WIRELESS OPEN-ACCESS
RESEARCH PLATFORM

Building a Sinusoid Generator

Chris Hunter, Siddharth Gupta, & Patrick Murphy

Rice University
WARP Project

Document Revision 1

December 17, 2008

1 Introduction

The goal of this lab exercise is to test a simple model in hardware. The model should be built in System Generator by following the tutorial at:

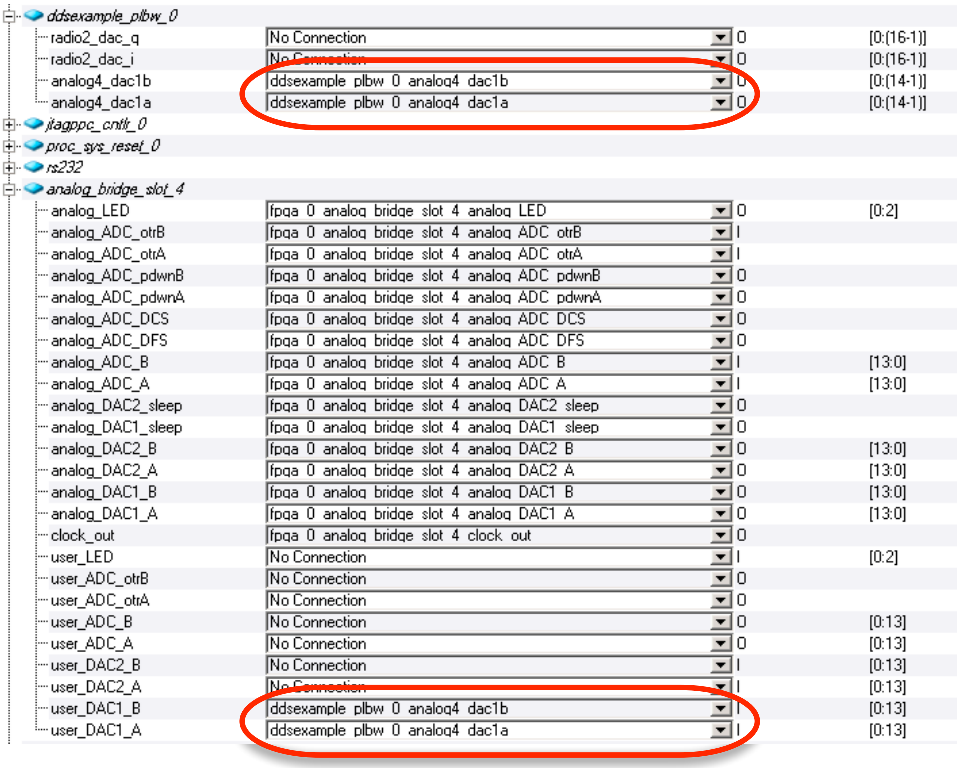
http://warp.rice.edu/svn/WARP/Documentation/Tutorials/Custom_Peripherals/html/Custom_Peripherals.html

The purpose of this document is to step through the integration of this peripheral with WARP platform support packages in Xilinx Platform Studio. When complete, the design will output a sine and cosine via the WARP analog board. The frequency of these sinusoids will be configurable in real time via WARP's serial port.

2 Integrating the Transmitter Model

We have provided a pre-built project in Xilinx Platform Studio for this exercise. You can download this project here: http://cmclab.rice.edu/433/demos/warpDDSExample.zip. **Note: Make sure to unzip this folder in a path that contains no spaces**. This project contains some relevant pieces of the WARP Platform Support Packages along with a pre-made software project to exercise the model you built using the tutorial.

1. Copy the `ddsexample_plbw_v1_00_a` pcore from the tutorial netlist folder into the `\pcores\` folder of the XPS project.
2. Open Xilinx Platform Studio. When prompted, select **Open a recent project**, click OK, then navigate to the `system.xmp` file.
3. Click the **IP Catalog** tab on the left of the screen, and expand the **Project Local pcores / USER** category.
4. Double-click the **ddsexample_plbw** core and click 'Yes'.
5. Click the **Bus Interface** button in the middle of the screen and look for the **ddsexample_plbw** core in the list of included peripherals. Expand the core's entry and click the hollow yellow circle to attach the core to the PLB.
6. Switch to the **Ports** view and scroll down to the **ddsexample_plbw** entry. Expand the entry. You'll see two ports that are relevant to this exercise: **analog4_dac1a** and **analog4_dac1b**. These ports represent the two DAC blocks in the System Generator model. For each port, click its drop-down list and choose New Connection. This will create a new net for each port. Leave the other ports unconnected.
7. Scroll to the **analog_bridge_slot_4** core and expand its entry. Scroll down the list of ports to **user_DAC1_A** and **user_DAC1_B**. In the drop-down list for each, select the corresponding new net created in the previous step. By choosing these nets, you're connecting the outputs of the transmitter model to the FPGA pins mapped to the analog board's D/A converters.
8. When complete, your ports list should look like those shown in Figure 1.
9. Switch to the **Addresses** view and click **Generate Addresses**. When this process finishes, the **ddsexample_plbw** core will have an automatically assigned base address.
10. Choose Hardware → Generate Bitstream to begin the hardware implementation process.This step will take 10-15 minutes. While it's running, you can look through the software project (described in the next section).



ddsexample_plbw_0			
radio2_dac_q	No Connection	O	[0:(16-1)]
radio2_dac_i	No Connection	O	[0:(16-1)]
analog4_dac1b	ddsexample plbw 0 analog4 dac1b	O	[0:(14-1)]
analog4_dac1a	ddsexample plbw 0 analog4 dac1a	O	[0:(14-1)]
jtagppc_cntlr_0			
proc_sys_reset_0			
rs232			
analog_bridge_slot_4			
analog_LED	fpga 0 analog bridge slot 4 analog LED	O	[0:2]
analog_ADC_otrB	fpga 0 analog bridge slot 4 analog ADC otrB	I	
analog_ADC_otrA	fpga 0 analog bridge slot 4 analog ADC otrA	I	
analog_ADC_pdwnB	fpga 0 analog bridge slot 4 analog ADC pdwnB	O	
analog_ADC_pdwnA	fpga 0 analog bridge slot 4 analog ADC pdwnA	O	
analog_ADC_DCS	fpga 0 analog bridge slot 4 analog ADC DCS	O	
analog_ADC_DFS	fpga 0 analog bridge slot 4 analog ADC DFS	O	
analog_ADC_B	fpga 0 analog bridge slot 4 analog ADC B	I	[13:0]
analog_ADC_A	fpga 0 analog bridge slot 4 analog ADC A	I	[13:0]
analog_DAC2_sleep	fpga 0 analog bridge slot 4 analog DAC2 sleep	O	
analog_DAC1_sleep	fpga 0 analog bridge slot 4 analog DAC1 sleep	O	
analog_DAC2_B	fpga 0 analog bridge slot 4 analog DAC2 B	O	[13:0]
analog_DAC2_A	fpga 0 analog bridge slot 4 analog DAC2 A	O	[13:0]
analog_DAC1_B	fpga 0 analog bridge slot 4 analog DAC1 B	O	[13:0]
analog_DAC1_A	fpga 0 analog bridge slot 4 analog DAC1 A	O	[13:0]
clock_out	fpga 0 analog bridge slot 4 clock out	O	
user_LED	No Connection	I	[0:2]
user_ADC_otrB	No Connection	O	
user_ADC_otrA	No Connection	O	
user_ADC_B	No Connection	O	[0:13]
user_ADC_A	No Connection	O	[0:13]
user_DAC2_B	No Connection	I	[0:13]
user_DAC2_A	No Connection	I	[0:13]
user_DAC1_B	ddsexample plbw 0 analog4 dac1b	I	[0:13]
user_DAC1_A	ddsexample plbw 0 analog4 dac1a	I	[0:13]

Figure 1: Custom core and analog bridge ports

3 Driving the System from Software

1. We have provided a software project which writes a suitable value to the "phaseInc" register of the **ddsexample_plbw** core. Additionally, it configures the serial port to allow user input to change the frequency of the sinusoid in real time.

2. Switch to the **Applications** tab on the left of the screen. Expand the Sources list and double-click the one file entry.

3. Look through the C code to understand how the serial port is used. Scroll down in the code and look for the lines which read and write the "phaseInc" register in the model. These lines start with `XIo_In32` and `XIo_Out32` respectively. Notice the first argument in these function calls: `PHASEINC`. This is a shorthand for `XPAR_DDSEXAMPLE_PLBWP_0_MEMMAP_PHASEINC`, which itself represents the memory address of the register in the model. The tools creates this constant automatically. The second argument of the write function is the value that gets written to the register.

4 Testing the Design in Hardware

1. Make sure your WARP FPGA board is connected to power and USB.
2. Launch TeraTermPro and open the COM port associated with the serial connection. Set the baud rate to 57600 in Setup → Serial Port...
3. In XPS, Choose Device Configuration → Download Bitstream. This will compile any code changes and update the FPGA programming file. It will then download that programming file to the board.
4. Press the "Autoset" button on the oscilloscope to get the settings in the ballpark of what we want. The result should be something like Figure 2.

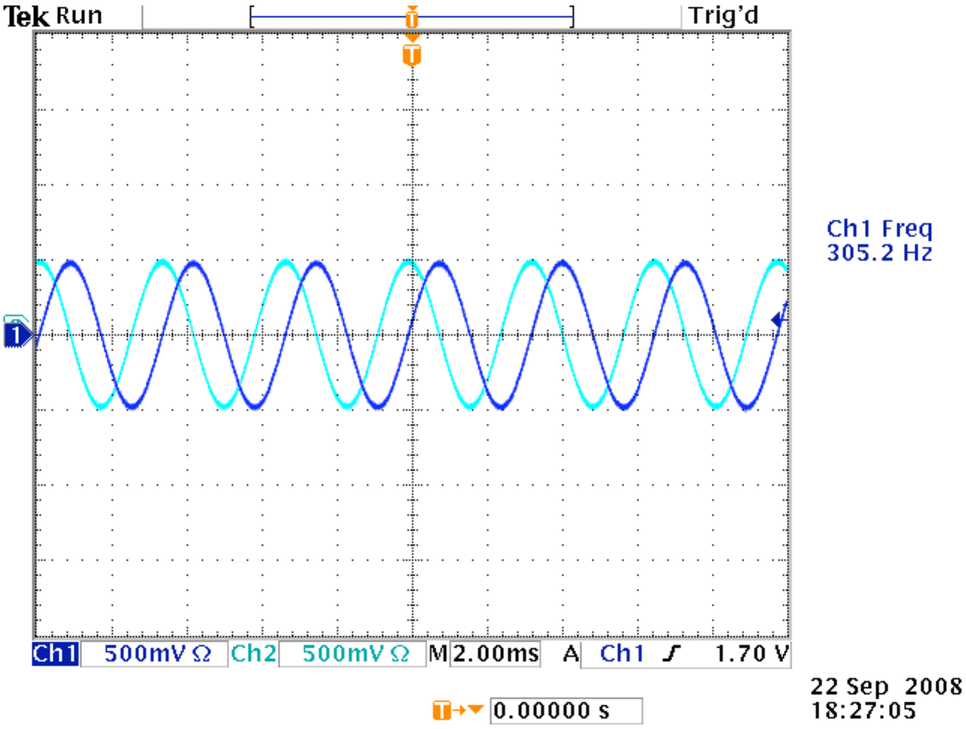


Figure 2: Oscilloscope Output

5. If everything works, the terminal will display "******WARP DDS EXAMPLE*******." At this point, pressing the "1" key on the keyboard will increase the phase increment by a factor of two. Pressing the "q" key will decrease the phase increment by a factor of two.