Music Player

Signal	**Direction**	**Description**
clk	input	Clock signal
reset	input	Reset signal
play_button	input	A one-cycle pulse indicating the play_button has been pressed
next_button	input	A one-cycle pulse indicating the next_button has been pressed
new_frame	input	The raw new_frame signal from the ac97_if codec
new_sample_generated	output	This output must go high for one cycle when a new sample is generated.
sample_out[17:0]	output	Our final output sample to the codec. This needs to be synced to new_frame

Master Control Unit

Signal	**Direction**	**Description**
clk	input	Clock signal
reset	input	Reset signal
play_button	input	A one-cycle pulse indicating the play_button has been pressed
next_button	input	A one-cycle pulse indicating the next_button has been pressed
song_done	input	From the song_reader indicating that the current song has finished.
play	output	True if the system should be playing, false if no audio output should be generated
reset_player	output	High when the player is moving on to the next song. Resets the other parts of the system so they aren't in the middle of their jobs.
song[1:0]	output	The song to play

Song Reader

Signal	**Direction**	**Description**
clk	input	Clock signal
reset	input	Reset signal
play	input	True if the song reader should be playing
song[1:0]	input	The song to play
note_done	input	From the note_player to indicate that the note has finished and that it is ready for the next note. It is created by ORing the note_done associated with each of the 3 note_player channels
note[5:0]	output	The note from the song_rom to play now.
duration[5:0]	output	The duration for the note from the song_rom to play now
new_note	output	One cycle pulse that tells the note_player to latch in the values on note and duration and start playing that note.
song_done	output	True if the song has finished
advance	output	From the beginning of a note's address - indicates whether or not time is passing yet and the notes should actually be played. True means there is passage of time.
parameters[2:0]	output	From the song rom, it passes along the three bits which indicate a certain condition

Note Player

Signal	**Direction**	**Description**
clk	input	Clock signal
reset	input	Reset signal
play_enable	input	True if the song reader should be playing
note_to load[5:0]	input	The note to load, passed from song_reader to one of the

		three note_player channels
duration[5:0]	input	The duration for the note from the song_rom to play now
load_new_note	input	One cycle pulse that tells the note_player to latch in the values on note and duration and start playing that note.
activate	input	From song_reader, indicates whether or not time is passing yet and therefore whether or not the note should be played yet
beat	input	Goes high for one cycle at 48Hz
generate_next_sample	input	From the codec_conditioner telling us to generate and output the next sample
note_done	output	Goes high when we have finished playing our note
sample_ready	output	Tells the codec_conditioner that we have a new sample ready for it.
sample_out[17:0]	output	The 18-bit audio sample output for our note
step_size[19:0]	output	From frequency rom, will be passed on to create_harmonic in order to generate a harmony

Create Harmonic

The purpose of this module is to return a sample that represents a harmony for the passed in sample_in. Weight, which is edited by interactive instrument editing, will determine how many samples are used to generate a harmony.

Signal	**Direction**	**Description**
clk	input	Clock signal
reset	input	Reset signal
play_enable	input	True if the song reader should be playing
generate_next_sample	input	From the codec_conditioner telling us to generate and output the next sample.
weight[1:0]	input	The user can go into interactive instrument editing mode and change the weight of the harmonics, i.e. the number

		of samples that will be generated for the note. The options are none, one or two. None means that it will be the base note only.
sample_in[17:0]	input	Generated by note_player, this is the base note and will be used to generate the rest of the samples in order to create a harmony.
step_size[19:0]	input	Passed in from note_player, used to generate the appropriate number of samples in order to create the harmony
harmonic_out[17:0]	output	The 18-bit audio sample output for our note, it is a combination of all our generate samples to create a harmony for our note.
sample_ready	output	Tells the codec_conditioner that we have a new sample ready for it.

Harmonic Chord Player
The purpose of this module is to create three different channels in order to make it possible to play chords of up to 3 notes. The module will determine which channel is empty when it has a load_new_note is true and set its respective note_player's duration, note_to_load and load_new_note. It will then instantiate three different create_harmonic modules in order to create a harmonic for each channel and then return final_sample which is made up of all the samples combined together.

Signal	**Direction**	**Description**
clk	input	Clock signal
reset	input	Reset signal
play_enable	input	True if the song reader should be playing
note_to load[5:0]	input	The note to load, passed from song_reader to one of the three note_player channels
duration[5:0]	input	The duration for the note from the song_rom to play now
load_new_note	input	One cycle pulse that tells the note_player to latch in the values on note and duration and start playing that note.

activate	input	From song_reader, indicates whether or not time is passing yet and therefore whether or not the note should be played yet
beat	input	Goes high for one cycle at 48Hz
generate_next_sample	input	From the codec_conditioner telling us to generate and output the next sample
note_done	output	Goes high when we have finished playing our note. It is created by ORing the note_dones for each of our 3 note_player channels.
sample_ready	output	Tells the codec_conditioner that we have a new sample ready for it. It is created by ANDing all the sample_readys from our three create_harmonic channels.
final_sample[17:0]	output	The 18-bit audio sample output for our note(s). It adds up the samples for each note we currently have in our note_player channels and their harmonies.

Interactive Instrument Editing

The purpose of this module is to edit weight, the variable which indicates how many samples will be used in create_harmonic to generate a harmony. Weight can only range from 0 (no harmony, only the base note) to 2 (base note at ⅝, sample1 at ¼ and sample2 at ⅛).

Signal	**Direction**	**Description**
clk	input	Clock signal
reset	input	Reset signal
switch1	input	If true then weight can be edited. False means we are not in interactive instrument editing mode. Switch1 is whether or not the switch is high on the board.
up_button	input	If true then we want to increase weight by 1. If weight is 2 then it will stay at 2.
down_button	input	If true then we want to increase weight by 1. If weight is 0 then it will stay at 0.
weight	output	By default it is 0. Outputs how many samples will make

		up the harmony for a note in create_harmonic.

Sine Reader

Signal	**Direction**	**Description**
clk	input	Clock signal
reset	input	Reset signal
step_size[19:0]	input	The step by which we count each time. This is the frequency.
generate_next_sample	input	From the codec_conditioner telling us to generate and output the next sample. True when we should create a new sample.
sample_ready	output	Tells the codec_conditioner that we have a new sample ready for it. True if we have a sample ready to output. (Remember the sine_rom takes a cycle!)
sample_out[17:0]	output	The 18-bit audio sample output for our note that we generate.

--- no changes yet to ---
Wave Capture

Wave Display

Wave Display Top