# Fundamental Combinatorial Blocks

## Topics

- Arithmetic and comparator circuits
- Behavioral simulation

## Problems

*Part I*

In this part, a 4-bit adder/subtractor circuit will be built, from cascaded 1-bit full adders. Assume a two's complement representation.

**1.** *[Paper and pencil]* Write the Boolean equations and draw the logic diagram, based on elementary logic gates, of a 1-bit full adder with the interface shown in Fig.1.

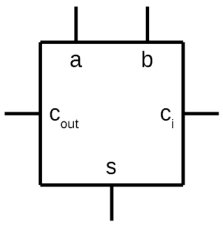


Fig. 1 – Full adder interface.

**2.** *[Paper and pencil]* Create a 4-bit adder/subtractor circuit using the 1-bit full adder as a building block. Fig. 2 shows its interface. Draw the logic diagram of the 4-bit adder/subtractor based on 1-bit full adder modules.

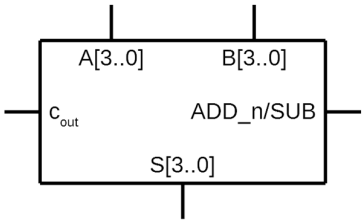


Fig. 2 – 4-bit adder/subtractor interface.

**3.** *[Quartus Prime]* Using the *Quartus Prime* software, create a new project named "AddSubDemo", with a top-level entity with the same name as the project. Create a new file for a schematic diagram called "FullAdder.bdf" to implement the full adder based on logic gates, accordingly to the logic diagram of the point 1. Create a symbol for the "FullAdder" module, so that it can be used in a schematic diagram and save it with the name "FullAdder.bsf".

**4.** *[Quartus Prime]* Create a new file for a schematic diagram called "AddSub4.bdf" to implement the 4-bit adder/subtractor, accordingly to the logic diagram of the point 2. Create a symbol for the "AddSub4" module, so that it can be used in a schematic diagram, and save it with the name "AddSub4.bsf".

**5.** *[Quartus Prime]* Create a new file for a schematic diagram called "AddSubDemo.bdf" that will act as the top-level of the project, instantiate the 4 bit adder/subtractor built in the previous point and connect it to input and output ports.

**6.** *[Quartus Prime]* Perform the behavioural simulation of the adder/subtractor, applying input stimulus to evaluate conveniently its operation.

**7.** *[Paper and pencil]* Include in the diagram of point 2 the required logic to detect overflow conditions for both unsigned and signed operands/results.

**8.** *[Quartus Prime]* Add the overflow detection logic to the diagram drawn in point 4 and validate it adequately through simulation.

## *Part II*

In this part a comparator circuit for two 4-bit unsigned words, A and B, will be designed and validated. The solution should be based on an iterative approach, consisting of a cascade of elementary blocks of comparison (also called 1-bit comparator cells). Each of these cells, in addition to the 1-bit inputs to be compared ($a_i$ and $b_i$), also includes inputs to receive information from upstream comparator cells and, of course, outputs indicating the result of the comparison. Fig. 3 shows the interface of the 1-bit comparator cell. The comparison should start from the least significant bits.

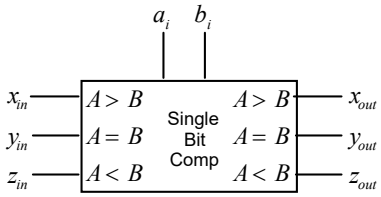


Figure 3 - Elementary comparison block.

**1.** *[Paper and pencil]* Start by manually exercising the comparison process with several comparison cases for 4-bit words and draw the block diagram of the 4-bit comparator based on 1-bit comparison cells. What must be the inputs of the first comparison cell?

**2.** *[Paper and pencil]* To proceed with implementation complete the following logical equations and draw the respective logic diagram.

$$y_{out} = (a_i \oplus b_i)' \cdot y_{in}$$
$$x_{out} = a_i \cdot b_i' + \cdots$$
$$z_{out} = a_i' \cdot b_i + \cdots$$

**3.** *[Quartus Prime]* Using the *Quartus Prime* software, create a new project named "CmpDemo", with a top-level entity with the same name as the project. Create a new file for a schematic diagram called "Cmp1bit.bdf" to implement the 1-bit comparator cell based on logic gates, accordingly to the logic diagram of the previous point. Create a symbol for the "Cmp1bit" module, so that it can be used in a schematic diagram, and save it with the name "Cmp1bit.bsf".

**4.** *[Quartus Prime]* Create a new file for a schematic diagram called "Cmp4bit.bdf" to implement the 4-bit comparator using the 1-bit cell comparator as a building block, accordingly to the block diagram created in point 1. Create a symbol for the "Cmp4bit" module, so that it can be used in a schematic diagram, and save it with the name "Cmp4bit.bsf".

**5.** *[Quartus Prime]* Create a new file for a schematic diagram called "CmpDemo.bdf" that will act as the top-level of the project, instantiate the 4 bit comparator built in the previous point and connect it to input and output ports.

**6.** *[Quartus Prime]* Perform the behavioural simulation of the comparator, applying input stimulus to evaluate conveniently its operation.

**7.** *[Homework after the class]* Repeat this part for signed numbers.

**8.** *[Homework after the class]* How do you modify the equations of the 1-bit comparison cell if the comparison starts from the most significant bits?