

Project Assignment #2**DUE: Wednesday, Apr 9th**

- 1) Copy the file **pr_step2.vhdl**
from directory `~degroat/ee762_assign/` or download from the website
to your local directory.
- 2) Write and enter an ENTITY and ARCHITECTURE (**using only concurrent signal assignment statements**) for the carry_chain unit. This time, use 5 ns for the delay of the carry out from the carry in.
- 3) In this exercise we will also use the generic functional block that we created in PS1 and should be in your work library.
- 4) Write and enter an ENTITY and ARCHITECTURE for a 1 bit slice of the ALU.
Make this a structural architecture. This architecture will only contain component instantiations. Don't forget both declaration and configuration of the components. When you configure a component you need only **one** declaration regardless of how many times you use the component. You should have 2 COMPONENT declarations, 2 FOR ALL configuration declarations, and a couple of internal signals. You will then have 4 component instantiations in the architectural part; 3 generic units and 1 carry unit.
- 5) In the testbench architecture enter the **declaration** and **configuration** of the 1 bit ALU, and then instantiate it. Take note of the information provided. You also need to instantiate the 1 bit ALU in the testbench..
- 6) Compile and simulate the design. You need to run the test bench for 2400 ns to allow all tests to run. **IMPORTANT: Before you do any simulation or bring up any simulator windows, such as the waveform window, bring up "simulator options" and reset the default radix to "hexadecimal" and the default run to 100 ns (100,000,000 fs).**
- 7) Turn in:
 - a) copy of your VHDL source code.
 - b) copy of part of the waveform (0 to ~650 ns)
 - c) copy of the complete waveform from 0 to 2400 ns
 - d) copy of a file listing the complete results of the simulation.

The following signals should be listed giving a listing looking something like the following. These are the signals in the testbench.

BE SURE TO COLAPSE DELTAS ON LISTING

ns	delta	oper	pval	kval	rval	a	b	cin	zout	cout
0	+0	OP_A	C	F	C	0	0	0	0	0
0	+1	OP_A	C	F	C	0	0	0	0	0
100		OP_A	C	F	C	0	1	0	0	0
200		OP_A	C	F	C	1	0	0	1	0
300		OP_A	C	F	C	1	1	0	1	0
400		OP_B	A	F	C	0	0	0	0	0
		.								
		.								
...		OP_BMINA	9	2	9	1	1	0	0	0

