FH University of Applied Sciences TECHNIKUM WIEN

Chip Design 1

Specification of Calculator Project
Part 3: Calculator Control Unit

Version: 1.0
Author: R. Höller

Copyright Notice

Introduction

This document describes the calculator control unit “calc_ctrl” which is the main control unit of the calculator project. It provides two operands OP1 and OP2 as well as the type of arithmetic/logic operation to the ALU (Arithmetic Logic Unit). Once the calculation is finished, the result coming from the ALU as well as any special conditions (calculation error, overflow …) will be evaluated by the calculator control unit and forwarded to the IO control unit. See the distance learning letter “Overview of Calculator Project” for a list of all ports of the calculator control unit. It is also worth having a look at the block diagram again.

Specification

The calculator control unit implements either “Variant A” or “Variant B” (depending on your number in the course attendance list) of the user interface, described in the distance learning letter “Overview of Calculator Project”, as a finite state machine (FSM). The state machine will have five basic states: “Enter Operand 1”, “Enter Operand 2”, “Enter Type of Operation”, “Calculate” and “Display Result”. During the states “Enter Operand 1” and “Enter Operand 2” the calculator control unit is responsible to immediately display any changes of the operands using the switches SW0-SW11 on the 7-segment digits via signals “dig0_o(7:0)”, “dig1_o(7:0)”, “dig2_o(7:0)” and “dig3_o(7:0)”. Similarly, any changes on SW12-SW15 must be immediately displayed on the 7-segment displays in the state “Enter Type of Operation”. Only the supported arithmetic/logic operations (according to Table 2 in the distance learning letter “Overview of Calculator Project”) should be displayed! That is, the three rightmost 7-segment digits should be all dark when an operation is selected via SW12-SW15 that is not implemented.

For “Variant A” of the user interface a calculation is started every fourth time when push button BNTL is pressed while for “Variant B” a calculation is started when push button BNTD is pressed. Starting a new calculation is done by the calculator control unit by generating a pulse of a single clock cycle on signal “start_o”, see Figure 1. Afterwards, the ALU performs the selected operation which requires the time t_{CALC} (this time depends on the type of arithmetic/logic operation). Note, that the state of the signals “op1_o(11:0)”, “op2_o(11:0)” and “optype_o(3:0)” must be held stable during t_{CALC} by the calculator control unit!

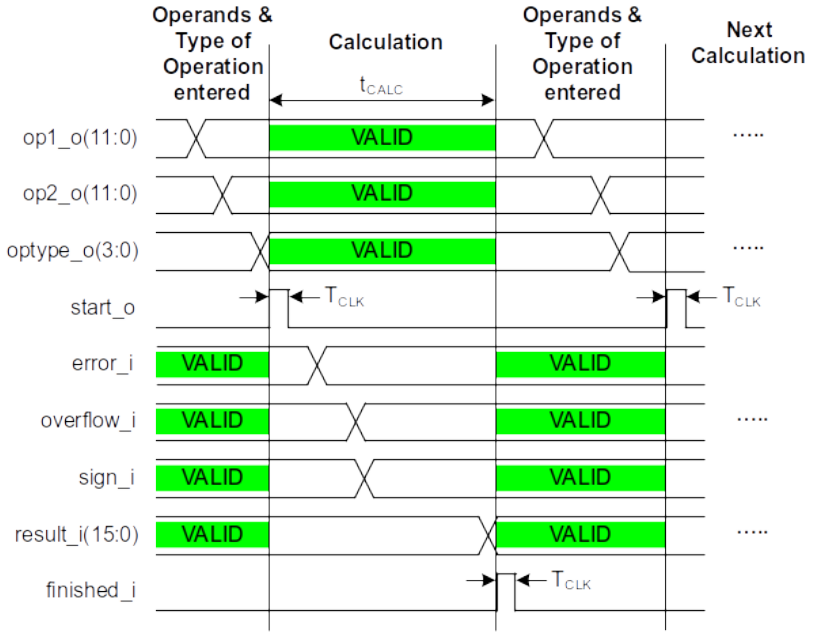

Figure 1: Timing of Interface between Calculator Control Unit and ALU

With a pulse of a single clock cycle on signal “finished_i” the ALU indicates that the calculation is finished. The way how to display the result of a calculation depends on the state of the signals “sign_i”, “overflow_i” and “error_i”, as described in Table 1. It can be assumed that the ALU does not change the state of the signals “result_i(15:0”), “sign_i”, “overflow_i” and “error_i” until a new calculation is started by the calculator control unit via “start_o”.

error_i	overflow_i	sign_i	Description
0	0	0	16-bit result can be directly displayed on the four 7-segment digits
0	0	1	Result of calculation is a negative number
0	1	X	Result of calculation exceeds 16 bits
1	X	X	Calculation error

Table 1: Possible Results of a Calculation

Besides the state machine that handles the user interface (mentioned above) the calculator control unit needs to implement a decoder for each of the four digits (dig0, dig1, dig2 and dig3) which controls the 7-segments and the decimal point of a digit. The decoder has to consider the state of the state machine as well as the signals “result_i(15:0)”, “sign_i”, “error_i”, “overflow_i” and “swsync_i(15:0)”, see Table 2.

FSM State	error_i	overflow_i	sign_i	dig3_o(7:0)	dig2_o(7:0)	dig1_o(7:0)	dig0_o(7:0)
Enter Operand 1	X	X	X	'1.'	hex digit defined by swsync_i(11:8)	hex digit defined by swsync_i(7:4)	hex digit defined by swsync_i(3:0)
Enter Operand 2	X	X	X	'2.'	hex digit defined by swsync_i(11:8)	hex digit defined by swsync_i(7:4)	hex digit defined by swsync_i(3:0)
Enter Operation	X	X	X	'o.'	abbreviation shown in Table 2 of distance learning letter "Overview of Calculator Project" as defined by swsync_i(15:12)		
Display Result	0	0	0	hex digit defined by result_i(15:12)	hex digit defined by result_i(11:8)	hex digit defined by result_i(7:4)	hex digit defined by result_i(3:0)
	0	0	1	'-'	hex digit defined by result_i(11:8)	hex digit defined by result_i(7:4)	hex digit defined by result_i(3:0)
	0	1	X	'oooo'			
	1	X	X	'Err '			

Table 2: 7-segment Decoder

Do not forget to turn on LED15 in the state "Display Result" as described in Section "User Interface of the Calculator" of the distance learning letter "Overview of Calculator Project".

How to Proceed?

The next steps in the project are as follows:

- Write a VHDL entity for the calculator control unit, name the file, for example, "calc_ctrl_.vhd" and store it in the "vhdl" sub-folder of your project directory. The entity ports can be found in the distance learning letter "Overview of Calculator Project". Have a look at the block diagram (which is also included in the distance learning letter "Overview of Calculator Project") to understand, how the calculator control unit communicates with other units in the design.
- Write a VHDL architecture for the calculator control unit, name the file, for example, "calc_ctrl_rtl.vhd" and store it in the "vhdl" sub-folder of your project directory. Break down the functionality into smaller sub-blocks (user interface state machine, 7-segment decoder …) and decide which of these blocks need to be coded as combinatorial logic and which of them need storage elements (registers). You can also create VHDL sub-components for certain pieces of functionality but since the complexity of the calculator control unit is not that high, it is rather recommended that you include everything that is described in this document in a single architecture.
- You can also create a VHDL configuration if you like to, but this is completely optional!
- Create a VHDL entity/architecture pair (and an optional configuration) for the testbench of the calculator control unit, name the files, for example, "tb_calc_ctrl_.vhd" and "tb_calc_ctrl_sim.vhd" and store them in the "tb" sub-folder of your project directory.

- Write “do-“scripts to compile and simulate the calculator control unit as described in the distance learning letter “Introduction to ModelSim-Intel FPGA Starter Edition” and store them in the “sim” sub-folder of your project directory.
- Simulate the calculator control unit using ModelSim and fix all bugs that you find.
- If the calculator control unit was tested successfully, proceed with the distance learning letter “Part 4: Arithmetic Logic Unit”.

Version

Version 1.0	Update to entire document for CHIP1

If you find mistakes or inconsistencies, please report them to the course supervisors via email. Thank you!