CprE 583 (Fall 2010)
MP3: PowerPC 440 Coprocessor Interface
(Simple Video Image processing)
(Due Friday: 11/5, Midnight)

I. Download and file locations

1. Download MP3.tar.gz from: http://class.ee.iastate.edu/cpre583/
   a) MP3_assignment.doc (or MP3_assignment.pdf)
   b) See III.1 for useful files that you should reference during MP3.
   c) Do one quick read through of this document

2. Uncompress into your U: (home directory)

3. EDK (XPS) project is located here: MP3/ml507_bsb_system.xmp
   a) Set up environment (you MUST do this for every window/terminal you open)
      i. linux32
      ii. bash
      iii. source ~/Xilinx_10_src
   b) To open project: xps ml507_bsb_system.xmp &

4. You should be able to edit your VHDL, C, simulate, build your bitfile, and download your bitfile from EDK (XPS)

5. Viewing the Hardware (Distance Students):
   a) xilinxcam2.ece.iastate.edu: used to see the Monitor for xilinx-1.ece.iastate.edu
   b) xilinxcam4.ece.iastate.edu: used to see the Monitor for xilinx-2.ece.iastate.edu
   c) xilinxcam6.ece.iastate.edu: used to see the Monitor for xilinx-3.ece.iastate.edu
   d) username: xilinxuser
   e) password: drjones

6. How to save a dataset and waveform format in Modelsim
   a) Dataset
      i. File -> Datasets -> (highlight vsim.wlf) -> Save As -> name_file.wlf -> Done
   b) Waveform format
      i. File -> Save Format -> name_file.do

7. How to load a data set and waveform format (Note load dataset first)
   a) Dataset
      i. File Datasets -> Open -> Browse -> (select wanted .wlf file)
   b) Waveform format
      i. File -> Load -> (select wanted .do file)

II. Getting started

1. Before you make any changes, make sure the base design works.
   a. Build the bit file (This will take ~30 minutes the first time)
      i. Hardware -> Generate Bitstream
   b. Compile the lwip_demo application
      i. In the Software Project window
         1. Right click “Project:lwipdemo”,
         2. Select Build Project
   c. Load the bitfile
      i. Device Configuration -> Download Bitstream
   d. Load the lwip application (using XMD)
      i. Debug->Launch XMD
      ii. cd ppc440_0/code
      iii. dow lwipdemo.elf (this loads the binary executable)
      iv. run
   e. Send an image from the PC to the Monitor
      i. Open a new window
      ii. Go to MP3/UDP_SW
      iii. Look at README file to see how to compile Send_ppm_video
      iv. How to send a single image (e.g. 0.ppm)
         1. ./exe_Send_ppm_video 192.168.1.10 0.ppm
   f. After 60 seconds or less, an image should appear on the monitor.
   g. To exit xmd: type
      i. stop
      ii. exit
      iii. NOTE!!!: not exiting xmd correctly can lock the JTAG cable, which may result in having to reboot a given machine
   h. For debugging you can add print statements to the PowerPC code
      i. Recompile after you change a file (e.g. echo.c located in lwipdemo -> Source)
      ii. Open a new window for minicom
      iii. Start “minicom”
      iv. Make sure minicom is set to 9600 baud. (See MP0)
      v. Repeat steps d) and e).

2. Read over echo.c to understand what it is doing
   a. It reads 2 UDP packets that contain 1 row of an image (1920 bytes = 640 pixels * 3 bytes per pixel)
   b. Writes an image row to a memory buffer
   c. Tells the video controller to read from the start of the memory buffer, after all the rows have been written to the buffer

III. (60 pts) Design a PowerPC coprocessor (APU) to Invert or Green Shift an image

You will modify the load and store command used by the Coprocessor to create two versions of the coprocessor. One that inverts the 128-bits that are sent to the Coprocessor during a load, and one that make all of the Green bytes within a loaded 128-bit word get set to 0xFF (i.e. Green Shift, make the image more green). Note there are other ways to green shift the image

You will write/use a small C/assembly program to test your VHDL. This program will be used in simulation, and will be used in hardware to verify your Coprocessor design.

1. Refer to the following documents:
    a. V5_PPC440 Reference guide: http://class.ee.iastate.edu/cpre583/ref/EDK/ug200_V5_PPC.pdf
        i. Chapter 12: "APU Controller" is the chapter you should reference
        ii. Skim over this chapter
        iii. Look over Table 12-1
        iv. You will want to understand the timing for implementing the store and load instructions in VHDL. (Pay special attention to the sections "Load Execution Details" and "Store Execution Details")
            1. **(5 pts)** Draw out timing diagrams for how you expect the signals for the load and store to behave.

2. Where to place your VHDL: MP3/MyRepository/pcores/MyProcessorIPLib/pcores/apu_inv_v1_00_a/hdl/vhdl/apu _inv.vhd (use this name for simulating hardware, and building bitfiles for both the invert and green shift coprocessor. It will make your life easier)
    a. There are a number of ways to implement the VHDL for this design, examples
        i. Statemachine(s)
        ii. Clocked processes
        iii. A combination of i, and ii.
    b. Use the simulation to help guide your hardware design, you may find that the timing of the signals being driven by the PowerPC do not exactly match the expected timing diagram you made in III.1.a.iv.1.

3. Where to place your C code for debugging your VHDL
    a. Applications tab (in XPS)
    b. Project: TestApp_Peripheral -> Sources -> TestApp_Peripheral.c
        i. Note: The code you need is already there. You should not need to modify this code. Unless you want to for some reason.
        ii. Note: Make sure the print statements are commented out for simulation!!
    c. To compile: Right click "Project:TestApp_peripheral"
        i. Select Build Project

4. How to simulate (Coprocessor running the test code):
    a. Make sure Project:TestApp_peripheral is selected to initialize the BRAM
    b. Make sure there are no prints in your TestApp_peripheral.c file!!!
    c. Simulations -> Launch HDL Simulator
        i. Type: c
        ii. Type: s
        iii. Type: w
        iv. Type: run 30 us
        v. All coprocessor (i.e. APU) signals are at the very bottom of the wave window
        vi. See FAQ for adding new signals to the simulation
        vii. You should find that the given stub for apu_inv.vhd locks up after the first load instruction is decoded.

5. (35 pts) Turn in Simulation:
    a. VHDL for the inverter and green shift versions of the coprocessor
        i. apu_inv.vhd
        ii. apu_green.vhd
    b. The software you wrote to test the coprocessor (You should be able to use the same software for both the inverter, and green shift coprocessor)
    c. Simulation Datasets and wave format files to show your design works in simulations for :
        i. apu_inv.do, apu_inv.wlf
        ii. apu_green.do, apu_green.wlf

6. (20 pts) Turn in Hardware
    a. Demo in HW
        i. Start "minicom" make sure the buadrate is set to 9600 (see MP0)
        ii. Make sure the TestApp_Peripheral program is used to initialize the BRAM. This application will start as soon as the FPGA is configured
        iii. If you have updated your test software but not the hardware, then you do NOT have to regenerate the bitfile. After you build the software just
            1. Device Configuration -> Update Bitstream
            2. Device Configuration -> Download Bitstream
        iv. Alternatively to iii, If you already have both bitfiles (i.e. invert, green) already built (and the BRAMs are initialized appropriately), then you can use impact to load the bitfile.

    b. A bitfile for:
        i. Coprocessor configured to Invert (MP3_invert.bit)
        ii. Coprocessor configured to Green-shift (MP3_green.bit)

IV. (40 pts) Integrate coprocessor test code into echo.c

The goal is to use the coprocessor to invert or green-shift an image's color depending on how the coprocessor is configured. Note: You may only want to use 12 of the 16 bytes (128-bits) when integrating. This may help a lot with respect to data alignment (e.g. byte vs. 32-bit integer, vs. 3-byte pixel alignment). NOTE: This part does NOT involve simulation!!

1. Location of echo.c
   a. Project:lwipdemo, Sources

2. You will take the load and store instruction (and any other useful/needed constructs) from TestApp_peripheral.c, and integrate them into echo.c

3. (40 pts) Turn in SW, and a hardware bitfiles that will:
   a. Invert the color of an image
   b. Green-shift the color of a image

FAQ and Common mistakes:

1. **Make sure you have the correct application initialized in the BRAM before loading a bitfile**
   1.1. TestApp_Peripheral: When testing your Coprocessor (APU), with the simple C/assembly program
   1.2. Defualt:ppc440_0_bootloop: When you want to load lwipdemo from the xmd prompt.

2. **How to initial the FPGA's BRAM (This will not rerun the entire hardware generation tool flow, so it is very fast)**
   2.1. Right click on the application you want to store to the BRAM
   2.2. Select initialize BRAM
   2.3. Device Configuration -> Update bitstream
   2.4. **NOTE:** The application has to be very small to load into the BRAM: Do NOT try to initialize the BRAM with lwipdemo, it is too big.

3. **How to add new signals to Modelsim with in the EDK (XPS) environment**
    3.1. Assuming you have don III.4
    3.2. To add signals for apu_inv
        3.2.1. In the workspace window of Modelsim
        3.2.2. dut->apu_inv0->apu_inv0
        3.2.3. In the objects window you should see all the waves located in apu_inv.vhd
        3.2.4. Drag and drop the signals you want into the wave window
    3.3. Save this format
        3.3.1. In Modelsim, select wave window
        3.3.2. File->save
        3.3.3. Browse to the top level directory of MP3
        3.3.4. Save the wave format as MP3_wave.do
    3.4. Now the next time you run the simulation from EDK (XPS)
        3.4.1. Simulations -> Launch HDL Simulator
        3.4.2. c
        3.4.3. s
        3.4.4. In ModelSim: File->load, browse to MP3_wave.do
        3.4.5. The new waves and/or formats you saved should be restored

4. **Exit XMD properly!!! (If not, then you may lock the JTAG cable = Likely have to reboot the machine)**
    4.1. Type
        4.1.1. stop
        4.1.2. exit

5. **XMD cannot connect to the PowerPC**
    5.1. One common cause of this is the PowerPC locking up because your APU is not working correctly in the hardware. For example the APU locks up during execution of the load or store instruction
    5.2. If this is the case then do the following to exit XMD properly
        5.2.1. stop
        5.2.2. exit
    5.3. Again when you want to run the lwipdemo application, make sure the BRAMs are initialized to Defualt:ppc440_0_bootloop

6. **When I try to simulate vsim says it cannot find some libraries**
    6.1. cp modelsim.ini simulation/behavioral/