XESS Corporation

Stereo Codec Interface

March 29, 2004 (Version 1.0)

Application Note by D. Vanden Bout

Summary

This application note describes a simple interface core that allows an FPGA to access the stereo codec on the XST and XSB-300E Boards.

Codec Interface Features

The codec interface core transforms the serial interface of the stereo codecs on the XST and XSB-300E Boards into a parallel interface for use by host-side logic in the FPGA. Serial data from the codec left- and right-channel analog-to-digital converters (ADCs) is gathered into a 20-bit shift-register and presented in parallel to the host side. Similarly, the host-side logic presents a 20-bit parallel value to the codec interface and this is serialized for transmission to the left- and right-channel digital-to-analog converters (DACs). The codec interface core generates all the timing and synchronization signals required on both the codec and host sides.

Codec Interface Parameters and I/O

Generic Parameters

Several generic parameters affect the operation of the codec interface:

XST_1_3_X: Setting this Boolean parameter to TRUE inserts output inverters and adjusts the timing of the interface to match the codec circuitry on the XST-1.3.X Board.

XST_2_0_X: Setting this Boolean parameter to TRUE adjusts the timing of the interface to match the codec circuitry on the XST-2.0.X Board.

XST_2_1_X: Setting this Boolean parameter to TRUE adjusts the timing of the interface to match the codec circuitry on the XST-2.1.X Board.

XSB_300E: Setting this Boolean parameter to TRUE adjusts the timing of the interface to match the codec circuitry on the XSB-300E Board.

DIVISOR: This parameter sets the ratio between the frequencies of the main FPGA clock and the codec master clock. The divisor should be a power of two. For example, if the FPGA receives a 50 MHz clock, then the divisor should be set to four so the codec gets a 12.5 MHz clock.

RESOLUTION: This parameter sets the number of bits of data received from the ADC and sent to the DAC through the host-side interface. This parameter affects the sizes of the shift registers used in the interface, however all serial transmissions to and from the codec are still twenty bits long.

I/O Ports

The host- and codec-side connections for the interface are shown in Figure 1. The functions of the I/O signals are as follows:

clk: This is the main clock input. The clock from the external oscillator enters the FPGA through a global clock input pin and drives this input.

rst_n: This active-low, asynchronous input resets the internal circuitry of the interface.

left_chan: When high, this output tells the host that the output on the ADC bus is from the left channel of the codec and values input through the DAC bus will also go to the left channel. When this output is low, then read and write operations will go to the right channel.

ADC: The output on this parallel bus is the value read serially from the left- or right-channel ADC as indicated by a high or low output on left_chan, respectively.

ADC_rdy: The value on the ADC bus is valid when this output is high. The value on the ADC bus should be stored in a register on the first rising clock edge after ADC_rdy goes high.

DAC: The value input through this parallel bus is loaded into a shift register and transmitted over to the left- or right-channel DAC as indicated by a high or low output on left_chan, respectively.

DAC_rdy: The value on the DAC bus is loaded into the shift register on the first rising clock edge after this output goes high.

mclk: This output drives the master clock input of the codec. The mclk frequency is the clk frequency divided by the DIVISOR parameter. To handle audio signals in the 0–20 KHz range, mclk should be around 12 MHz.

sclk: This is the serial bit clock that synchronizes the transfer of bits in and out of the codec. The sclk frequency is 1/4 the mclk frequency.

lrck: When high, this output activates the left-channel of the codec and the right-channel is active when low. The lrck frequency is 1/64 the sclk frequency. This establishes a 32-bit frame for each channel.

sdti: This output drives the SDTI input of the codec with the 20-bit serial transmission from the DAC shift register in the codec interface. For the AK4520 codec used in the XST-1.3.X and XST-2.0.X Boards or the AK4565 codec used in the XSB-300E Board, the 20-bit value is transmitted MSB-first within the first twenty bits of each 32-bit frame. For the AK4551 used in the XST-2.1.X Board, the 20-bit value is transmitted MSB-first within the last twenty bits of each 32-bit frame.

sdto: This input receives the 20-bit serial transmission from the left or right ADC in the codec. The 20-bit value is received MSB-first within the first twenty bits of each 32-bit frame.

Using the Codec Interface

The timing waveforms for the codec interface when used with an XST-1.3.X, XST-2.0.X or XSB-300E Board are shown in Figure 2. At ①, DAC_rdy goes high to signal that data for the left-channel DAC is needed (as indicated by left_chan being high). The data must be presented on the DAC bus before the next rising edge on clk (not sclk). The value is shifted out MSB-first on sdti during interval ②. During the same interval, the data from the left-channel ADC is received through sdto. ADC_rdy goes high at ③ to indicate all the data from the ADC is available. The host must store the ADC data on the first rising edge of clk after ADC_rdy goes high. The same operations are performed for the right channel at points ④, ⑤ and ⑥.

The timing waveforms for the codec interface when used with an XST-2.1.X Board are shown in Figure 3. These are the similar to the previous example except the data is transferred to the DAC in the last twenty bits of each 32-bit frame.

Codec Loop-Back Test Design

A simple loop-back design demonstrates the use of the codec interface. The loop-back design just accepts data from the left and right ADCs and sends them back to the left and right DACs, respectively. This takes any audio input to the codec and replicates it on its audio outputs.

The source files that describe this design are:

common.vhd: Some functions and definitions useful in many applications are provided in this file.

codec.vhd: This file contains the descriptions of the codec interface and the top-level loop-back example.

xsa.ucf, xsb.ucf: These files contain the loop-back design pin assignments for the XSA+XST Board combination or the XSB-300E Board, respectively.

codectst50.npl, codectst100.npl, codectst300e.npl: These files tell WebPACK 6.1 how to combine the source files to create the loop-back bitstream for the XSA-50, XSA-100 and XSB-300E Boards, respectively.

Testing the Loop-Back Design on the XST-1.3.X Board

1. Create a bitstream for the loop-back design targeted at the XSA-50 or XSA-100 Board.

2. Insert the XSA-50 or XSA-100 Board into the XST-1.3.X Board.

3. On the XST Board, place DIP switches SW1-3,4,5,6,7 (or SW1-C,D,E,F,G) in the OPEN (OFF) position.

4. Remove the shunt from jumper J11 on the XST Board.

5. Download the FPGA bitstream using GXSLOAD.

6. Connect a stereo audio source (e.g. the headphone output of a CD player or the speaker output of a sound card will do) to input jack J9 on the XST Board.

7. Connect headphones to output jack J10 on the XST Board.

8. You should hear the audio playing through the headphones.

Testing the Loop-Back Design on the XST-2.0.X and XST-2.1.X Boards

1. Create a bitstream for the loop-back design targeted at the XSA-50 or XSA-100 Board.

2. Insert the XSA-50 or XSA-100 Board into the XST-2.0.X or XST-2.1.X Board.

3. On the XST Board, place DIP switches SW1-3,4,6 (or SW1-C,D,F) in the OPEN (OFF) position.

4. Remove the shunts from jumper JP1, JP2 and JP3 on the XST Board. Place shunts on JP4 and JP5.

5. Download the FPGA bitstream using GXSLOAD.

6. Connect a stereo audio source (e.g. the headphone output of a CD player or the speaker output of a sound card will do) to input jack J1 on the XST Board.

7. Connect headphones to output jack J2 on the XST Board.

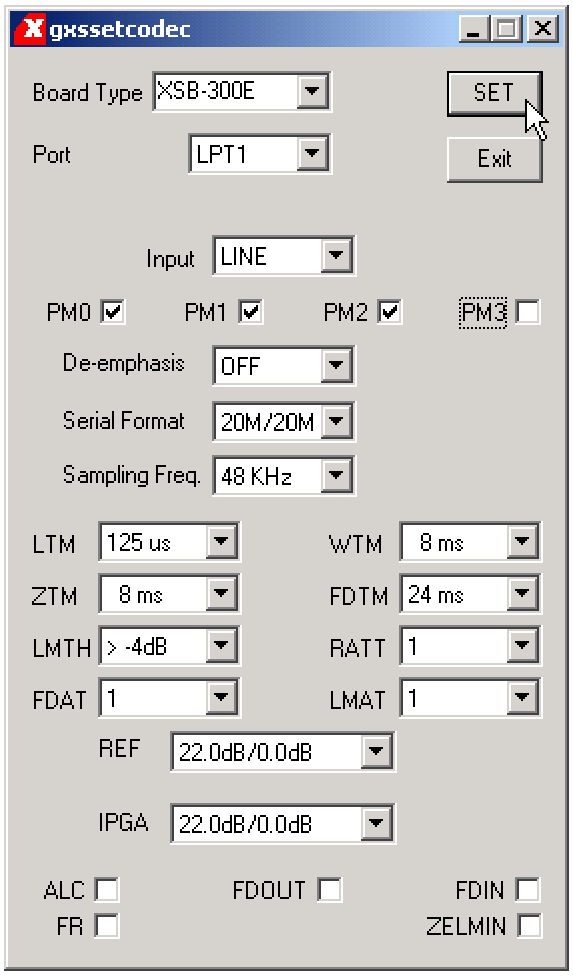
8. You should hear the audio playing through the headphones.

Testing the Loop-Back Application on the XSB-300E Board

1. Create a bitstream for the loop-back design targeted at the XSB-300E Board.

2. Place a shunt on the INT position of jumper JP8.

3. Double-click the gxssetcodec icon in the XSTOOLS folder. Then click on the SET button in the window that appears. This will load the codec chip with a default configuration. Then click on the Exit button.



Board Type XSB-300E
Port LPT1
Input LINE
PM0 ✓ PM1 ✓ PM2 ✓ PM3 ☐
De-emphasis OFF
Serial Format 20M/20M
Sampling Freq. 48 KHz
LTM 125 us
WTM 8 ms
ZTM 8 ms
FDTM 24 ms
LMTH > -4dB
RATT 1
FDAT 1
LMAT 1
REF 22.0dB/0.0dB
IPGA 22.0dB/0.0dB
ALC ☐ FDOUT ☐ FDIN ☐
FR ☐ ZELMIN ☐

4. Download the FPGA bitstream using GXSLOAD.

5. Connect a stereo audio source (e.g. the headphone output of a CD player or the speaker output of a sound card will do) to the blue input jack on J1.

6. Connect headphones to the green output jack on J1.

7. You should hear the audio playing through the headphones.

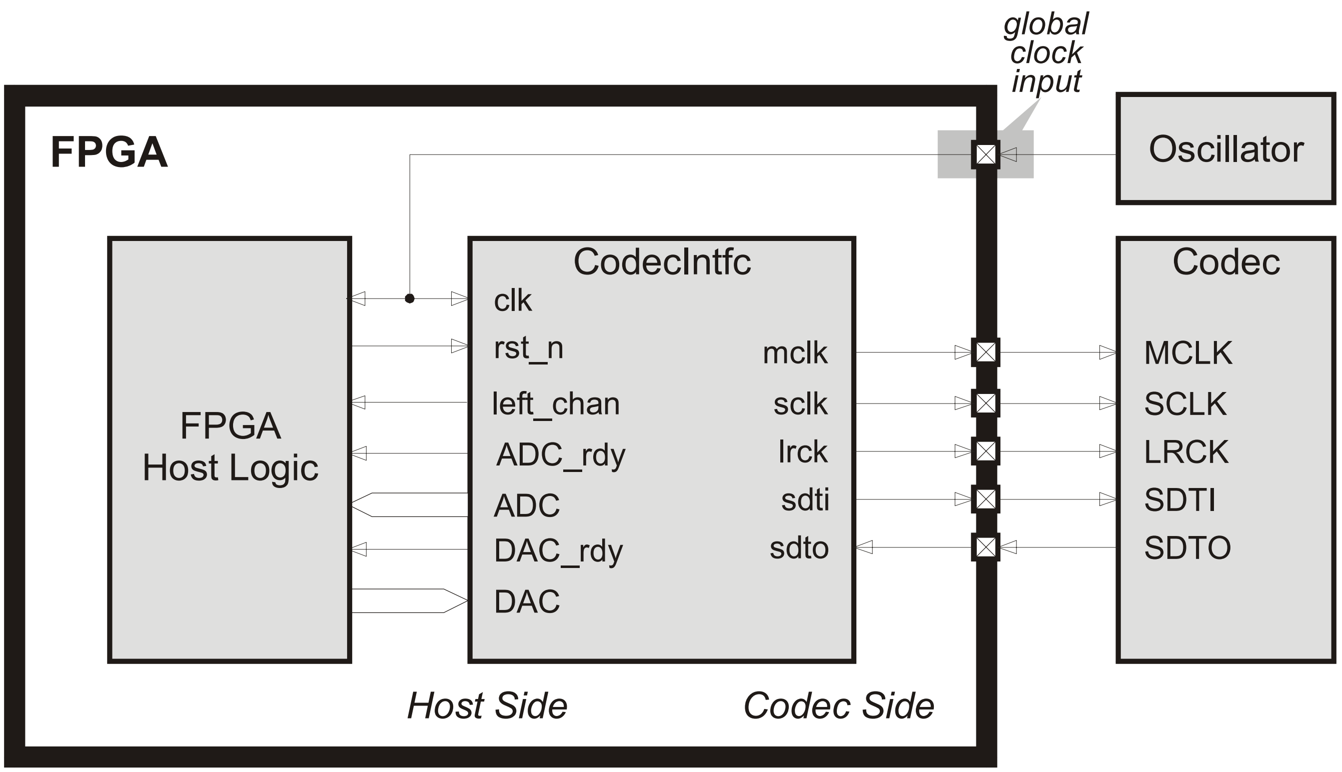


Figure 1: Codec interface connections.

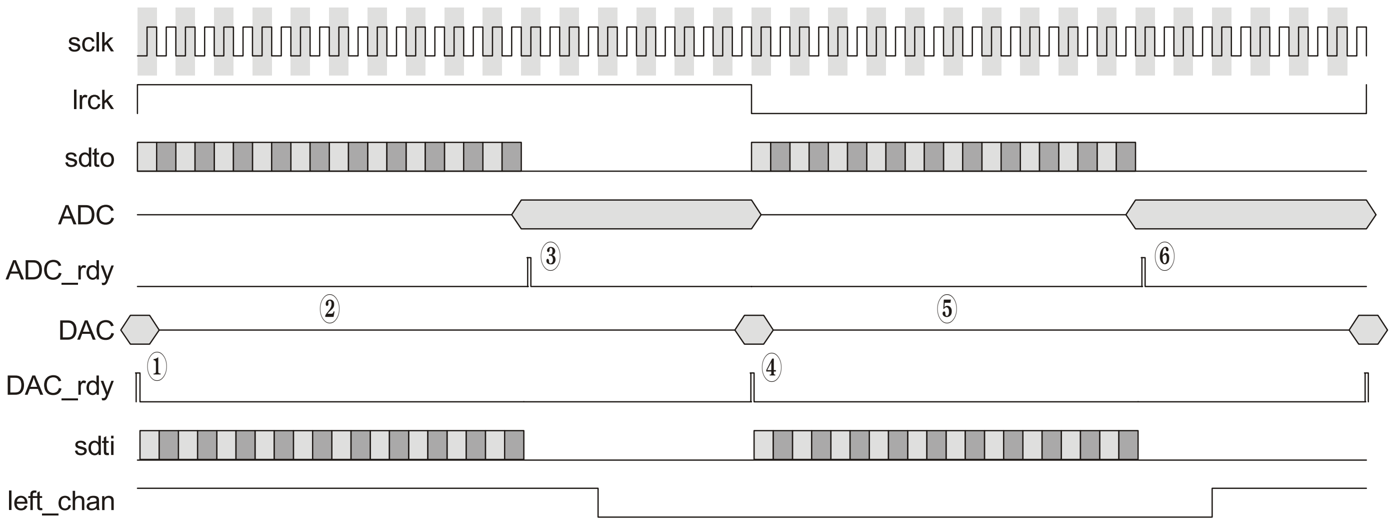


Figure 2: Codec interface waveforms for the XST-1.3.X, XST-2.0.X and XSB-300E Boards.

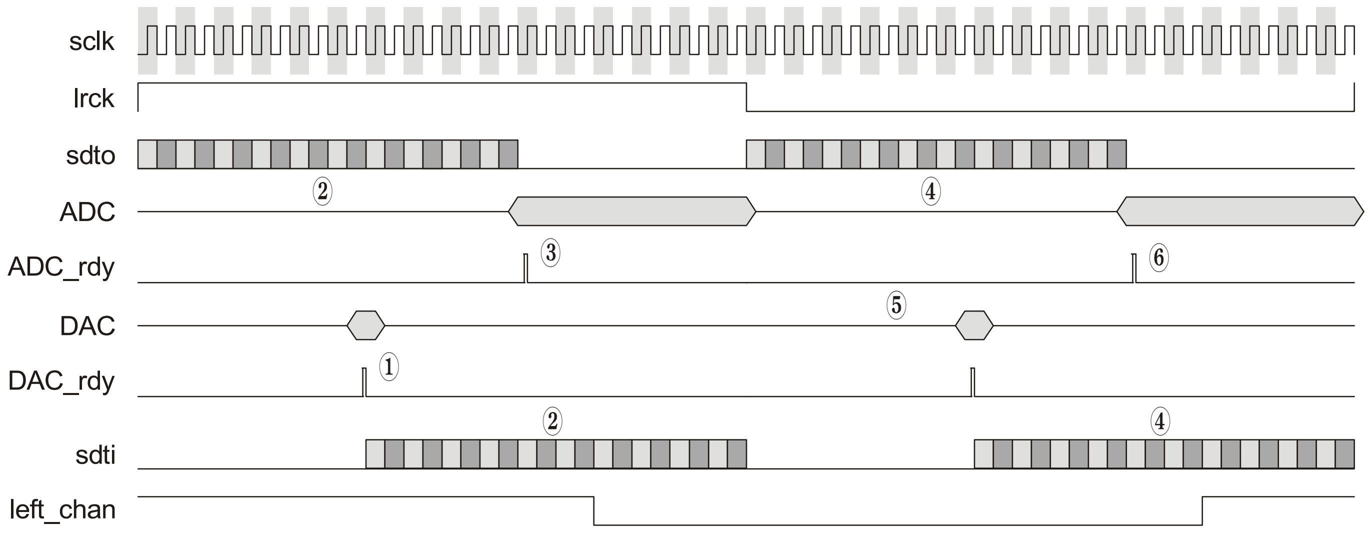


Figure 3: Codec interface waveforms for the XST-2.1.X Board.