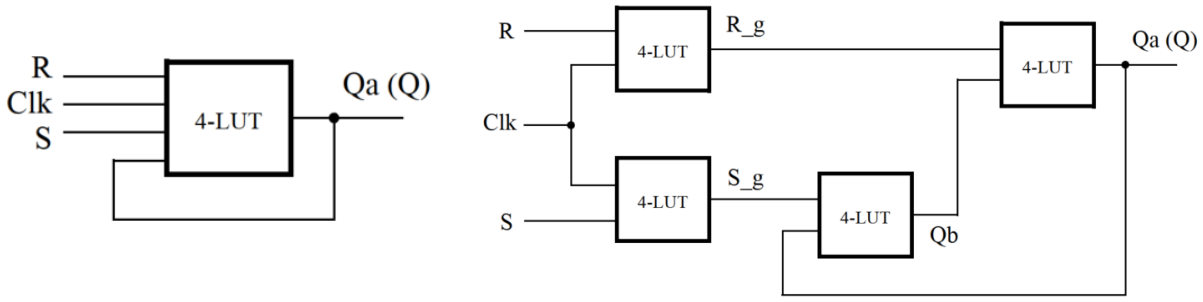
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY part1 IS
PORT ( Clk, R, S : IN STD_LOGIC;
Q : OUT STD_LOGIC);
END part1;

ARCHITECTURE Structural OF part1 IS
SIGNAL R_g, S_g, Qa, Qb : STD_LOGIC ;
ATTRIBUTE KEEP : BOOLEAN;
ATTRIBUTE KEEP OF R_g, S_g, Qa, Qb : SIGNAL
IS TRUE;
BEGIN
R_g <= R AND Clk;
S_g <= S AND Clk;
Qa <= NOT (R_g OR Qb);
Qb <= NOT (S_g OR Qa);
Q <= Qa;
END Structural;
```

Pre Laboratory 2:

ADDER AND FLIP-FLOP



(a) Using one 4-input lookup table (b) Using four 4-input lookup tables

Figure 3: Gated RS latch circuits using lookup tables.

If this latch is implemented in an FPGA that has 4-input lookup tables (LUTs), then only one lookup table is needed, as shown in Figure 3*a*. Although the latch can be correctly realized in one 4-input LUT, this implementation does not allow its internal signals, such as *R_g* and *S_g*, to be observed, because they are not provided as outputs from the LUT. To preserve these internal signals in the implemented circuit, it is necessary to include a *compiler directive* in the code. In the code above, the directive KEEP is included by using a VHDL ATTRIBUTE statement to instruct the Quartus compiler to use separate logic elements for each of the signals *R_g; S_g; Qa;* and *Qb*. Compiling the code produces the circuit with four 4-LUTs depicted in Figure 3*b*.

Instruction:

- Generate a VHDL file with the code above and include it in the project.
- Compile the code. Use the Quartus RTL Viewer tool to examine the gate-level circuit produced from the code, then verify that the latch is implemented as shown in Figure 3b.
- Simulate the behavior of your VHDL code by using the simulation feature provided in the Quartus software. The waveform should be similar to Figure 4. The waveforms in the figure begin by setting *Clk* = 1 and *R* = 1, which allows the simulation tool to initialize all of the signals inside of the latch to known values. If the waveforms provided to the simulation tool do not allow for initialization of all signals in the circuit, then the simulation tool may exit with an error condition.

Pre Laboratory 2:

ADDER AND FLIP-FLOP

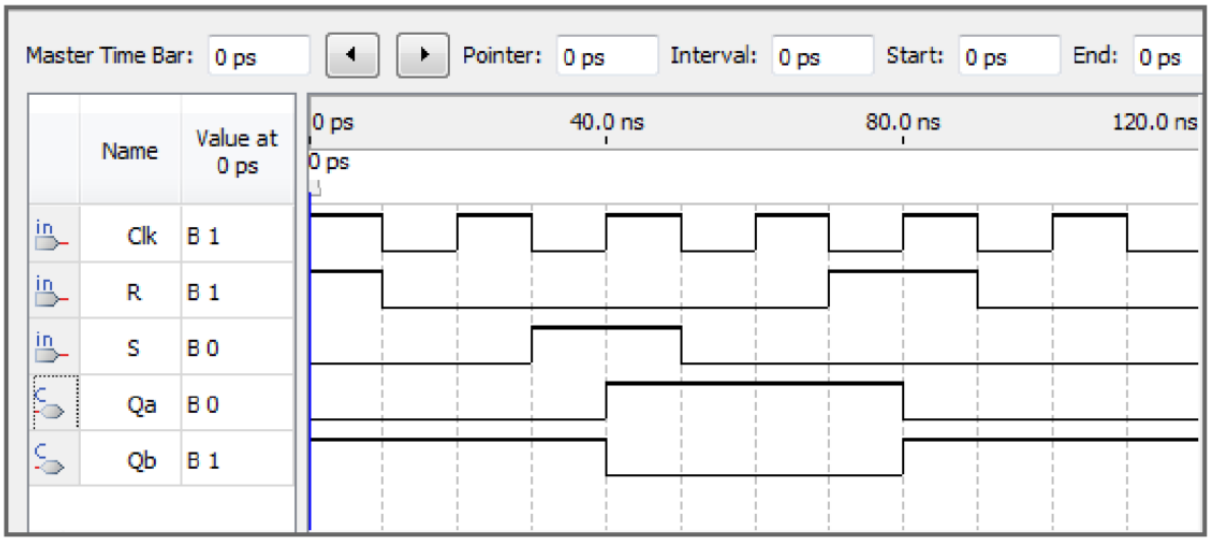


Figure 4: Simulation waveforms for the RS latch.

Check: Your report has to show two results:

- The waveform to prove the circuit works correctly.
- The result of RTL viewer.

Pre Laboratory 2:

ADDER AND FLIP-FLOP

EXERCISE 3

Objective: Understand D-latched circuit.

Requirement: Figure 5 shows a gated D latch circuit. Write VHDL code that uses logic expressions to describe this circuit.

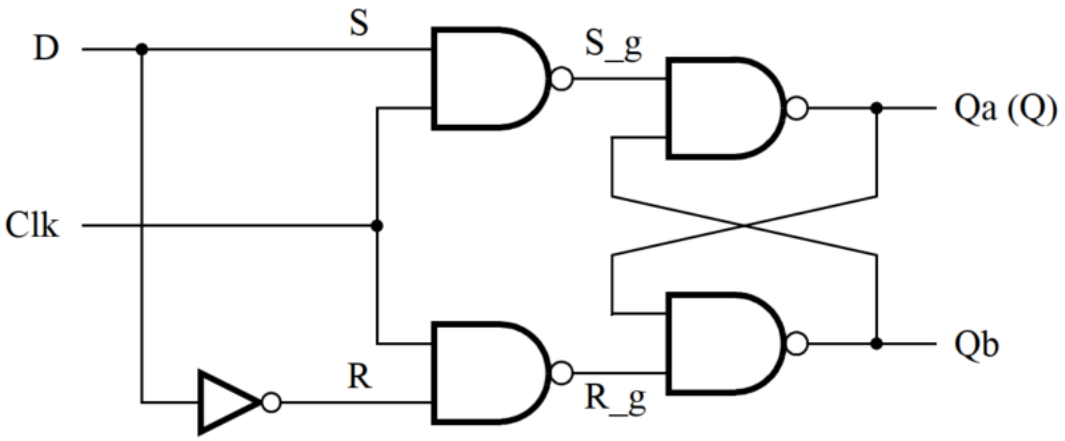


Figure 5: A gated D latch circuit.

Instruction:

- Modify the code in exercise 2 to describe the circuit in Figure 5. Use the KEEP directive to ensure that separate logic elements are used to implement the signals *R*, *S_g*, *R_g*, *Qa*, and *Qb*.
- Generate a VHDL file with the code above and include it in the project.
- Compile the code. Use the Quartus RTL Viewer tool to examine the gate-level circuit produced from the code, then verify that the latch is implemented as shown in Figure 3b.
- Simulate the behavior of your VHDL code by using the simulation feature provided in the Quartus software. Verify that the latch works properly for all input conditions by using functional simulation. Examine the timing characteristics of the circuit by using timing simulation.

Check: Your report has to show two results: The waveform and the RTL result.