TSEK06
Final Project Report

Group 5

Editor: Johannes Klasson

Version P1A

Status

Reviewed	Johannes Klasson	2016-05-26
Approved	Martin Nielsen-Lönn	-

PROJECT IDENTITY

VT, 2016, Group 5
Linköpings Tekniska Högskola, ISY

Group members

Name	**Responsibility**	**Phone**	**E-mail**
Johan Isaksson	Project Leader	070-2688785	johis024@student.liu.se
Johannes Klasson	Document Manager	073-8209003	johkl226@student.liu.se
Jonas Tarassu	VLSI Designer	070-5738583	jonta760@student.liu.se
Alexander Yngve	VLSI Designer	076-2749762	aleyn573@student.liu.se

Customer: ISY
Contact at customer: Martin Nielsen-Lönn
Course resposible: Atila Alvandpour
Consultant: Martin Nielsen-Lönn

Contents

Document history

Version	**Date**	**Changes**	**Performed by**
P1A	2016-05-26	First draft	Johan Isaksson

1 Introduction

This document describes the state of the 16-bit Kogge-Stone adder project in the course TSEK06 after finishing the transistor level design. The meaning of transistor level is that the every basic logic gate is implemented with 0.35 µm CMOS transistors. The main reason for doing this is to be able to simulate all logic to make sure that everything works as intended. Updated block diagrams can be found in section **??**, simulation results in section 3 together with appendix B. A small risk analysis can be found in section 5. In appendix **??** the time plan for the next phase can be found.

2 Project Description

The objective of this project was to implement a 16-bit Kogge-Stone Adder in 0.35 µm technology which from now on will be called KS adder. The KS adder belongs to the family of parallel-prefix adders which reduces the critical path compared to regular ripple-carry adders. The cost for this is paid in area and complex routing which prolongs development time.

Fast adders are very important in all types of integrated circuits since almost all algorithms one can come up with consists of at least one one addition.

The official goal was to reach a speed of 200 MHz but the group decided on an unofficial goal to reach a speed of 400 MHz.

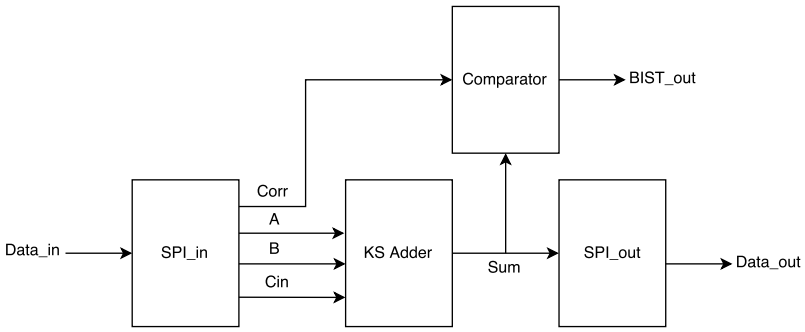

Figure 1 – Block diagram of the system first iteration.

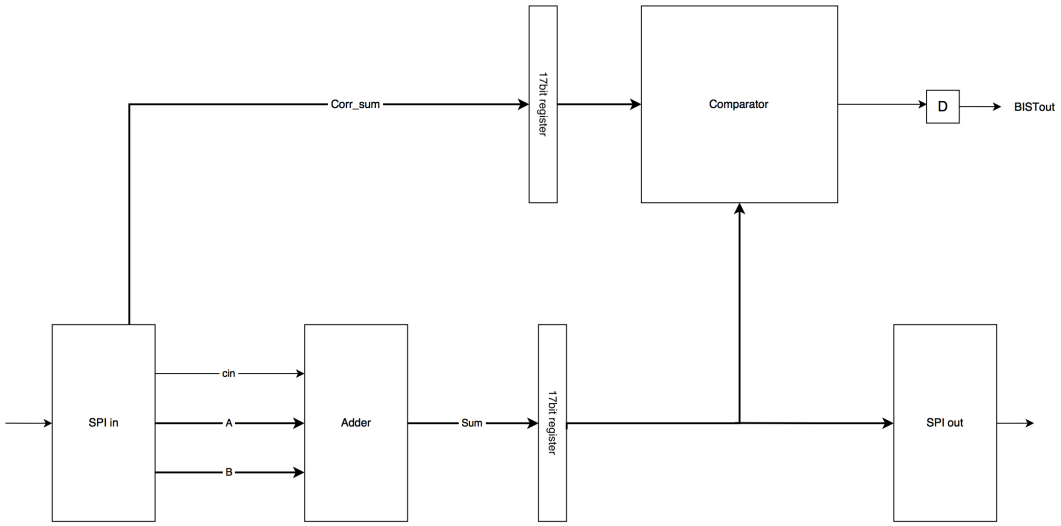

Figure 2 – Block diagram of the system second iteration.

In figure 1 the block diagram of the system after the first design phase is depicted. One can note that the signal goes from one block to the next directly. In figure 2 some registers have been introduced to make sure BISTout stays the same between clock pulses when new data is evaluated and to give a more stable signal to the comparator and SPI-out unit. In the final iteration additional register have been inserted between SPI in and the adder to cut the path from the registers in the SPI unit to the adder and to make sure that all bits in the same word arrives at the same time to the adder. The block diagram from the last iteration can be seen in figure **??**.

3 Simulation Results

This section describes the high level simulation results. All files referenced to in this section can be found in the attached zip-file.

3.1 SPI In

The first thing to test in the system is where it all begins, at the input. The basics of it can be seen in test_spi_receive. As can be seen, as soon as the SPI_enable signal goes low and the SPI_clk starts we start to receive one bit on every positive edge. The data is then shifted trough all of the 16 registers. As the 16th bit is shifted in, the first load signal is triggered. Every 16 bits after that another load signal is triggered, and this can be seen in test_spi_load. One can also see that once the last load signal is triggered, SPI_enable goes high again.
The last thing in the SPI_in module to test is how the data travels out of the PRBS registers. This can be seen in test_spi_prbs. One important thing to note is that as soon as SPI_enable goes high, the registers are triggered on the system clock. As one can see in the figure, the first bit is ready for a long time, and as soon as the last bit is ready, we start to add at full speed. And after the four bits are done the system continues to add the pseudo random numbers.

3.2 Kogge-Stone Adder

The simulation of the adder can be seen in Appendix B Fig. **??**. The input sequence for A, B and cin is `0xFFFF`, `0x0000` and `1` respectively. This sequence should generate the worst case propagation delay which in our case, as can be seen in the simulation, is 1.36 ns. The reason to why the propagation delay is measured between the LSB bit of the A signal and cout is because cin is generated through an ideal voltage source which is set to high in this case. The A signal on the other hand is generated through a clocked test generation block written in VerilogA. The BISTout signal, which indicates if the addition was carried out successfully or not goes high two clock pulses after the data was fed into the adder, just as intended.

3.3 Comparator

The simulation of the comparator was left out since the comparator block is part of the test bench for the adder. All outputs from the adder goes into the comparator together with a Corr-sum signal. If these two signals matches, meaning the addition was carried out correctly in the adder, BISTout goes high.

3.4 SPI Out

The critical parts of this module are the events after a transition on the spi enable signal.

In Fig. **??** a simulation of the behaviour when the SPI enable goes high. The simulation shows that the buffering of the signals successfully achieved satisfying rise and fall times For the enable signals.

When spi enable goes low, the four enable signals are the same as the spi clock which can be seen in the image **??**.

3.5 Top Level

One can get a overall picture of the behaviour of the system by looking at at the simulations in the order spi_receive, spi_prbs and last the spi_out. This is possible because all this simulations are part of a bigger one. The Kogge-Stone adder was left out here for simplicity.

4 Evaluation Plan and PAD List

Table 1 shows the pin assignments for the chip.

Table 1 – Pin assignments

Name	Direction	Type	Description
Vdd1	INOUT	Analog	Will provide most of the system with power and will be a steady 3.3 V.
Vdd2	INOUT	Analog	Will provide the adder with power and it might vary from 3.3 V downto threshold-voltage.
GND	INOUT	Analog	Ground.
Clk	IN	Digital	This is the clock for the adder, some registers and control logic. Should have a frequency of at least 200 MHz at 3.3 V. Will be lower as we decrease the voltage of Vdd2.
SPI_clk	IN	Digital	This clock is used by the input and output unit and should be at least five times slower than the system clock. Should also be low if SPI_en is inactive.
SPI_en	IN	Digital	Active low. Should go high on the first negative flank of SPI_clk after the last value is read.
SPI_in	IN	Digital	Updates it's value as soon as SPI_en goes low, and should have it's value ready on the first positive flank of SPI_clk, since this is when we read the value. The value of SPI_in should then be updated on every negative flank of SPI_clk.
SPI_out	OUT	Digital	The data is available for read on the first falling edge of SPI_clk after SPI_en has gone active.
BISTout	OUT	Digital	If the IN-data is correct, BISTout should be constant high after the first addition is done until the the PRBS-bit is set.
Cin	IN	Digital	Used to measure propagation delay.
Cout	OUT	Analog	Used to measure propagation delay.
Sum15	OUT	Analog	Used to measure propagation delay.

5 Risks

During the transistor level design there hasn't been any considerable delays. The only thing that is an issue is that all members in the group has quite different schedules which sometimes can hinder cooperation. However the group has solved this by using good tools for project tracking (Trello) and communication (Slack). This has helped considerably with the delegation of tasks and keeping track of what needs to be done.

During the next phase of the project, cooperation will be more important since the tasks will be harder. We will need to plan ahead and schedule occasions where we all can meet and work together.

6 Project Evaluation

During the course of the project we gained a lot of experience related to tools, design and also cooperation.

6.1 Cooperation

The group has functioned very well during the whole project except for two weeks during the beginning of the layout phase. There were some miscommunication since half of the group were on holiday the first week while the other half were on holiday during the second week of this period. This later led to some integration issues.

6.2 Tools

In retrospect, less time should have been spent on Verilog simulations and more time on schematic simulations. The Verilog models were very inaccurate and we had no idea what values to choose for propagation delays and rise times etc. The schematic simulations were much more accurate and at the same time quite fast. Since the simulations when doing layout were very slow, a lot of time was spent on waiting for results. Doing more design work at the schematic level would have led to more effective work and possibly a better overall design.

Another tool related issue encountered was the problem with configurations. Using configurations in Cadence seemed like a very useful feature which could help us keep a clean structure of the project by having several schematics and layouts of different verions of the same cell in one place, instead of creating several very similiar cells. It turned out however that it was very hard to maintain and the configuration editor in Cadence randomly crashed all the time when creating somewhat advanced configuations. We should have stuck to having only one schematic and one layout view in each cell, even though this led to many cells.

Quite early in the project we researched the possibility of having automatic tests of all cells in the project using a tool called Ocean. Unfortunately we could not get it to work, but it would have been very useful and could have helped us pinpoint errors with ease.

6.3 Design

The design work went smoothly until the integration phase. We expected some hardships during integration, but maybe not to the extent we experienced them. Things seemed to stop working for no apparent reason. Cells that worked yesterday, didn't function today and so on. Integration is hard, this is a lesson all of us will remember.

Looking back, there are several things we could have done much better. Clock distribution, interconnects and floor planning should have been considered much earlier in the project, during the same time as when layouting the basic gates. Floor planning and interconnects could have been done much better if we would have communicated better during the early layout phase, but due to the reasons mentioned in 6.1 this didn't happen. This led to a situation were individual parts functioned well but weren't optimally adapted to eachother. For example ome components of the chip had different widths and therefore the power rails and data interconnects couldn't be aligned easily. If the floor planning had been done earlier in the layout phase, we could also have designed the basic blocks better. When we designed the basic blocks we did it almost exclusivly using metal 1 and 2 layers, in order to leave room for power supply and clock signals on metal 3 and 4 layers. If we would have considered the power supply and clock signals at the same time as when we designed the basic blocks, we could probably have ended up with a tighter design.

Late in the project we discovered some timing issues probably related to the SPI control logic, but we didn't have time to make a new easier and faster control logic design since the layout was already done and the simulations took considerable time. If we had done more testing in the schematic stage, with more realistic loads and possible doing worst case corner testing in schematic

simulations, this problem would have been discovered earlier and the control logic could have been redesigned. The problem was eventually solved without a new design but not in an optimal way.

A Time Report

B Simulation results