Audio Driver Documentation

For those of you interested in working with sound in your final projects, this is the sound driver that allows you to interact with the WM8731 Wolfson Audio Codec included onboard the Altera DE2 development board. I've included the file audio\_interface.vhd which handles the communication between some other entity and the audio codec. If you look at the port declarations on the audio\_interface.vhd file, you'll notice a large set of signals. The signals AUD\_MCLK, AUD\_BCLK, AUD\_ADCDAT, AUD\_DACDAT, AUD\_DACLRCK, AUD\_ADCLRCK, I2C\_SDAT, and I2C\_SCLK are signals which are connected to the WM8731. Please check the Altera DE2 pinout file to see the pins that these signals correspond to in the pinout.

The WM8731 Wolfson Audio Codec is a device that is configured using the I2C protocol, and is then fed data with a DSP protocol. I2C is a protocol which works by reading or writing a number of virtual registers on a given device. In order to change settings on the WM8731, a number of registers are given values. I describe the configuration settings I use in abbreviated detail in the comments for SCI\_REG\_ROM and SCI\_DAT\_ROM. I advise great caution if you feel the need to change configuration settings, because many of the settings on the WM8731 are poorly described, and have unintended effects on your audio. However if you really do wish to change the configuration procedure, simply change the word\_limit constant which determines the length of the SCI\_REG\_ROM and SCI\_DAT\_ROM if you wish to add new configuration commands. The SCI\_REG\_ROM contains the actual virtual registers being written to, and the SCI\_DAT\_ROM contains the data being written into the corresponding virtual register. One fairly safe value to play with is the Headphone Volume value being stored into register R2. Play around with this value to change the volume.

The actual operation of the WM8731 consists of an initialization phase and then a data phase. During the initialization phase, all the initialization data specified in SCI\_REG\_ROM and SCI\_DAT\_ROM is fed to the WM8731. After this, the ADC and the DAC begin operation, the ADC reading in analog samples and converting them into digital values, and the DAC reading out digital sample values.

To actually interact with the sound driver, you'll need to use 9 signals: Clk, Reset, LDATA, RDATA, INIT, INIT\_FINISH, adc\_full, data\_over, and ADCDATA. Clk and Reset are the master clock and Reset signals respectively. LDATA is a 16 bit two's complement sample value that you are feeding to the left audio channel. RDATA is a 16 bit two's complement sample value that you are feeding to the right audio channel. To use the driver, you must first raise INIT high, at which point the sound driver will begin the initialization process. The sound driver will raise INIT\_FINISH when it has finished the initialization process. After this, data will be continuously read and written out. The adc\_full signal will be raised when one full 32 bit sample (a left-channel 16 bit two's complement sample concatenated with a right-channel two's complement sample) has been read, and the valid output will be found in ADCDATA. Likewise, to feed data to the DAC, left channel data must be fed (16-bit two's complement) into LDATA, and right channel data must be fed (16-bit two's complement) into RDATA. When a single sample has been fed correctly into the DAC, data\_over is raised high, and when the next sample begins being read out, data\_over is again lowered.

My suggestion in using the driver is to create a state machine that raises the INIT signal, and waits for INIT\_FINISH to be raised. Once INIT\_FINISH is raised, regular ADC or DAC operation can begin.