Lab #8 – Design of an Arithmetic and Logic Unit

An Arithmetic and Logic Unit (ALU) is a combinational circuit that performs logic and arithmetic micro-operations on a pair of n-bit operands (ex. A[3:0] and B[3:0]). The block diagram of such an ALU is depicted in Figure 1. The operations performed by an ALU are controlled by a set of *function-select* inputs.

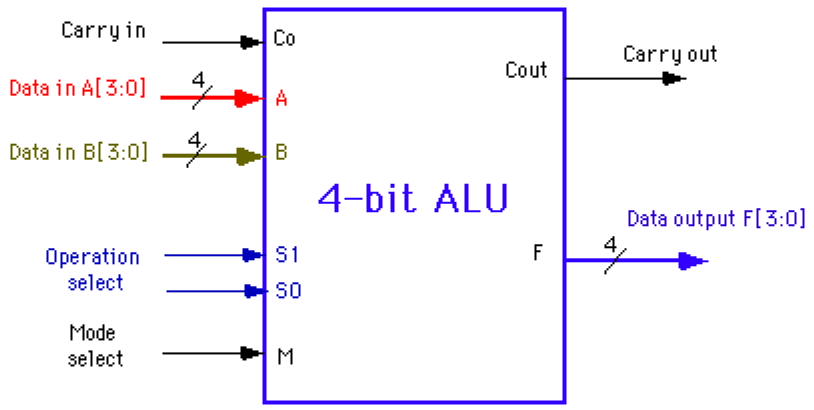

Figure 1: Block diagram of an ALU

In this lab you will utlize schematic capture of Xilinx tools and a hierarchal approach, to design and implement a 4-bit ALU in an FPGA. Your ALU should have 3 function-select inputs: Mode M, Select S1 and S0 inputs. The mode input M selects between an Arithmetic (M=0) and Logic (M=1) operation. The functions performed by the ALU are specified in Table 1.

Table 1: Functional operation of an ALU

Arithmetic Operations:	**Logical Operations:**
Add: A + B	AND: A ^ B
Subtract: A - B	OR: A v B
Increment: A+1	XOR: A ⊕ B
Decrement: A - 1	NOT: A'

In addition, your circuit should appropriately generate N (sign), Z (zero), C (carry), and O (overflow) status signals.

Design strategy:

When designing the ALU we will follow the principle "Divide and Conquer" in order to use a modular design that consists of smaller, more manageable blocks, some of which can be re-used. Instead of designing the 4-bit ALU as one circuit we will first design a one-bit ALU, also called a ***bit-slice***. These bit-slices can then be put together to make a 4-bit ALU.

There are different ways to design a bit-slice of the ALU. One method consists of writing the truth table for the one-bit ALU. This table has 6 inputs (M, S1, S0, C_0, A_i and B_i) and two outputs F_i and C_{i+1}. This can be done but may be tedious when it has to be done by hand.

An alternative way is to split the ALU into two modules, one Logic and one Arithmetic module. Designing each module separately will be easier than designing a bit-slice as one unit. A possible block diagram of the ALU is shown in Figure 2. It consists of three modules: 2:1 MUX, a Logic unit and an Arithmetic unit. This approach is recommended.

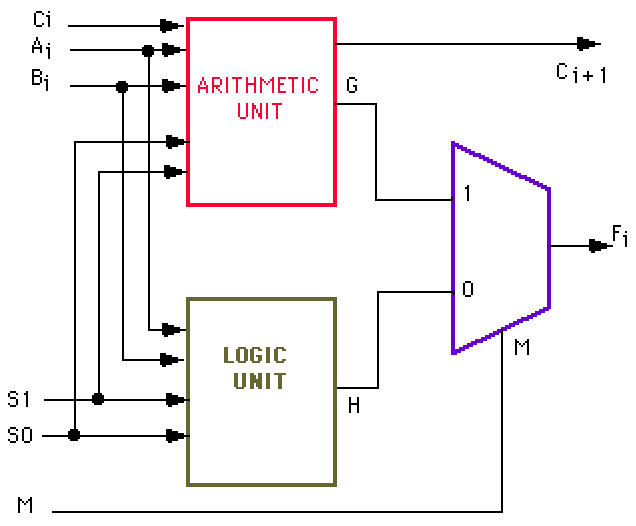

Figure 2: Modular blocks of an ALU

Tasks:

1. Become familiar with Xilinx toolset by going through the Xilinx tutorial .

2. Design a 4-bit ALU using schematic captures and indicated design specification.

3. Simulate your design using ISIM Simulator.

4. Download your design onto the Xilinx board

5. Verify the circuit and have it signed by your instructor or TA.

Some suggestion to improve Xilinx performance in the lab.

1. Try to use USB flash drive
2. If you use the N drive,
    a. Copy you project files to the C drive (or USB flash drive) BEFORE starting ISE/Xilinx.
    b. Start up Xilinx
    c. Load the project from the local C drive (or USB drive).
3. ***If working from the C drive***... Save frequently to the C drive but use Windows explorer to copy the project folder to the N drive. Do not use Xilinx to save directly to the N drive. This may cause a problem if, in the act of saving from within Xilinx, it changes the working environment from "C" to the network.