Department of Computer Engineering
Tallinn University of Technology

IAS0600

Digital System Design with VHDL
Laboratory Report
Experiment #1: Comparator

Written by
Olatayo Olukotun Joshua
172626IASM

1.0 Introduction

The objective of this Lab work is to describe a digital Comparator (1-bit and 2-bits) circuit using VHDL in Xilinx ISE (Integrated Software Environment). I also examined the two possible approaches of describing 2-bits comparators in VHDL. The functional simulation to verify the correctness of the comparators description was done. Furthermore, the comparator description was also implemented on the Nexys 4 FPGA board for physical testing.

2.0 Background

In digital system, comparison of two numbers is an arithmetic operation that determines if one number is greater than, equal to, or less than the other number [1]. So, comparator is used for this purpose. Magnitude comparator is a combinational circuit that compares two numbers, A and B, and determines their relative magnitudes. The outcome of comparison is specified by three binary variables that indicate whether In1 > In2, In1 = In2, In1 < In2.

In Fig 2.1 the output of a comparator is specified by three possible outcome which are In1 = In2, In1 > In2 and In1<In2

Fig 2.1: Block diagram of a comparator

3.0 Design and implementation of the 2-bit comparator

3.0.1 Writing out Boolean equations for the output of a 2 bit-comparator

A 2-bits comparator compares two number that are two bits (In1 (In1(1), In1(0)) and In2 (In2(1) and In2(0)). The truth table for a two bits comparator with inputs (In1(1), In1(0), In2(1) and In2(0) and output (eq_0, gr_0 and ls_0) is shown Table 3.1 below.

Boolean equation for simple 1-bit comparator can be derived using Karnaugh map.

$$eq\_0 = (In1.In2) + (\overline{In1}.\overline{In2}) = In1 \text{ XNOR } In2 \quad \text{----------3.1}$$

$$gr\_0 = (In1.\overline{In2}) = In1 \text{ AND NOT } In2 \quad \text{----------3.2}$$

$$ls\_0 = (\overline{In1}.In2) = \text{NOT } In1 \text{ AND } In2 \quad \text{----------3.3}$$

Table 3.1: Truth table for a 2-bit comparator

In1(0)	In1(1)	In2(0)	In2(1)	eq_0	gr_0	Ls_0
0	0	0	0	1	0	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	0	1	0
0	1	0	1	1	0	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	0	1	0
1	0	0	1	0	1	0
1	0	1	0	1	0	0
1	0	1	1	0	0	1
1	1	0	0	0	1	0
1	1	0	1	0	1	0
1	1	1	0	0	1	0
1	1	1	1	1	0	0

Boolean equation for a 2-bits comparator can be derived using Karnaugh map

1			
	1		
		1	
			1

(a)

eq_0 = $(In1(1))\bar{}.(In2(1))\bar{}$ $((In1(0))\bar{}(In2(0))\bar{}$ + In1(0). In2(0)) + In1(1). In2(1) $((In1(0))\bar{}.(In2(0))\bar{}$ + In1(0). In2(0) ----------3.4

eq_0 = $((In1(0))\bar{}(In2(0))\bar{}$ + In1(0). In2(0)) $((In1(1))\bar{}(In2(1))\bar{}$ + $\text{In1}(1).\text{In2}(1)$) ----------3.5

eq_0 = (In1(0) XNOR In2(0)) AND (In1(1) XNOR In2(1)) ----------3.6

1			
1	1		1
1	1		

(b)

gr_0 = In1(1)$(In2(1))\bar{}(In2(0)\bar{})$ + In1(0) $(In2(0))\bar{}$+ In1(0)In1(1)$(In2(1))\bar{}$ ----------3.7

gr_0 = In1(0) AND NOT $In2(1)$ AND NOT $In2(0)$ OR In1(1) AND NOT In2(1) OR In1(1) AND In1(0) AND NOT $In2(0)$ ----------3.8

	1	1	1
		1	1
		1	

(c)

ls_0 = $(In1(0))\bar{}$In2(1)In2(0) + $(In1(1))\bar{}(In1(0))\bar{}$In2(0) + $(In1(1))\bar{}$In2(1) ----------3.9

ls_0 = NOT In1(0) AND In2(1) AND In2(0) OR NOT In1(1) AND NOT In1(0) AND In2(0) OR NOT In1(1) AND In2(1) ----------3.10

3.0.2 Use the equation to describe the 2-bit comparator in VHDL

The design of the 2-bit comparator using VHDL was done using the behavioral style description. The code contains the entity declarations and the architecture. The statement part of the architecture includes;

Listing 3.1: The statement part of the architecture for a 2-bit comparator

```
architecture Behavioral of two_bit_comp is

begin

eq_0 <= (In1(0) XNOR In2(0)) AND (In1(1) XNOR In2(1))

gr_0 <= In1(0) AND NOT In2(1) AND NOT In2(0) OR In1(1) AND NOT In2(1) OR In1(1) AND In1(0) AND
NOT In2(0)

ls_0 <= NOT In1(1) AND In2(1) AND In2(0) OR NOT In1(1) AND NOT In1(0) AND In2(0) OR NOT In1(0) AND
In2(1) AND In2(0)

end Behavioral;
```

3.0.3 Performing functional simulation to verify correctness

The simulation view displayed the results as seen in Figure 3.1

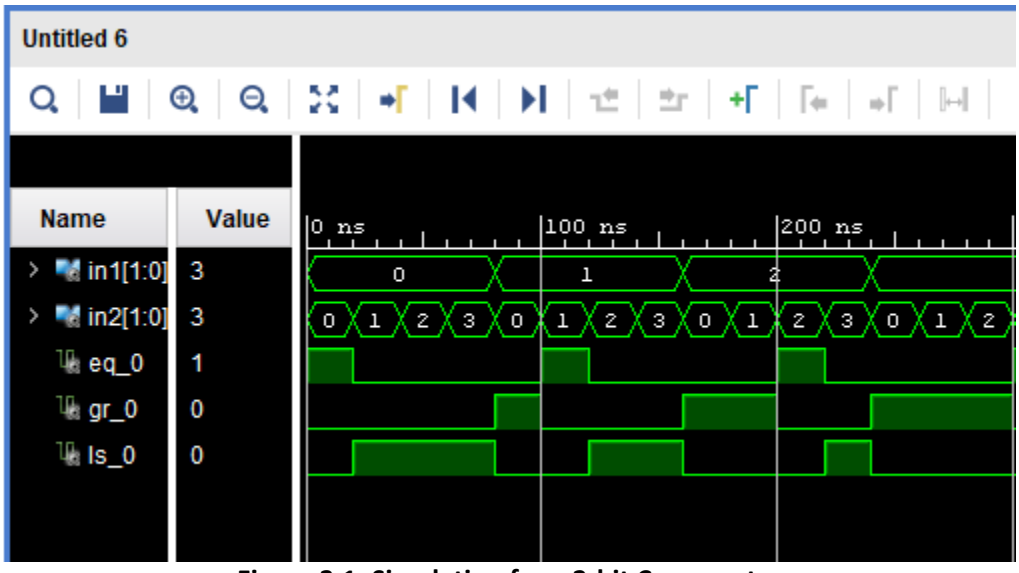

Figure 3.1: Simulation for a 2-bit Comparator

4.0. Design and implementation of the 2-bit comparator using "when...else" statement in VHDL

4.0.1 VHDL Description for 2-Bit Comparator using "when…else" Statement

The description was done in VHDL using "when ... else" statement as seen in the listing 4.1 below.

Listing 4.1: The statement part of the architecture for a 2-bit comparator using when …… else

```
eq_0<= '1' when (in1=in2) else '0';
gr_0<= '1' when (in1<in2) else '0';
ls_0<= '1' when (in1>in2) else '0';
```

4.0.2 Functional simulation to verify correctness of the received VHDL descriptions

Inputs are assigned a test value. After a wait period of 20ns, the outputs are compared to the expected values using "assert" statement (eq_0, gr_0 and ls_0 should be equal to '0'). An error message is reported if they do not match. The statement is reported using a "severity" statement. The severity level is set to "error" and the exact message to be displayed was set using "report".

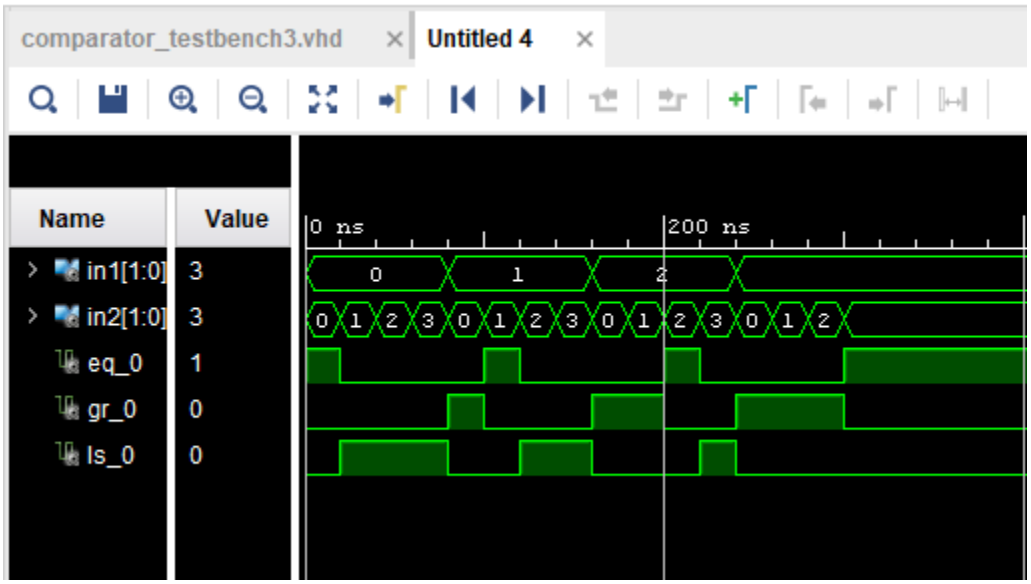

Figure 4.1: Simulation of a 2-bit comparator using "when…..else" statement

5.0. Design and implementation of the 2-bit comparator using "with...select" statement in VHDL

5.0.1 VHDL Description for 2-Bit Comparator using "with…select" Statement

The description was done in VHDL using "when ... else" statement as seen in the listing 5.1 below.

Listing 5.1: The statement part of the architecture for a 2-bit comparator using with …… select

```
with in1=in2 select eq_0<= '1' when true,
        '0' when others;

with in1>in2 select gr_0<='1' when true,
    '0' when others;
```

with in1<in2 select ls_0<='1' when true,
'0' when others;

5.0.2 Functional simulation to verify correctness of the received VHDL descriptions

The simulation process is the same as in section 4.0.2. The result of the simulation is shown in figure 5.1

Figure 5.1: Simulation of a 2-bit comparator using "with…..select" statement

6.0 Result and Discussion

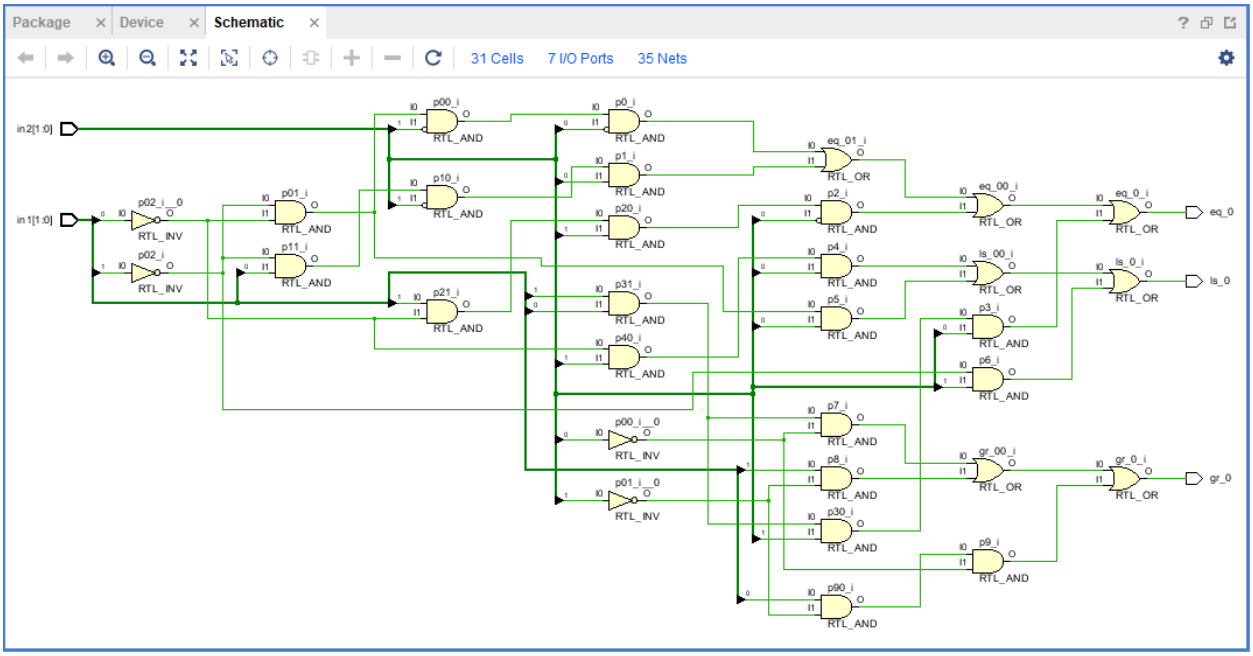

Figure 6.1a: RTL Schematic of a 2-bit Comparator using Boolean

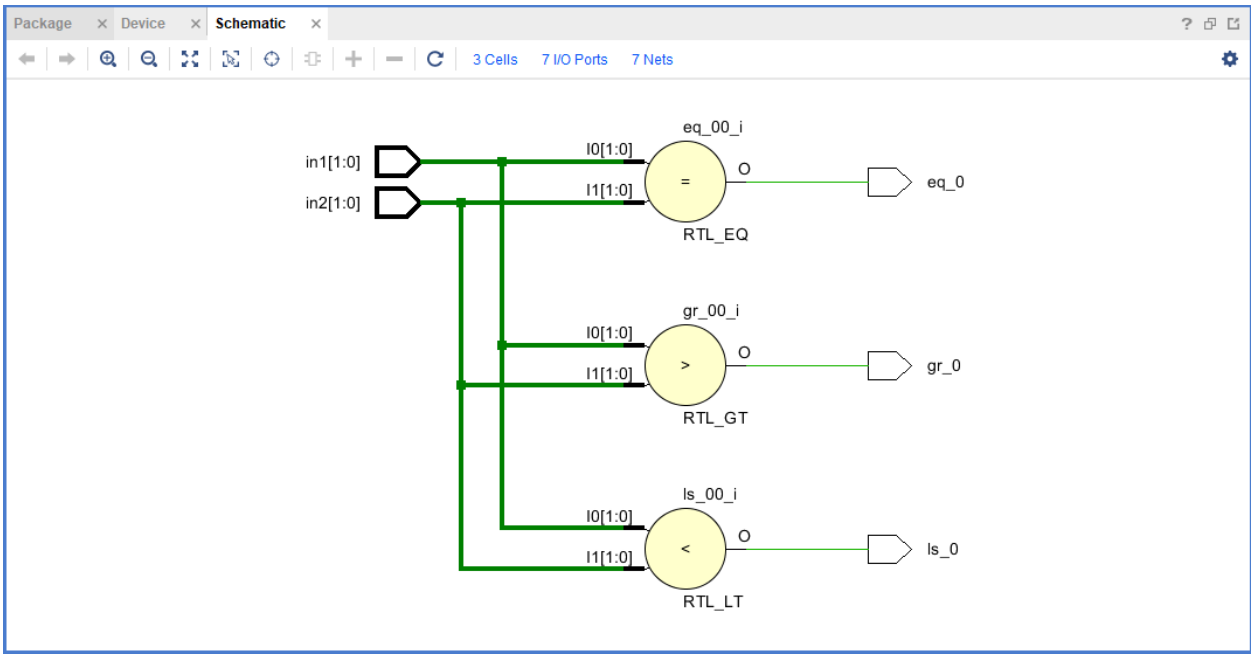

Figure 6.1b: RTL Schematic of a 2-bit Comparator using "when ……else" statement

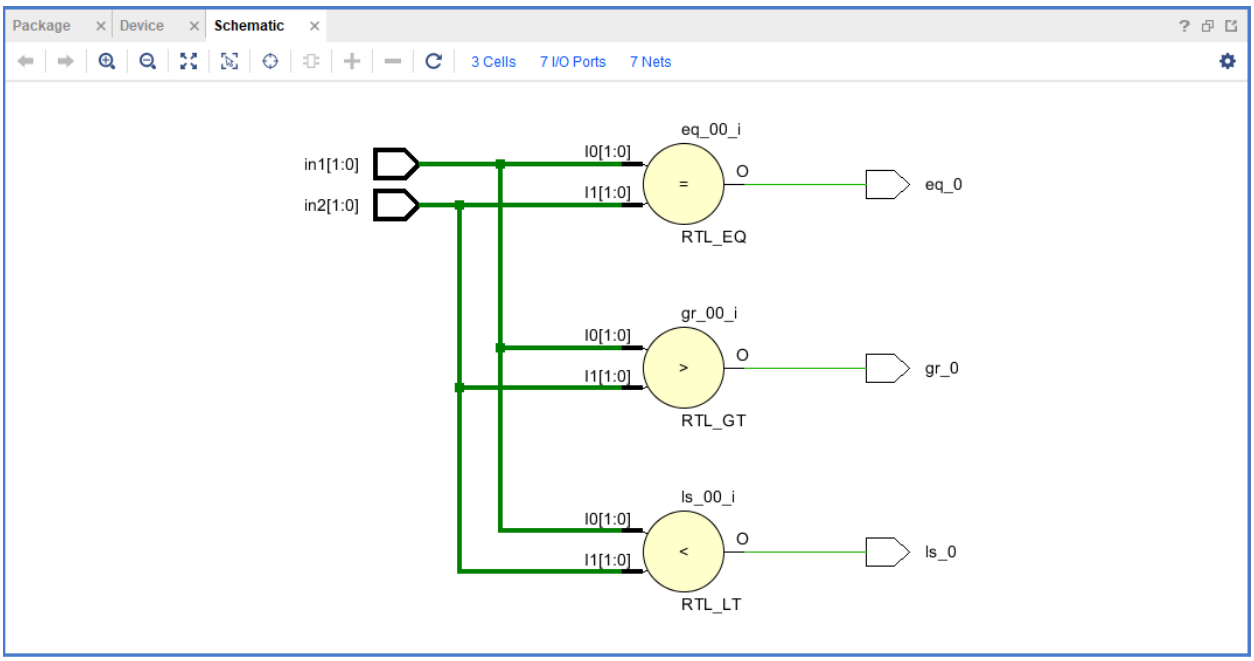

Figure 6.1c: RTL Schematic of a 2-bit Comparator using "with …… select" statement

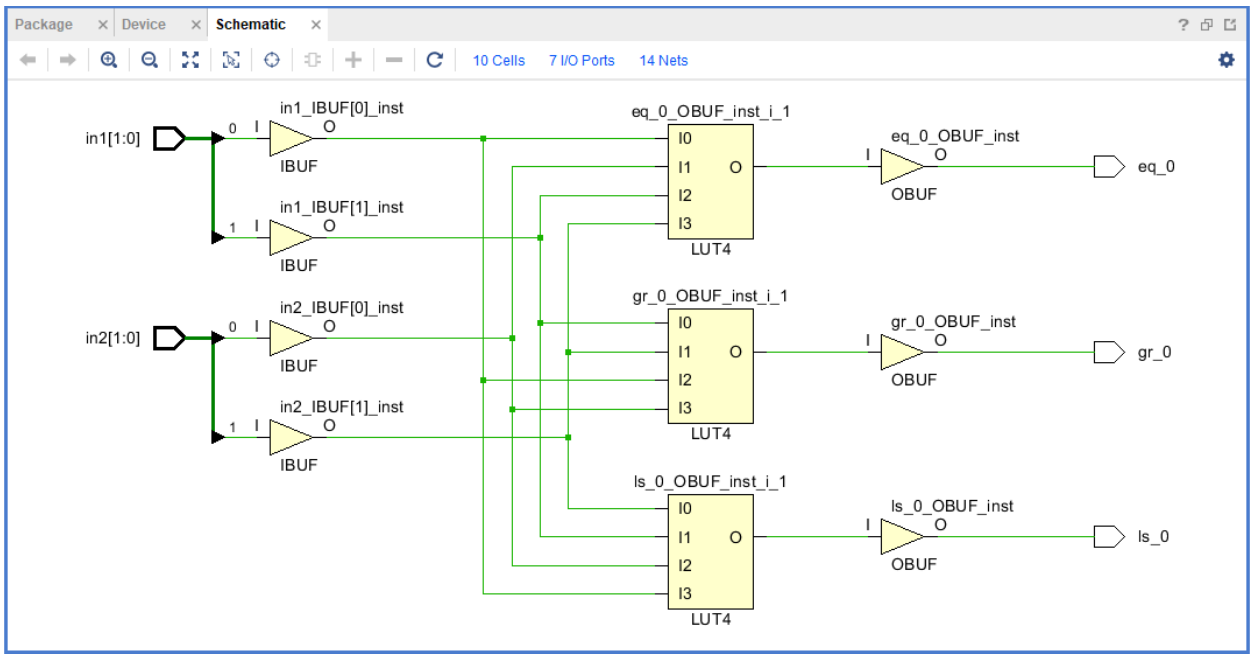

Figure 6.2: implementation Schematic of a 2-bit comparator for Boolean, "when……. else" statement and "with …. select" statement

Comparing the RTL, Synthesis and implementation schematics of all methods used for implementation, some similarities and differences were observed.

The RTL Schematic for Boolean equation is shown in figure 6.1a, it made use of the basic logic circuit building blocks such as the AND, OR, NOT & XOR. Hence, the synthesis was done at the gate level. While the RTL schematic for "when…else" and "with…. select" uses three comparators logic circuits. This is shown in Figure 6.1b and 6.1c respectively. But, the implemented schematics of the three approaches are similar and it is shown in figure 6.2.

The Implementation schematics for the 2-bit comparator using Boolean equation and that of when…else statement and with …. select statement are the same because when the systems execute the implementations, it uses the look up table in the memory, which is the same for all the methods.

7.0 Conclusion

After the implementation of the designs on FPGA development board, I was able to slide the input switches to carry out a visual verification. The input switches were slid based on the values on the truth table and corresponding output were obtained through the LED indicators. The 2-bits comparator was implemented in three different ways (using Boolean equation, "when…else" statement and "with …. select" statement. Corresponding outputs (on both functional simulation and FPGA board) were obtained for the two methods of implementation.

To implement the design on FPGA development board Nexys 4 FPGA Board was used. A XDC (Xilinx Design Constraints) file was created which specifies constraints that are placed on the design. Inputs/outputs ports to device pin mapping were specified in the I/O ports tab of the Result Window area in the RTL Analysis flow section layout. Where inputs In1 and In2 were assigned to M17 and M18 switches of FPGA device package pins respectively. The output eq_0, gr_0 and

ls_0 were assigned H17, K15 and J13 LED of the FPGA board respectively. Then I set I/O Std to "LVCMOS33" in every case. A bit file was created which stores all configuration data that defines the internal logic and interconnections in the target device. The bit file downloaded into the FPGA device's memory cells. The FPGA board was connected to the PC and then powered ON for testing.

From the RLT schematics of the three approaches of describing a 2-bit comparator, one can conclude that the most efficient method using the Boolean equation because it provides a gate level description. Although the Boolean equation approach is quite cumbersome in implementing.