Part 3: 8-bit arithmetic adder

# Task description:

Aside from the 8 bit ripple carry adder from the previous task we were also provided with an .vhd file for an 8 bit arithmetic adder (ARITH\_ADDER.vhd). Instead of assigning the bits individually, the arithmetic adder simply used the addition operator + to evaluate the sum, including the carry.

In our prep simulation using ModelSim we used the same 5 test cases as in the task before to verify its correct functionality. Both simulations yielded the same results, although the delay for the arithmetic adder was a constant 5ns for each operation in contrast to the varying delay times for the 8 bit ripple carry adder.

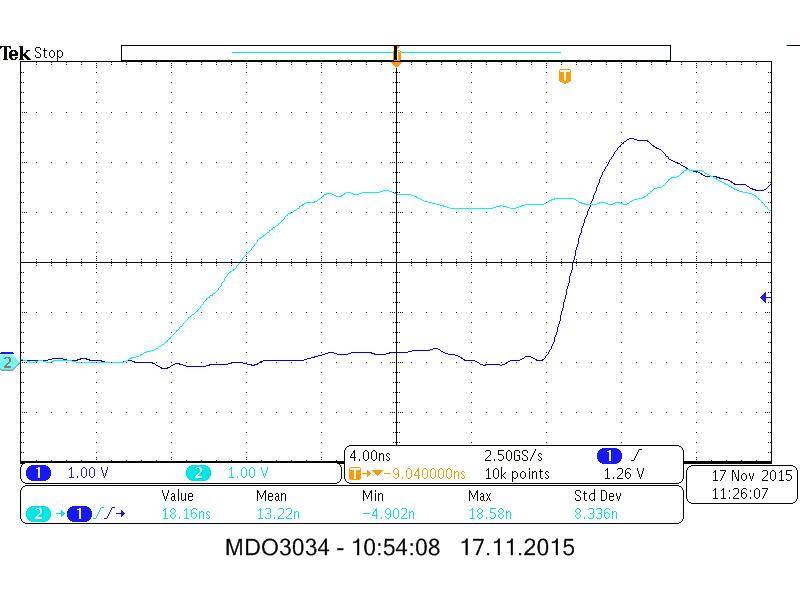
# 3.1 Verification of test cases

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | Simulation Result | Lab Result |
| 05 | F3 | 0F8 | 0F8 |
| 3C | 9E | 0DA | 0DA |
| B5 | 97 | 14C | 14C |
| CD | EF | 1BC | 1BC |
| FF | FF | 1FE | 1FE |

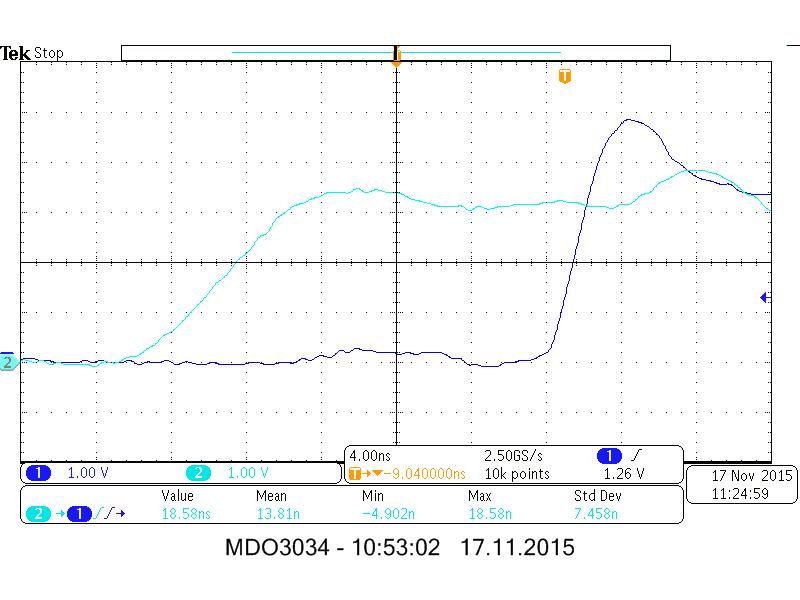
Our implemented adder on the CPLD gave us exactly the same results that we experienced in the simulation, thus it is safe to presume that the adder was synthesized correctly and works in the desired way.

# 3.1 Time delay measurements

For the measurement of the time delay we used an oscilloscope where we connected 1 channel to the input of the carry and the second channel to one of the outputs (S[7] or C\_IN). With the use of the delay measurement of the oscilloscope as well the single shot method we were able to capture the following screenshots:



S[7] time delay



C\_IN time delay

Again, the carry bit experienced a slightly higher but negligible delay time (18.5ns) compared to S[7] (18.2ns)

# 3.2 Resources

The synthesis of the .vhd and the .ucf file outputted us a fitter report and a timing report, where information about the implementation on the CPLD was stored.

Function blocks

Taken from this report, we merely used 2 out of the 8 available function blocks.

Macro cells

Each function block contains of 16 macrocells, of which the function blocks in use were occupying 8 each.

Product terms

One function block can make use of up to 56 product terms. For our implementation only 25 per active function block were used.

When we compare the 2 adders upon their delays and resources, it becomes obvious that even though their architecture in their .do files differ, the implementation on the CPLD appears to be the same, as the delays are similar as well as the usage of available resources.

Our assumption is that the synthesis tool is responsible for this, as it acts as a link between the different .vhd files and the identical fitter reports.

# 3.3 Timing report

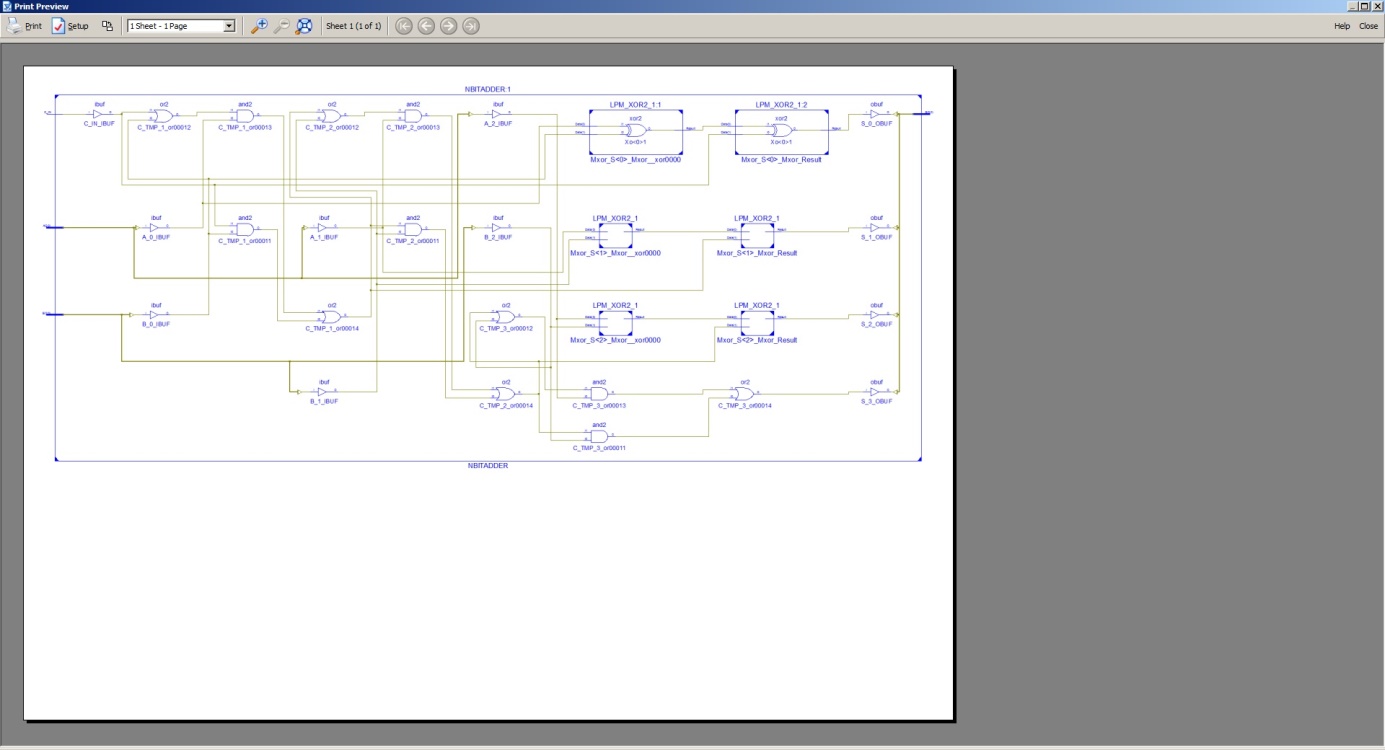
Instead of implementing the two adders for an N of 2, we used N=3 in the lab, yet we conclude that this minor change won’t have any difference on the outcome of this experiment.

The timing reports for the two adders showed to be identical, once again supporting our thesis from 3.2 where we assumed that both adders were synthesized in the same manner.

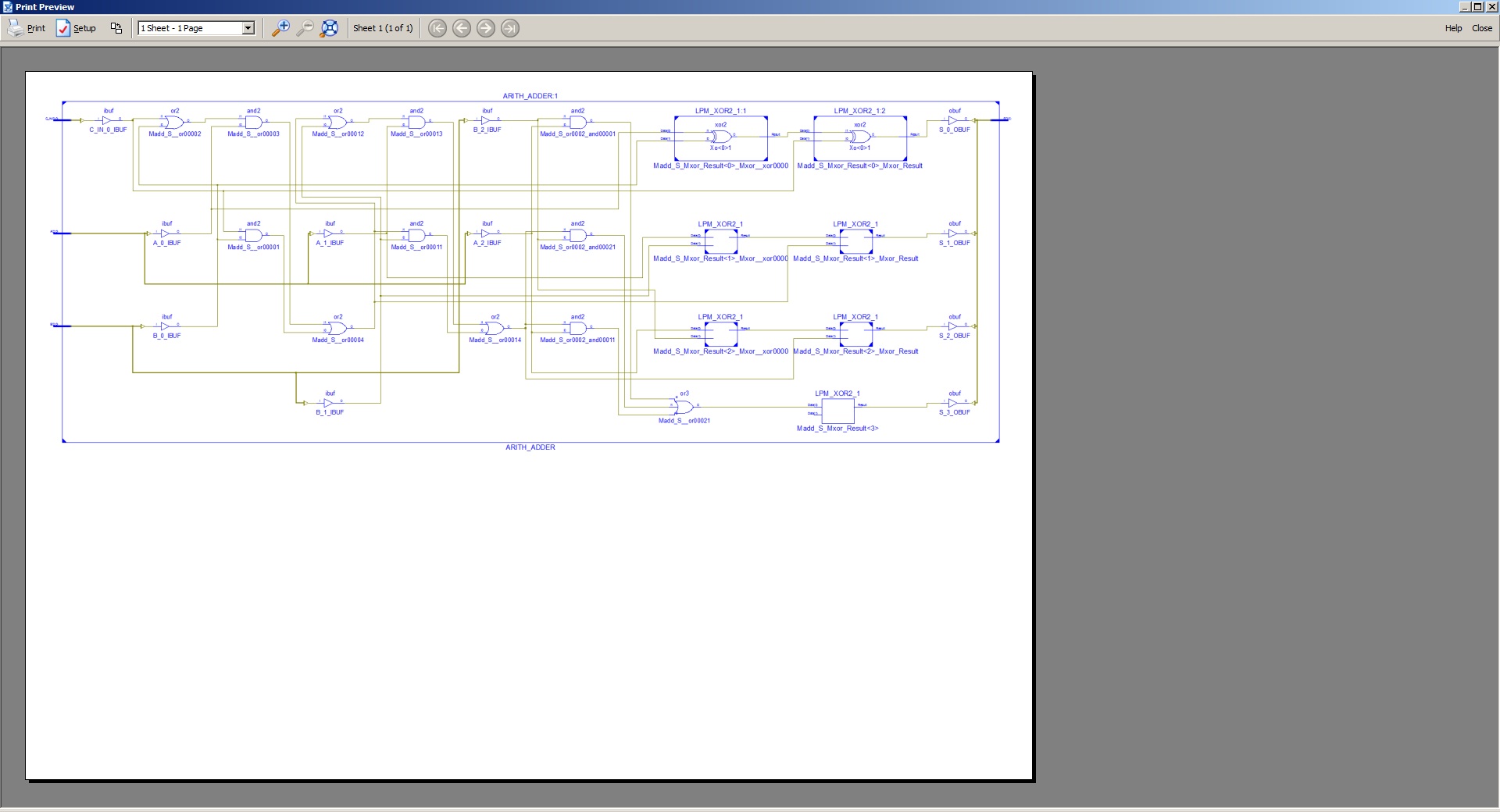
# Extra task

This task required us to look at the RTL schematics as well as the equations view from the fitter report of the synthesized adders from 3.3.

3 bit ripple carry adder schematic:



3 bit arithmetic adder schematic:



Both schematics prove to be identical.