

## Lab 2 – Co-emulation using Xilinx FPGA's

Due Tuesday 10-10-2000

**Abstract:** The purpose of this lab is to familiarize you with co-design/emulation strategies using programmable technologies. In this lab you will take the ALU that was designed in the 6<sup>th</sup> Xilinx tutorial and implement it on an FPGA. You will then create a C program to interface with this design and take appropriate estimations of timing for the analysis of your design.

### 1. Setup

This lab will use the Foundation Series Xilinx Software (found on the Windows NT machines), the preprogrammed interface that I will provide (via the class web-site), and a Xess XS40 board. You can download both the Xilinx parallel port interface and the C program from the course web-site.

To install the project, simply go into Xilinx and go to File->Restore Project. Then select the "489lab2.zip" file that you got off the web-site and it will uncompress it for you. You cannot program the XS40 boards with the Xilinx foundation software, so you will have to use the GXSLOAD program. Simply open the program and drag the .bit file generated by Xilinx during implementation into the program.

The C code for the project can also be found on the web-site and is called "commune.zip". The code works best when you extract it into the "C:\Program Files\Microsoft Visual Studio\MyProjects" directory. Use the Microsoft Visual C++ compiler when working with C programs on the Windows NT machines.

### 2. How it works

The parallel port interface is made to be as simple as possible, so that it can accommodate a wide array of applications. If you would like to modify to way it works, feel free. The current implementation has 4 registers inside the FPGA, which can be read from and written to by the parallel port interface. The interface uses a Read, Write, and Reset control signals to manipulate to ASIC design. There is one address register which controls which data register will be written to or read from based on the control signals. More information on the specifics of this can be found at: <http://www.xess.com/ho03000.html>

### 3. The Calculator

Use the Verilog ALU that you designed in the Xilinx tutorials and design a calculator program that does all of the math functions on the FPGA. This can be accomplished by tying the inputs, control and output of the ALU to each of the four registers. Turn into me a printout of your C code, the schematic design, and a description of how it works.

### 4. Advanced Calculator

Implement multiplication and division in your ALU so that they require multiple clock cycles to complete. Modify the design so that the C program must monitor the ALU control register to find out when the ALU is done. It may be necessary to slow down the clock going into the ALU so that it is slower than the parallel port. Time how long it takes for different operations from the time the C program sends data until it receives the result. Try this for varied clock speeds and put your results in table form. Make sure you turn in printouts of all work as well as a description of its operation. Do not vary the clock speed of the parallel port interface, only the ALU.