EG3204 Programmable Electronics lab week 4
Designing 1-bit and 2-bit full adders in VHDL

1 Designing 1-bit and 2-bit full adders

Over the term, we will build up a portfolio of working examples of VHDL code on the FPGA board. The purpose of this week's lab is to learn about hierarchical design approaches and re-use of components.

2 About this exercise

In this weeks lab you will design, test, and simulate a 1-bit full adder using the half adder component which you designed last week. You will then design, test and simulate 2-bit full adders using 1-bit full adder components. Make sure your 1-bit and 2-bit full adder designs are in separate Quartus II projects.

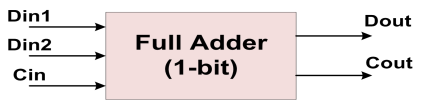
A full adder is a combinatorial circuit which is used to add three 1-bit binary digits including a carry input. It produces two 1-bit binary outputs—sum and carry out. The full adder circuit consists of two half adder circuits and an OR gate. The truth table with the circuit diagram of a full adder is shown in figure 1.

2.1 Design guidance

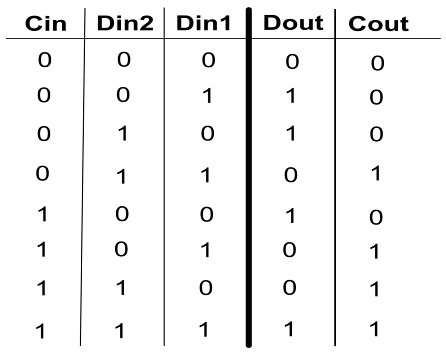
Last week we investigated half adder designs implemented using dataflow modelling. In figure 1(a) the interface of a full adder is described where the input ports are *Din1*, *Din2* and *Cin* and the output ports are *sum* and *cout*.

Architecture declarations describe what the circuit actually does—it implements the associated entity. All the statements in an architecture block execute concurrently and order independently. We were also introduced to the structural level which is used to connect (instantiate) existing components with each other. The main feature of this level is to describe the arrangement and interconnections of components via their ports.

In figure 1(c) we can see the interconnections of two half adders and one OR gate with each others. We can instantiate our half adder component (from last week labs) twice in an architecture and implement the OR gate using concurrent assignment statement. For instance the OR gate and instantiation of a half adder component is described in figure 2(a) and figure 2(b) respectively.

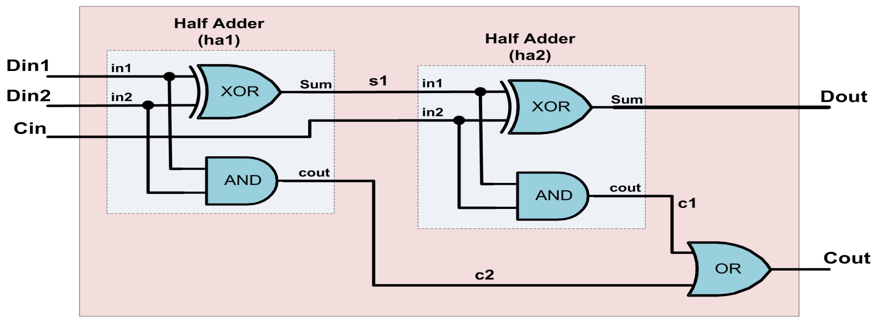


(a)



Cin	Din2	Din1	Dout	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

(b)



(c)

Figure 1: 1-bit Full adder design shows (a) block diagram with Input/Output ports, (b) truth table and (c) the circuit diagram.

3 Test your code in the simulator

Register Transfer Level (RTL) is the name given to the actual format we download onto the FPGA. Once we have completed our RTL design, we need to test its functionality. For this purpose we build a test bench in VHDL where we run the simulation to test the functionality of our RTL design under all the possible test cases. Throughout the term we will use the *Modelsim-Altera tool* for the simulation.

For this lab exercise we provide you two test benches — one is for 1-bit half adder design and other is for 2-bit full adder design. You should use these to fully investigate the functional simulation of your RTL designs.

After testing your 1-bit full adder you need to design a 2-bit full adder. The block diagram for this is described in figures 3(a) and 3(b) respectively.

```
-- For OR gate
Cout <= c1 OR c2;
```

(a)

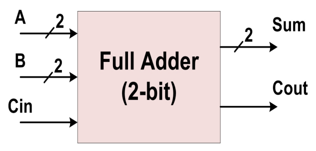
```
-- Instantiation of a half adder
ha1: HalfAdder port map(
                        in1 => Din1,
                        in2 => Din2,
                        Sum => s1,
                        cout => c2
                       );
```

(b)

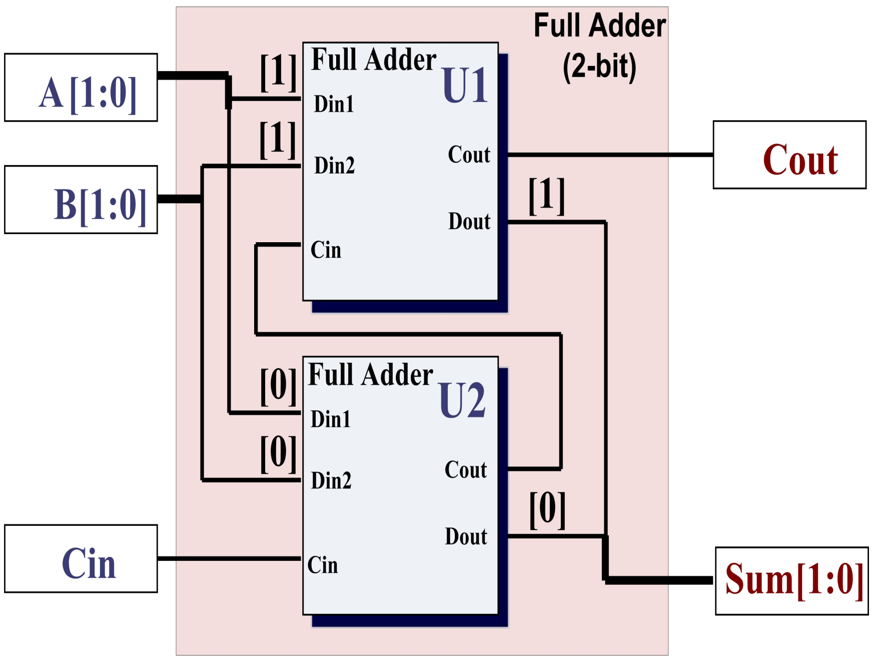
Figure 2: (a) Concurrent assignment statement for OR gate design, (b) Instantiation of a half adder component.

4 Download the code onto the board

Finally, we are in a position to download the compiled bit file (the le that contains the FPGA conguration) onto the board as you have done in previous weeks. If you encounter any difficulties, please ask the demonstrators.



(a)



(b)

Figure 3: 2-bit full adder design shows (a) block diagram with Input/Output ports, (b) Structural diagram.