Group 07

Paul Doumet
260 226 189

Simon Foucher
260 223 197

Circuit Name:
G07_full_adder

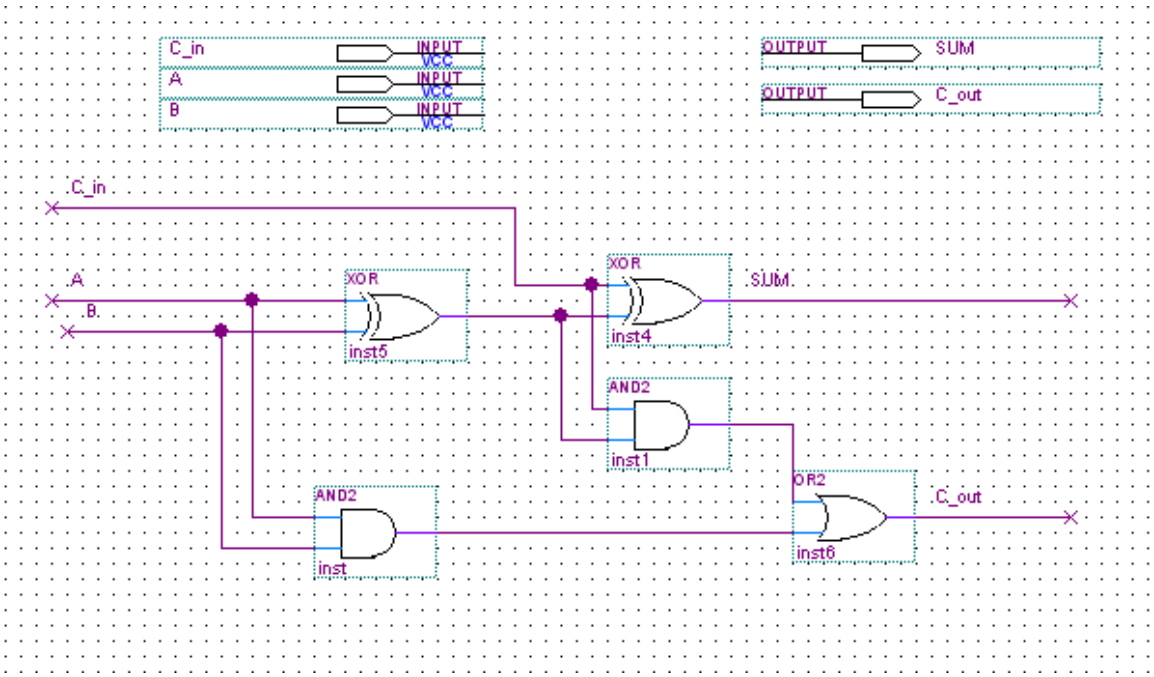
Part 2: The adder.

Circuit Description and schematics

The basic building block used for the 28 bit adder is a full adder. It has 3 inputs: the 2 bits to be compared and a carry in line. The outputs are the sum of both bits, and a carry out. The basic adding operation is mapped out on the following truth table:

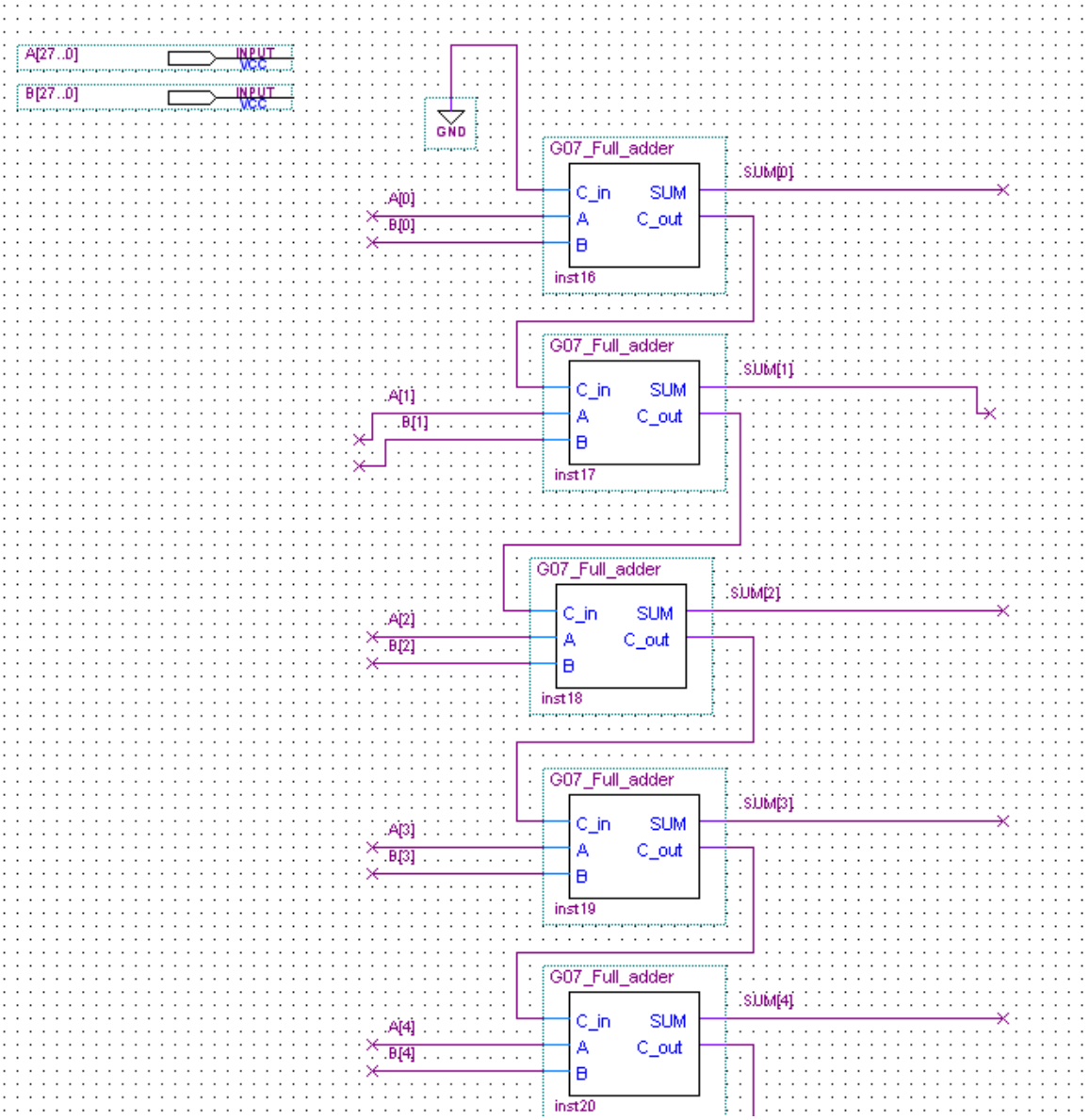
A	B	Carry in	Carry OUT	Sum
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1

Which we implemented using the following circuit:

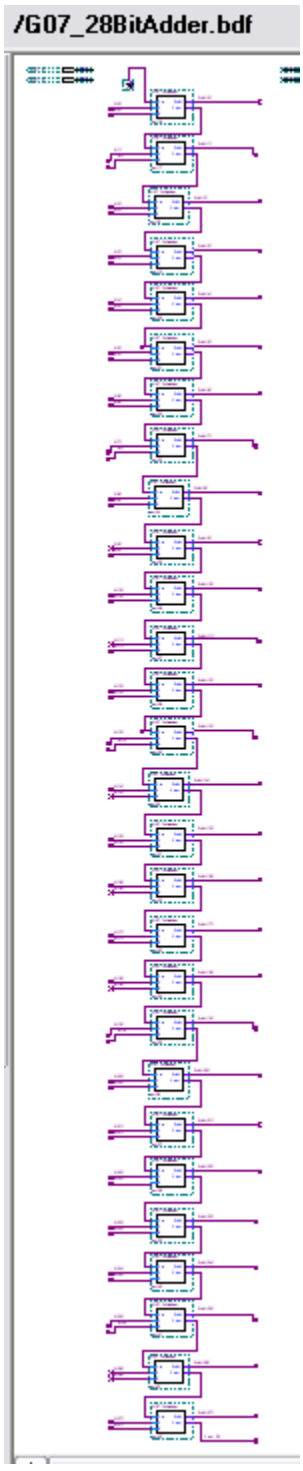


This full adder's main function is to add 3 numbers: A and B (both user specified) and C_in; the carry in from the adjacent less significant bit. The sum is outputted in the SUM line, and the carry out in C_out.

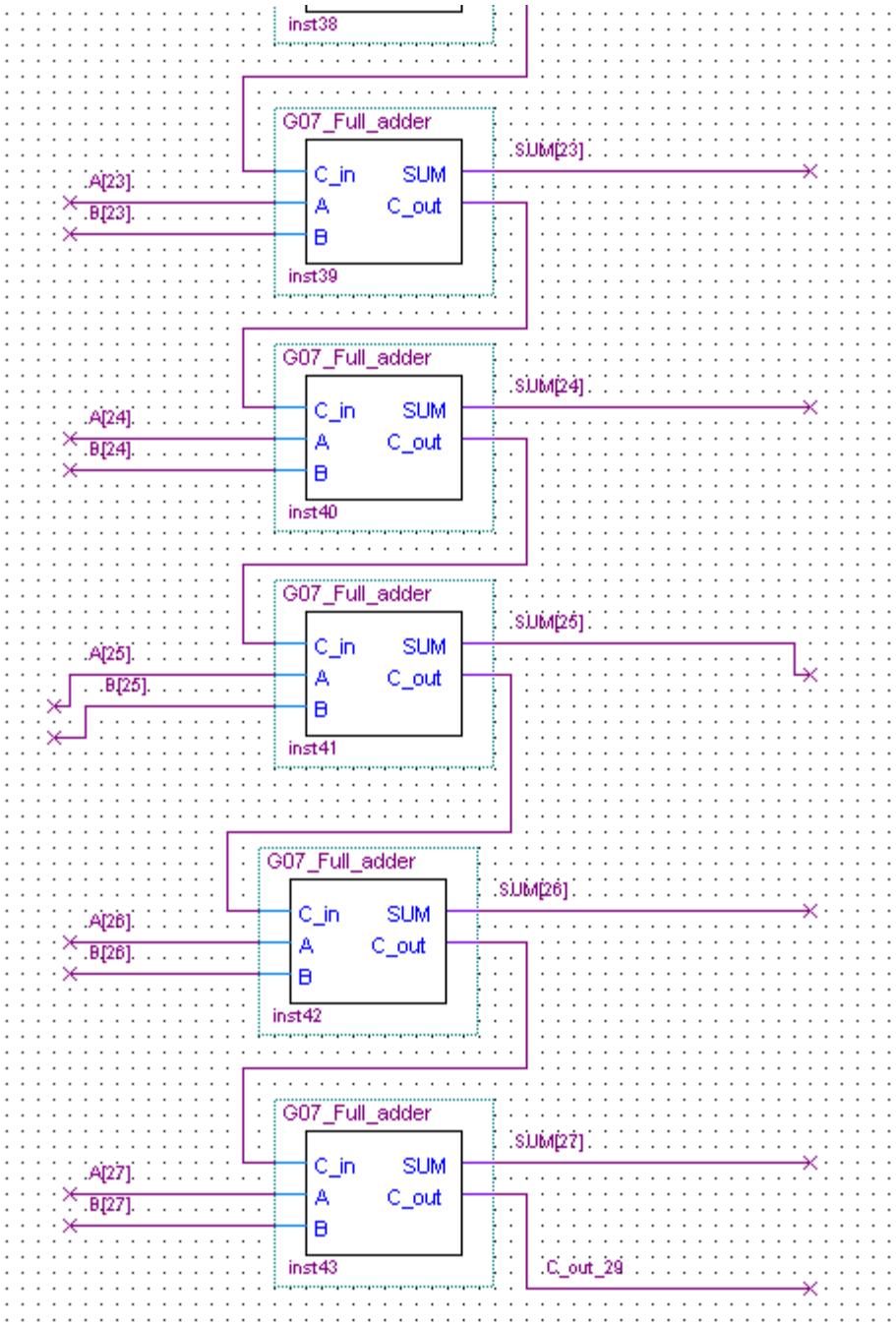
In order to build a 28 bit adder, we linked 28 of these building blocks. For the adder handling the least significant bit, the C_in got connected to ground (since there can't be a case of carry in for the LSB)



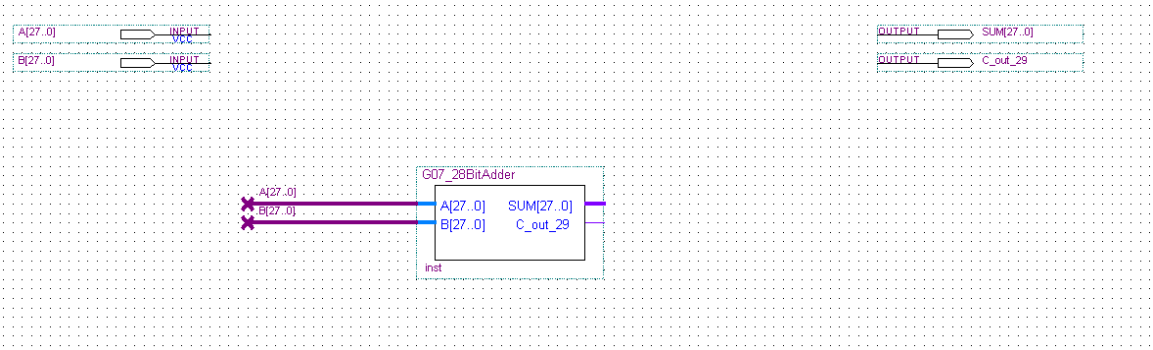
The subsequent adders were linked to each other with the carry line; C_out of the less significant bit connected to C_in of the more significant bit.



The adder handling the most significant bit had a carry out line as a system output, letting the user know that the addition went into overflow.



This circuit's main function is to perform the addition between two 28 bit numbers, A and B.

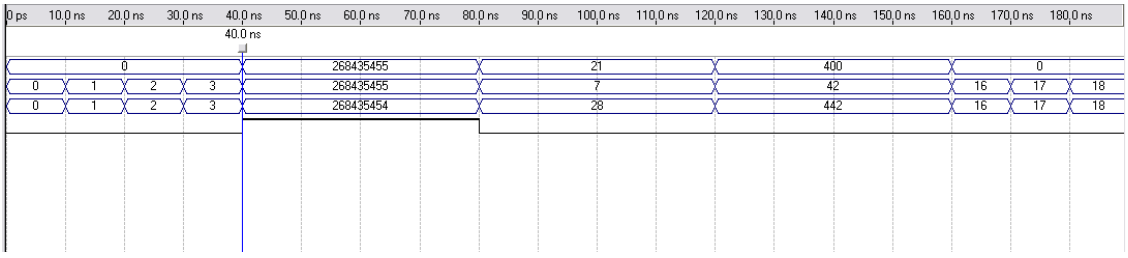


The circuit's inputs are A and B; both 28 bit bus lines to input the numbers to be added. The outputs are SUM, a 28 bit bus line to output the addition result and C_out, a 29th bit to express the result of a 28 bit overflow. Note that connecting SUM to pins 0-27 and C_out to pin 28 of a 29 bit register will generate a complete addition answer.

Circuit testing and simulation

For the full adder, the circuit was tested with the Quartus compiler to make sure that there were no loose connections. Since it is a rather simple and excessively well documented circuit, we had no doubts that it would perform the function required, so we didn't run wave function test. Instead, we encapsulated as a symbol which would be used as a building block for the 28 bit adder. By testing the 28 bit adder, we effectively tested both the 28 bit adder and the full adder. We figured that if we ran into issues with the 28 bit adder, we could easily come back, troubleshoot and edit the encapsulated full adder independently of the 28 bit adder. By doing so, we reduced testing time by a factor of 2 (since everything worked well with the 28 bit adder)

The 28 bit adder was first tested by running it into the software's compiler, which returned no errors. Afterwards, we ran a wave test using different inputs to make sure that the sum outputted was accurate, and that the C_out line was high when required. We performed the following operations:
0 + 0 = 0 (all inputs low)
2^29-1 + 2^29-1 (all inputs high) = 2^29
21 + 7 = 28
400 + 42 = 442
We also ran the wave function adding 0 to all possible 28 bit values. The wave function is as follows:



By manually evaluating these additions and comparing the true value with the computed value, we came to the conclusion that our 28 bit adder works well. This obviously implies that the encapsulated full adder also functioned as intended.

A description of the circuit's function, listing the inputs and outputs. Provide a pinout or symbol diagram.

•A gate level schematic diagram of the circuit.

• A discussion of how the circuit was tested, showing representative simulation plots. How do you know the circuit works correctly?