www.RetroMicro.com

Created by: Douglas Hodson
If you find this project useful, please email me at
doug@RetroMicro.com

Project Name: xsocXR16

Creation Date: June 2003
Development Board: XESS XSA-100 Plus XStend Version 2
Development Software: Xilinx ISE Version 5.1.03i

Description

The project uses the xr16 cpu Jan Gray designed in the popular XSOC project (see references). In the original project, the XSOC was targeted for the XESS XS40-005XL fpga board that includes SRAM. This project extracts the cpu fully intact and interfaces it to the SDRAM contained on the XSA-100 board.

The XStend board is not required in this project.

The XSA-100 board frequency must be set to 25 MHz!

"C" Compiler and Assembler

The custom xr16 assembler was written by Jan Gray. The "C" compiler was created by modifying lcc (A Retargetable Compiler for ANSI C).

I had problems compiling the original sources as provided in the XSOC project. Quite honestly I don't remember what I had to do to fix the problem, other than I remember that only a few modification had to be made and maybe a few syntax errors had to be fixed. (If I remember correctly, one problem related to compiling for a MS-DOS command prompt verses the newer Windows Command Prompt.)

Nevertheless, executables that do work are located in the lcc directory. To use them effectively, create the following directory tree and copy them there:

C:\Program Files\lcc\4.1\bin

Then add that path to your system PATH environment variable (Right click on "My Computer", click "Properties", and finally click the "Advanced" tab.)

Using the compiler and assembler is straightforward from this point on. For example to compile the fibonacci example, take the following steps.

1. Open a Command Prompt

2. Change to the ……\xsocXR16\progs\fibonacci\xr16 directory

3. Issue the command: “lcc-xr16 –S fib.c” – this will compile the “C” code into assembler.

4. Issue the command: “xr16 –hex=fib.hex –lst=fib.lst reset.s fib.s” – this will assemble first the reset.s code then assemble and concatenate the fib.s code. Two files will be generated. The first is a hex formatted file that can be downloaded to SDRAM using the XESS tools, and secondly a listing file for all the code.

A Little Bit About Clocks and the SDRAM Controller

When interfacing to SDRAM, you drive the controller with a clock signal wired to the clkin port. After that the controller provides three clock signals back for the design to use; bufclk, clk0 and clk2x. They are defined something like this:

- bufclk : simply a buffered version of the "clkin" signal you are feeding to controller. In other words, the clock signal direct from a pin on the fpga chip with a buffer in the stream.

- clk0 : a clock signal generated within the sdram controller itself to account for the printed circuit board (PCB) delays in interfacing to the externally driven sdram memory.

- clk2x : a doubled version of clk0

When using the SDRAM controller, you supply clkin direct from fpga clock pin, then you should use the clk0 signal as the master clock for your design. You should also use bufclk in conjunction with another sdram provided signal, "lock" to determine when clk0 has "locked" on to the master clock you have provided. This is accomplished with a clkdll primitive.

This project uses the latest version of the SDRAM controller provided by XESS. It should be noted that I have found this version substantially better and more reliable than an earlier version.

Project Directory Structure

This project is organized into the following directories:

./ - contains all the files need to synthesize the design.

./config – effectively a backup of all main files in the root directory

./docs – this PDF file. Source files used to build this PDF are located in ./docs/src.

./lcc – Retargetable “C” compiler and assembler for xr16 cpu.

./progs – example programs that can be downloaded and executed by the xr16. If the example is sufficiently complex you might find a “windows” version of the code used to test and determine what the outputs should be. Check out the Fibonacci example. Windows programs are compiled with the excellent free Dev-C++ compiler.

./src – source directory for all HDL files

./temp – temporary directory used during the build process.

Synthesis

The project is built and maintained using two windows based batch files. The first is “make.bat”. It simply issues are the commands required to “compile” all the source files in the ./src directory and eventually generate the chipIO.bit file to downloaded to the fpga.

The second batch file is called “clean.bat”. Its sole purpose is to delete most of the unwanted files generated during the build process.

References

The primary reference for building this project is Jan Gray’s XSOC project. This project is an excellent place to start on how to build a fully pipelined risc based cpu interfaced with a video display and other devices. See the www.fpgacpu.org website for the complete XSOC project and information.