Processor Project Documentation

# Log

27.11.2016 Initial version of the documentation

Contents

[Introduction 4](#_Toc468035135)

[Project Directory structure 5](#_Toc468035136)

[Top Level Structure 6](#_Toc468035137)

[Control Block 7](#_Toc468035138)

[Registers 8](#_Toc468035139)

[Arithmetic Logical Unit 9](#_Toc468035140)

[Adder Subtractor 9](#_Toc468035141)

[Multiplier 9](#_Toc468035142)

# Introduction

This project is completely done by Toni Lammi. The meaning of this project is to learn basics of vhdl and produce a virtual 4-bit processor on Altera Cyclone II FPGA board. Initially the processor is meant to be a 4-bit processor but the designs are designed to be easily scaled for larger bit-counts. This is done by using generic statements. If a block doesn’t support scalability, it is named with postfix of bit count: e.g. multiplier\_4bit.vhd.

# Project Directory structure

The project contains following directories:

db/ contains all temporary files created by the synthesis

do\_files/ has all the .do files used by ModelSim

doc/ cointains all the documentation not included in .vhd files (e.g. this file).

sim/ is where the model sim is started from

syn/ directory from which the Quartus synthesis tool is run.

tb/ contains all test bench files. Files are named with DUV’s name and postfix \_tb.vhd

vhd/ contains all .vhd files.

work/ is the working directory for MultiSim

# Top Level Structure

This processor consists of 3 main blocks: Control Unit, Registers, and Arithmetic Logical Unit. The first one is the block that controls the whole functionality of the design. Registers contain the values the processor works on: e.g. stack pointer and program counter. Lastly, ALU is the block that handles all the logical and arithmetic operations.

# Control Block

# Registers

# Arithmetic Logical Unit

ALU takes in 3 values with custom bit width. Two of the inputs are operands and one is used for identifying the operation that is done using these inputs. ALU also contains two output busses both with same bit width as the inputs and multiple status flag outputs.

## Adder Subtractor

Block used for both binary adding and subtractíng

## Multiplier

Block for multiplying binary numbers.