So far you have modeled and FPGA synthesized a single and 8-bit 2-to-1 and 4-to-1 multiplexer. These are essential building blocks of the Arithmetic Logic Unit (ALU) of any computer.

In class a discussion of modern ALU design will be held. In this assignment, additional building block components will be implemented.

Write and test the VHDL for an 8-bit full adder (single bit). The testbench should test this architecture but not exhaustively. The design should be written in a dataflow fashion. Vectors for A should be 00000000, 11111111, 01010101, 10101010, 01110111, 11101110. Vectors for the B should be the same and each combination of A and B used, a total of 36 inputs. These should be done when the carry in is 0 and then repeated when the carry in is 1.

In class we will discuss how to make this testbench easy to write.

Compile your VHDL code. Modify the testbench of the ha and fa to test your code and simulate so that all cases are simulated. Prepare a report showing your VHDL results.

Start up Quartis and perform a synthesis on your design. In your report be sure to show the circuit synthesized by Quartis ala the sample report.

Submit it to the dropbox.

A possible testbench:

ENTITY fa8tb IS

END fa8tb;

ARCHITECTURE one OF fa8tb IS

--Declare component and configure

--Local declarations

SIGNAL a,b : bit\_vector(7 downto 0);

SIGNAL cin,cout : bit;

BEGIN

--instantiate component

--apply stimulus

PROCESS

BEGIN

Cin <= ‘0’;

a <= “00000000”; b<= “00000000”;

WAIT FOR 10 ns;

b<= “11111111”;

WAIT FOR 10 ns;

b<= “01010101”;

WAIT FOR 10 ns;

b<= “10101010”;

WAIT FOR 10 ns;

b<= “01110111”;

WAIT FOR 10 ns;

b<= “11101110”;

WAIT FOR 10 ns;

a<=”11111111”; b<=”00000000”;

WAIT FOR 10 ns;

--repeat b signal sequence

--repeat for all these choices for a

--then set cin<=’1’ and repeat all of the a and b stimulus

--this setup is easier than trying to be creative with it

END;