Configuring the Audio Codec on the XSB-300E Board

November 15, 2003 (Version 1.0) Application Note by D. Vanden Bout

Summary

This application note describes the circuitry and software that are used to configure the AK4565 audio codec on the XSB-300E Board.

Introduction

The XSB-300E Board processes stereo audio signals with its AK4565 codec chip. The AK4565 has many programmable options that are controlled by setting values into the registers within the chip. The registers are loaded through a serial interface.

It can be daunting to design the interface circuit to the registers and the state machine that loads the appropriate register values in addition to designing the signal processing circuitry that handles the data streaming to and from the codec. Therefore, XESS provides a simple PC-based utility, GXSSETCODEC, that allows you to initialize the codec registers. Once initialized, the registers will retain their values until power to the XSB-300E Board is interrupted. After the codec is initialized, you can download your FPGA bitstream file that configures the FPGA for audio data stream processing.

GXSSETCODEC employs a two-phase operation:

Phase 1: The FPGA is configured with a bitstream that creates a path from the the PC parallel port to the serial register interface pins of the audio codec.

Phase 2: A GUI is used to select the desired register values that are then downloaded into the codec through the parallel port.

Codec Parallel Port Interface

A high-level schematic for the parallel port interface to the audio codec is shown in Figure 1.

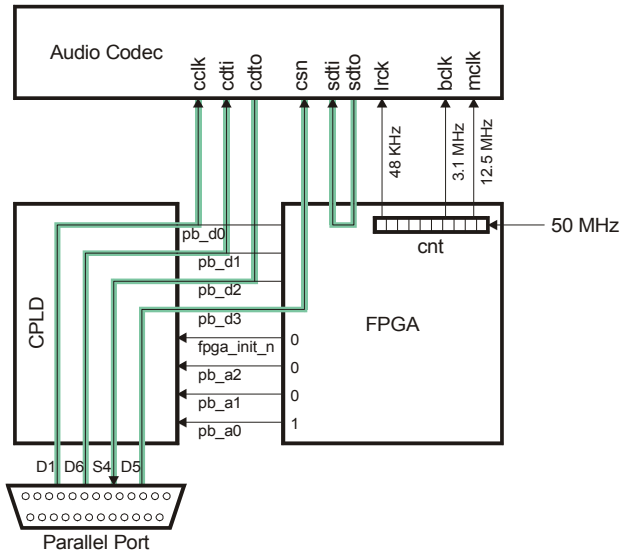


Figure 1: Codec parallel port interface schematic.

Listing 1 shows the VHDL code for the default parallel port interface that is programmed into the XC9572XL CPLD on the XSB-300E Board. Lines 68–72 are responsible for the parallel port interface to the audio codec. This circuitry is activated after the FPGA is configured and the FPGA lowers the the fpga_init_n line and places the bitstring “001” onto the lower three bits of the peripheral bus (pb_a(2:0)) as described by lines 27–28 of Listing 3. The chip-select for the codec is connected to bit D5 of the parallel port through the CPLD and FPGA. Data bit D1 of the parallel port drives the clock pin of the codec register interface while data bit D6 connects to the serial data input pin. Serial data output from the codec arrives back at the PC through the S4 status pin of the parallel port.

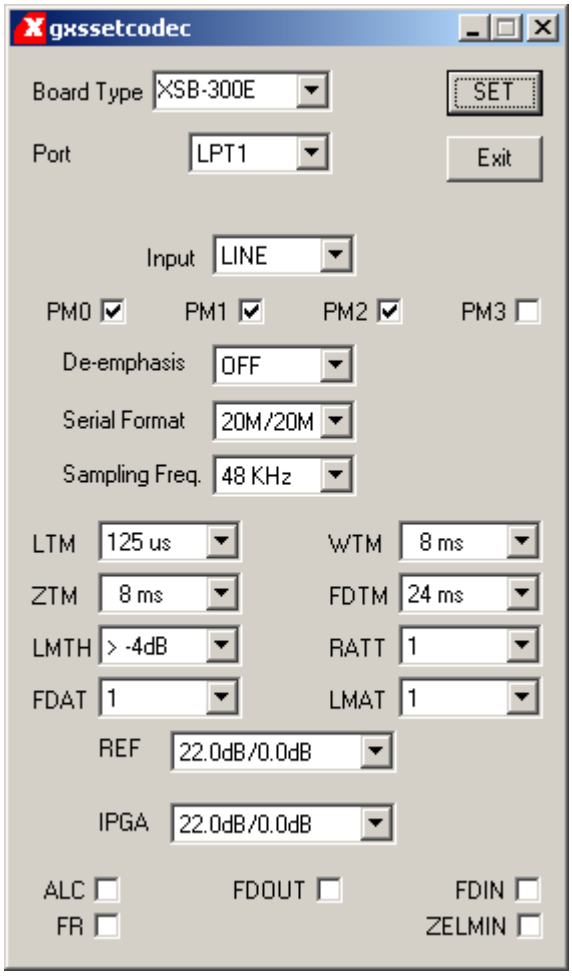
The FPGA also implements a simple loopback circuit that allows you to pass an audio signal through the

codec and listen to it to gauge the effect of changing the configuration registers. This loopback is accomplished by piping the digitized output from the codec back into its input. The necessary codec clock signals are generated from a ten-bit counter that is clocked by the 50 MHz output of the programmable oscillator chip on the XSB-300E Board.

GXSSETCODEC Graphical User Interface

The GXSSETCODEC utility is started by clicking on the 

icon. Then the **gxssetcodec** window will appear as shown below. Set the Board Type field to XSB-300E since this is the only board from XESS that uses the AK4565 codec and GXSSETCODEC will not work with any of the other XESS boards. Then set the Port field to LPT1, LPT2 or LPT3 depending upon what parallel port the XSB-300E is connected to. Finally, click on the SET button to load the FPGA with the bitstream for configuring the codec through the parallel port.



The codec registers are also initialized at this time with the default values shown in the window. Any further changes to the codec option settings will be immediately loaded into the codec registers. The effect of the settings for the various codec options will now be discussed.

The Input field selects one of four audio inputs to the codec. Select EXT if the audio signal is coming from a microphone connected to the pink jack on J1 of the XSB-300E Board. Select LINE if the audio signal is arriving through the blue jack of J1. In general, INT0 or INT1 should not be used unless you have a source of audio wired to the appropriate pins of the JP1 header.

The PM0, PM1, PM2 checkboxes control the power that goes to the input amplifier, ADC and DAC blocks of the AK4565, respectively. In general, these checkboxes should all be checked so these blocks receive power. The PM3 checkbox activates an analog loopback mode in the codec and should usually be left unchecked.

The De-emphasis field selects whether a filter is inserted into the audio input signal path to attenuate higher frequencies. This field should usually be set to OFF so that the codec operates with a flat input frequency response.

The Serial Format field selects the bit ordering and timing for the serial streams that come from and go to the codec ADC and DAC, respectively. The 20M/16L setting programs the codec to output 20 bits from the ADC starting with the most-significant bit in the first time slot of a 32-slot frame while 16 bits are sent to the DAC with the least-significant bit occupying the last slot of the frame. The 20M/20L is similar except that the DAC receives 20 bits instead of 16. Finally, the 20M/20M setting programs the codec to output 20 bits from the ADC starting with the most-significant bit in the first time slot of a 32-slot frame while 20 bits are sent to the DAC with the most-significant bit also occupying the first slot of the frame. The 20M/20M setting must be used if you want to apply an audio source and listen to it using the loopback feature of GXSSETCODEC.

The Sampling Freq. Field selects whether the codec operates at 32 or 48 Ksamples/second. This field should be set to 48 KHz if you are using the loopback feature of GXSSETCODEC.

The LTM, WTM, ZTM, FDTM, LMTH, RATT, FDAT, LMAT and REF fields all exert control over the automatic level control circuitry of the programmable gain amplifier on the input to the ADC. These settings are not critical to the basic operation of the codec. You should read the datasheet for the AK4565 to determine the exact effects of these settings.

The IPGA field selects the gain for the amplifier on the input to the ADC in the codec. This allows you to amplify low-level signals to make the best use of the signal range afforded by the ADC. The gain is automatically increased by an additional 22 dB if the Input field is set to EXT since this usually means a low-level signal from a microphone is the source of the audio input.

When checked, the ALC checkbox activates the automatic level control circuitry.

When checked, the FDIN and FDOUT checkboxes enable the circuitry that gradually increases and decreases the output from the DAC.

When checked, the FR checkbox allows the automatic level control to respond to impulse noise.

When not checked, the ZELMIN checkbox allows the automatic level control circuitry to change the input gain only on zero-crossings of the input signal.

Listing 1: VHDL code for the default CPLD parallel port interface.

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use IEEE.std_logic_arith.all;
4
5  entity dwnldpar is
6     port(
7        -- parallel port data and status pins
8        pp_d:            in     std_logic_vector(7 downto 0);
9        pp_s:            out    std_logic_vector(5 downto 3);
10
11       -- FPGA configuration pins
12       fpga_m:          out   std_logic_vector(0 downto 0);  -- config. mode select (out)
13       fpga_program_n:  out    std_logic;    -- active-low config. initiate (out)
14       fpga_cclk:       out    std_logic;    -- config. clock (out)
15       fpga_cs_n:       out    std_logic;    -- active-low chip-select (out)
16       fpga_write_n:    out    std_logic;    -- active-low write-enable (out)
17       fpga_init_n:     inout  std_logic;    -- config. initialization (in)
18       fpga_done:       in     std_logic;    -- config. done (in)
19       fpga_tck:        out    std_logic;    -- JTAG clock (out)
20
21       -- peripheral bus
22       pb_d:            inout  std_logic_vector(7 downto 0); -- config. data (out)
23       pb_a:            inout  std_logic_vector(19 downto 0) -- address bus (in)
24
25    );
26 end entity dwnldpar;
27
28
29 architecture arch of dwnldpar is
30    constant LO: std_logic := '0';
31    constant HI: std_logic := '1';
32    constant HIZ: std_logic := 'Z';
33    constant SLAVE_PARALLEL_MODE: std_logic_vector(0 downto 0) := "0";
34    signal cclk: std_logic;
35    signal config_data, nybble: std_logic_vector(3 downto 0);
36    signal cpld_addr: std_logic_vector(3 downto 0);
37
38    begin
39
40    nybble    <= pp_d(5 downto 2);          -- data from PC to board
41    cpld_addr <= fpga_init_n & pb_a(2 downto 0);   -- selects CPLD profile
42
43    process(pp_d,pb_a,nybble,config_data,cclk,fpga_done,cpld_addr)
44    begin
45       fpga_cs_n       <= HIZ;
46       fpga_write_n    <= HIZ;
47       pb_a            <= (others=>HIZ);
48       pb_d            <= (others=>HIZ);
49       fpga_tck        <= LO;              -- deactivate FPGA JTAG circuit
50       fpga_m          <= SLAVE_PARALLEL_MODE;-- set FPGA config mode
51       fpga_program_n  <= pp_d(7);         -- FPGA PROGRAM# comes from parallel port
52       cclk            <= not pp_d(0);     -- internal configuration clock
53       fpga_cclk       <= not cclk;        -- FPGA configuration clock
54
55       if fpga_done=LO then-- FPGA is not configured
56          fpga_cs_n    <= LO;              -- enable writing of config. data
57          fpga_write_n <= LO;
58          pb_d         <= config_data & nybble;   -- two nybbles of config data
59       else   -- FPGA is configured
60          fpga_cs_n    <= not pp_d(1);     -- clock for data interchange
61          fpga_write_n <= pp_d(6);         -- reset for data interchange FSM
62          case conv_integer(unsigned(cpld_addr)) is
```

```
63              when 0 => -- I2C programming is selected by the FPGA
64                 pb_d(0)   <= not pp_d(1);  -- I2C clock to SAA7114 & osc chips
65                 pb_d(1)   <= pp_d(6);      -- I2C data to SAA7114 & osc chips
66                 pp_s(3)   <= pb_a(3);      -- I2C clock from SAA7114 & osc chips
67                 pp_s(4)   <= pb_a(4);      -- I2C data from SAA7114 & osc chips
68              when 1 => -- stereo codec programming is selected by the FPGA
69                 pb_d(0)   <= not pp_d(1);  -- config. clock to codec chip
70                 pb_d(1)   <= pp_d(6);      -- config. data to codec chip
71                 pb_d(3)   <= pp_d(5);      -- chip-select to codec chip
72                 pp_s(4)   <= pb_d(2);      -- config. data from codec chip
73              when 2 => -- data interchange interface
74                 pb_d(5 downto 2) <= pp_d(5 downto 2); -- data nybble from PC to FPGA
75                 pp_s(5 downto 3) <= pb_a(5 downto 3); -- data bits from FPGA to PC
76              when 3 => -- GXSPORT/XSPORT interface
77                 pb_d(6 downto 0) <= pp_d(6 downto 0);
78                 pp_s(5 downto 3) <= pb_a(5 downto 3);
79              when 4 | 5 | 6 | 7 =>  -- currently undefine CPLD modes
80              when others => -- CPLD is not selected by the FPGA
81                 pp_s(5)      <= pb_a(19); -- for reporting GXSTEST status
82           end case;
83        end if;
84
85     end process;
86
87     -- gather 4-bit data from parallel port
88     process(cclk)
89     begin
90        if rising_edge(cclk) then
91           config_data <= nybble;
92        end if;
93     end process;
94
95  end architecture arch;
```

Listing 2: User-constraint file for the CPLD pin assignments.

```
#
# pin assignments for the XC9572XL CPLD chip on the XSB Board

# peripheral bus
net pb_d<0>      loc=p2;   # data bit D0 (in/out)
net pb_d<1>      loc=p4;   # data bit D1 (in/out)
net pb_d<2>      loc=p5;   # data bit D2 (in/out)
net pb_d<3>      loc=p6;   # data bit D3 (in/out)
net pb_d<4>      loc=p7;   # data bit D4 (in/out)
net pb_d<5>      loc=p8;   # data bit D5 (in/out)
net pb_d<6>      loc=p9;   # data bit D6 (in/out)
net pb_d<7>      loc=p10;  # data bit D7 (in/out)
net pb_a<0>      loc=p1;   # address bit A0 (in/out)
net pb_a<1>      loc=p64;  # address bit A1 (in/out)
net pb_a<2>      loc=p63;  # address bit A2 (in/out)
net pb_a<3>      loc=p62;  # address bit A3 (in/out)
net pb_a<4>      loc=p61;  # address bit A4 (in/out)
net pb_a<5>      loc=p60;  # address bit A5 (in/out)
net pb_a<19>     loc=p50;  # address bit A19 (in/out)

# parallel port
net pp_d<0>      loc=p33;  # data pin D0 (in)
net pp_d<1>      loc=p32;  # data pin D1 (in)
net pp_d<2>      loc=p31;  # data pin D2 (in)
net pp_d<3>      loc=p27;  # data pin D3 (in)
net pp_d<4>      loc=p25;  # data pin D4 (in)
net pp_d<5>      loc=p24;  # data pin D5 (in)
net pp_d<6>      loc=p23;  # data pin D6 (in)
net pp_d<7>      loc=p22;  # data pin D7 (in)
net pp_s<3>      loc=p34;  # status pin S3 (out)
net pp_s<4>      loc=p20;  # status pin S4 (out)
net pp_s<5>      loc=p35;  # status pin S5 (out)

# FPGA configuration interface
net fpga_m<0>    loc=p36;  # config. mode select (out)
net fpga_program_n  loc=p39;  # active-low config. initiate (out)
net fpga_cclk    loc=p16;  # config. clock (out)
net fpga_cs_n    loc=p15;  # active-low chip-select (out)
net fpga_write_n loc=p19;  # active-low write-enable (out)
net fpga_init_n  loc=p38;  # config. initialization (in)
net fpga_done    loc=p40;  # config. done (in)
net fpga_tck     loc=p13;  # JTAG clock (out)
```

Listing 3: VHDL code for the FPGA audio codec interface.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity cfgcodec is
    port
    (
        fpga_init_n: out std_logic;                 -- INIT# pin
        fpga_clk: in std_logic_vector(1 downto 0);
        pb_d: in std_logic_vector(15 downto 0);     -- to parallel port data pin
        pb_a: out std_logic_vector(19 downto 0);    -- to parallel port status pin
        au_csn_n: out std_logic;                    -- audio codec chip-enable
        au_bclk: out std_logic;
        au_mclk: out std_logic;
        au_lrck: out std_logic;
        au_sdti: out std_logic;
        au_sdto: in std_logic
    );
end cfgcodec;


architecture arch of cfgcodec is
    signal cnt: std_logic_vector(9 downto 0);
begin

    fpga_init_n <= '0';     -- indicate that the codec is available for programming
    pb_a(2 downto 0) <= "001";

    -- parallel port codec configuration interface
    au_csn_n <= pb_d(3);

    process(fpga_clk(0))
    begin
        if(fpga_clk(0)'event and fpga_clk(0)='1') then
            cnt <= cnt+1;
        end if;
    end process;

    au_lrck <= cnt(9);
    au_bclk <= cnt(3);
    au_mclk <= cnt(1);
    au_sdti <= au_sdto;

end arch;
```

Listing 4: User-constraint file for the FPGA pin assignments.

```
#
# pin assignments for the XC2S300E FPGA chip on the XSB Board

net pb_a<0>         loc=p83;
net pb_a<1>         loc=p84;
net pb_a<2>         loc=p86;
net fpga_init_n     loc=p107;
net pb_d<3>         loc=p135;
net au_csn_n        loc=p165;
net au_bclk         loc=p166;
net au_mclk         loc=p167;
net au_lrck         loc=p168;
net au_sdti         loc=p169;
net au_sdto         loc=p173;
net fpga_clk<0>     loc=p182;
```