De1-SoC Board Audio Output Using the WM8731 Audio Codec

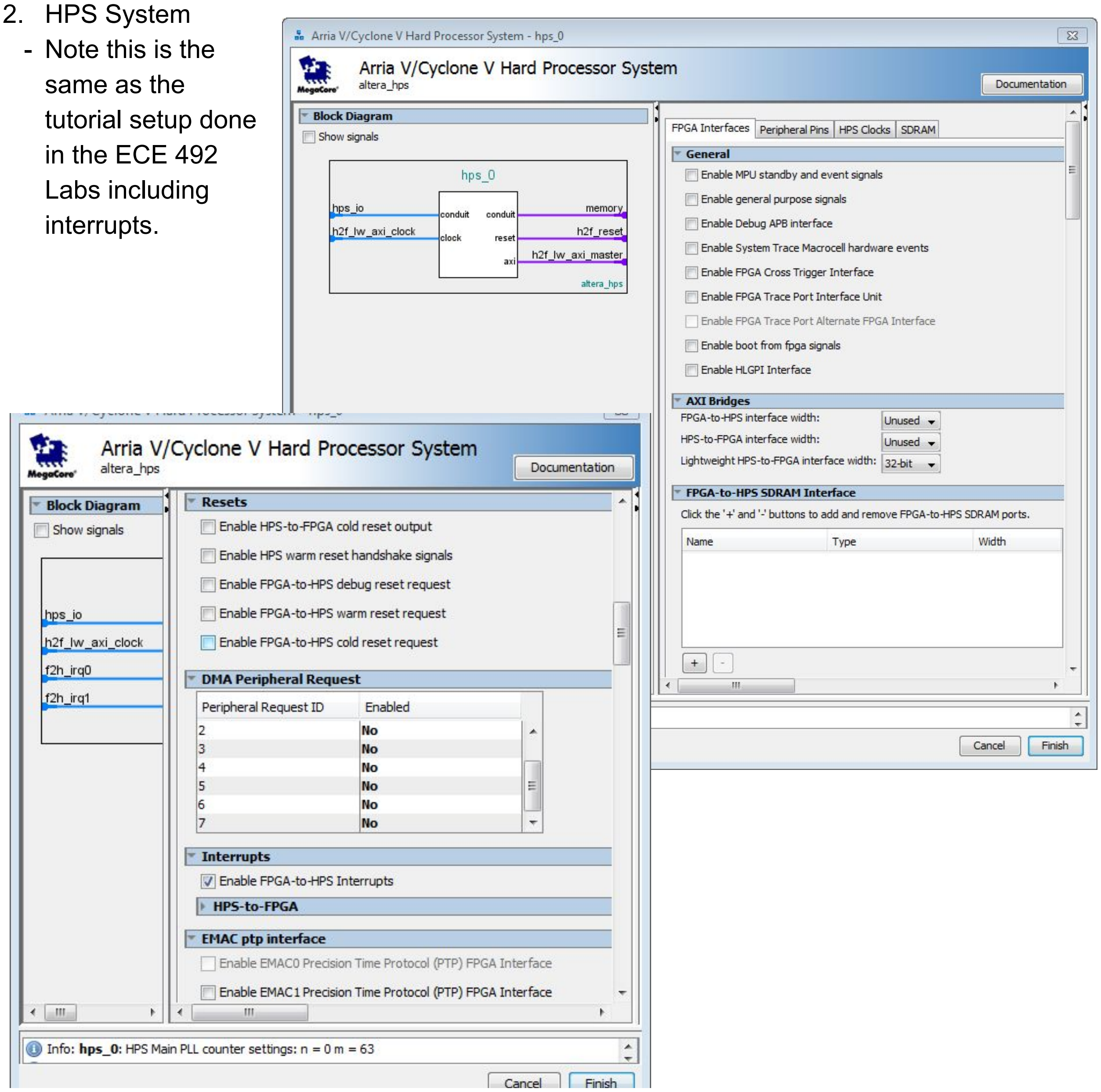
Introduction

These notes will show how to get the WM8731 audio codec configured and running, as well as outputting a simple frequency sine wave.

Hardware Procedure

The following images show the minimum required components in the Qsys system as well as the required settings.

1. Clock
    - Use the default clock source created by the New Project Wizard
2. HPS System
    - Note this is the same as the tutorial setup done in the ECE 492 Labs including interrupts.



General

- [ ] Enable MPU standby and event signals
- [ ] Enable general purpose signals
- [ ] Enable Debug APB interface
- [ ] Enable System Trace Macrocell hardware events
- [ ] Enable FPGA Cross Trigger Interface
- [ ] Enable FPGA Trace Port Interface Unit
- [ ] Enable FPGA Trace Port Alternate FPGA Interface
- [ ] Enable boot from fpga signals
- [ ] Enable HLGPI Interface

AXI Bridges

FPGA-to-HPS interface width:	Unused
HPS-to-FPGA interface width:	Unused
Lightweight HPS-to-FPGA interface width:	32-bit

FPGA-to-HPS SDRAM Interface

Click the '+' and '-' buttons to add and remove FPGA-to-HPS SDRAM ports.

Name	Type	Width

Resets

- [ ] Enable HPS-to-FPGA cold reset output
- [ ] Enable HPS warm reset handshake signals
- [ ] Enable FPGA-to-HPS debug reset request
- [ ] Enable FPGA-to-HPS warm reset request
- [ ] Enable FPGA-to-HPS cold reset request

DMA Peripheral Request

Peripheral Request ID	Enabled
2	No
3	No
4	No
5	No
6	No
7	No

Interrupts

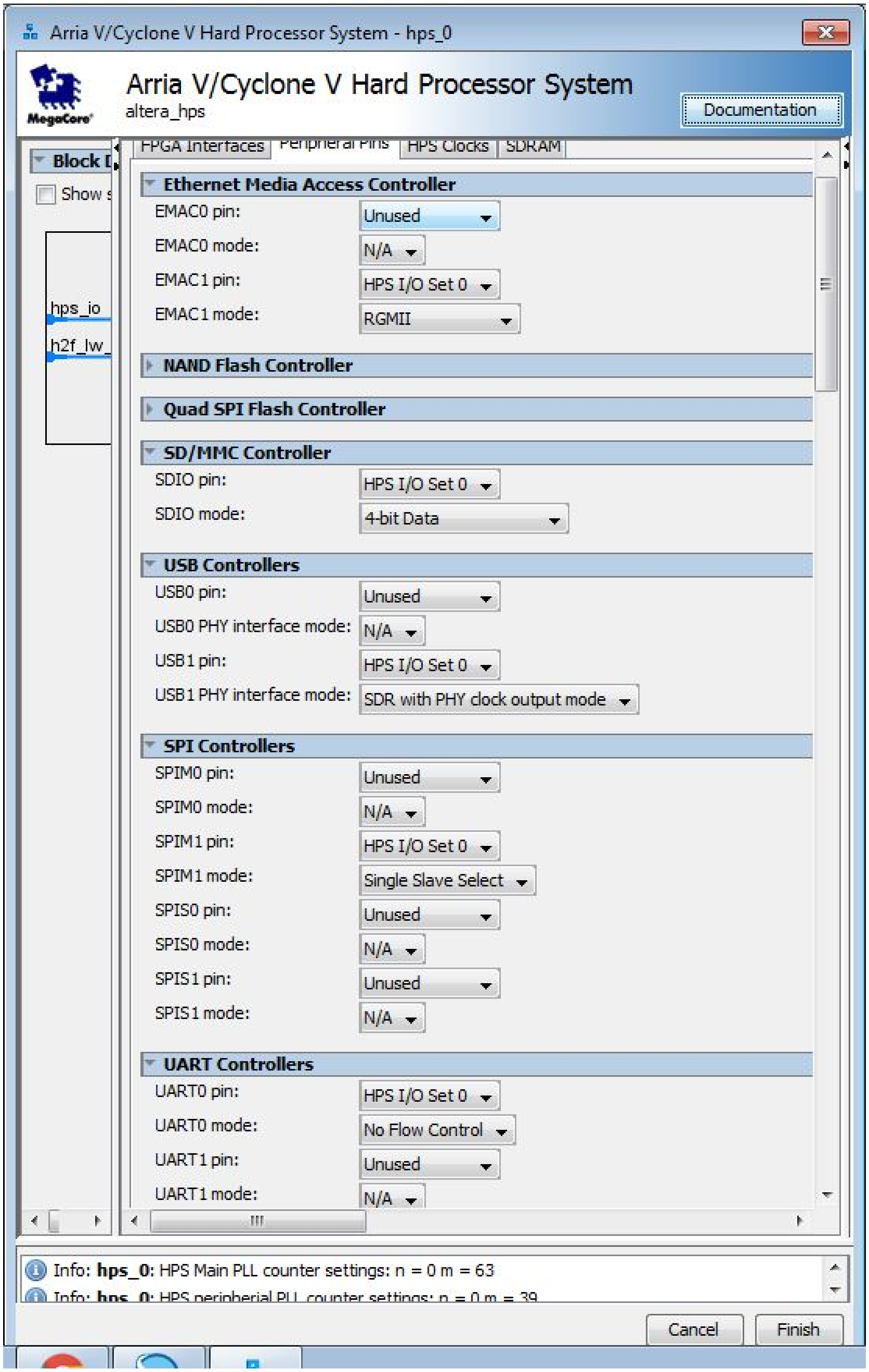
- [✓] Enable FPGA-to-HPS Interrupts

HPS-to-FPGA

EMAC ptp interface

- [ ] Enable EMAC0 Precision Time Protocol (PTP) FPGA Interface
- [ ] Enable EMAC1 Precision Time Protocol (PTP) FPGA Interface

Info: **hps_0:** HPS Main PLL counter settings: n = 0 m = 63



Ethernet Media Access Controller

EMAC0 pin:	Unused
EMAC0 mode:	N/A
EMAC1 pin:	HPS I/O Set 0
EMAC1 mode:	RGMII

NAND Flash Controller

Quad SPI Flash Controller

SD/MMC Controller

SDIO pin:	HPS I/O Set 0
SDIO mode:	4-bit Data

USB Controllers

USB0 pin:	Unused
USB0 PHY interface mode:	N/A
USB1 pin:	HPS I/O Set 0
USB1 PHY interface mode:	SDR with PHY clock output mode

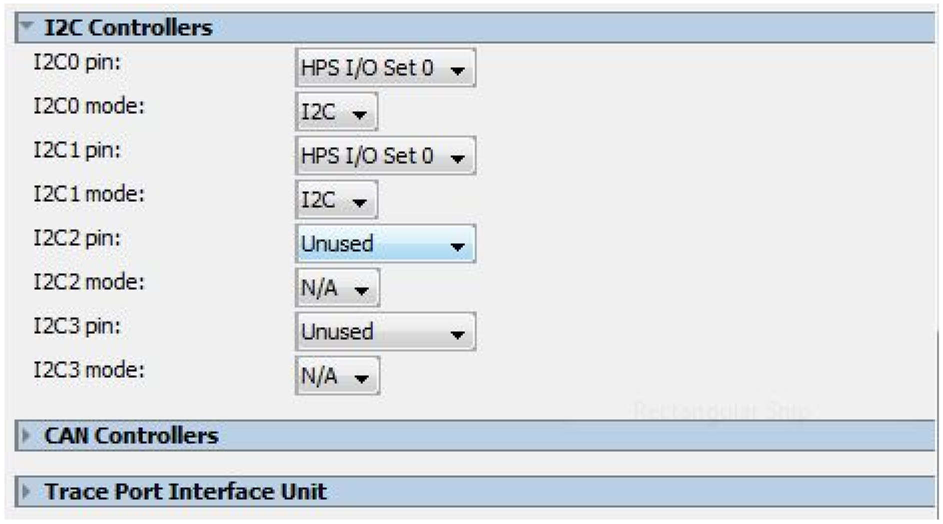
SPI Controllers

SPIM0 pin:	Unused
SPIM0 mode:	N/A
SPIM1 pin:	HPS I/O Set 0
SPIM1 mode:	Single Slave Select
SPIS0 pin:	Unused
SPIS0 mode:	N/A
SPIS1 pin:	Unused
SPIS1 mode:	N/A

UART Controllers

UART0 pin:	HPS I/O Set 0
UART0 mode:	No Flow Control
UART1 pin:	Unused
UART1 mode:	N/A

```
Info: hps_0: HPS Main PLL counter settings: n = 0 m = 63
Info: hps_0: HPS peripherial PLL counter settings: n = 0 m = 39
```



I2C Controllers

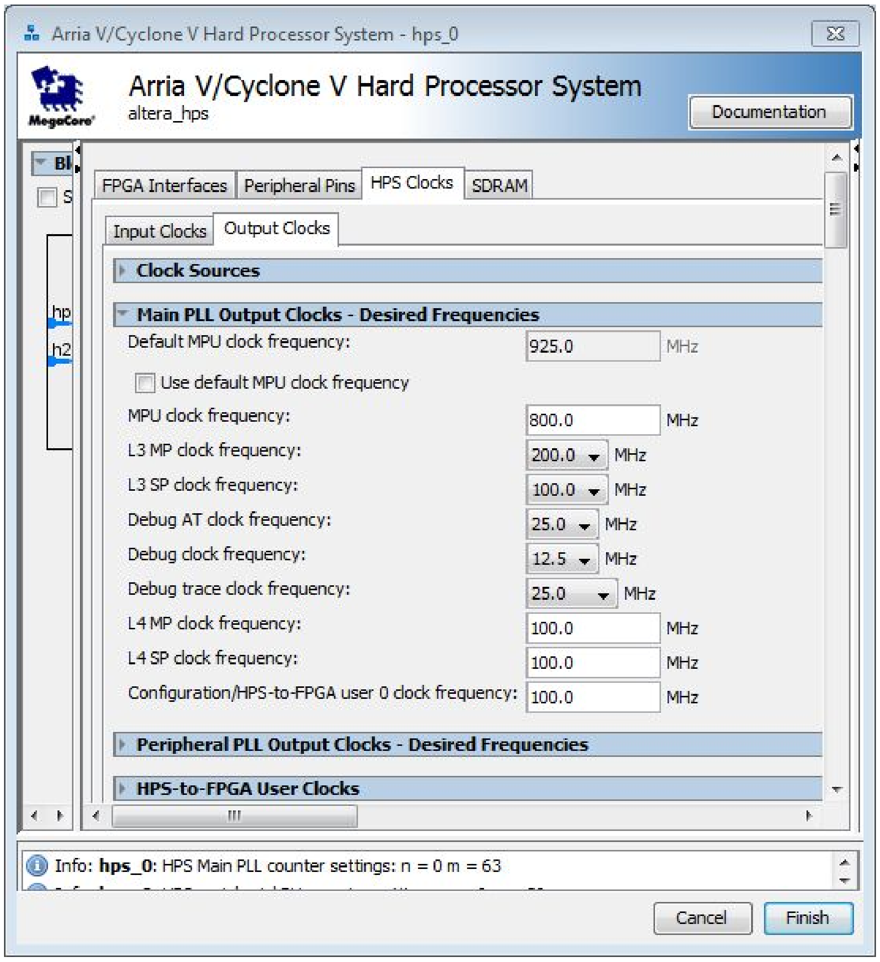
I2C0 pin:	HPS I/O Set 0
I2C0 mode:	I2C
I2C1 pin:	HPS I/O Set 0
I2C1 mode:	I2C
I2C2 pin:	Unused
I2C2 mode:	N/A
I2C3 pin:	Unused
I2C3 mode:	N/A

CAN Controllers

Trace Port Interface Unit

In the Peripheral Mux Table click on the GPIO09, GPIO35, GPIO40, GPIO48, GPIO53, GPIO54 and GPIO61 pins to appropriately map them manually. .

Under the HPS Clocks tab leave the Input Clocks tab as the default.
The output clocks are as follows:



Arria V/Cyclone V Hard Processor System
altera_hps

Clock Sources

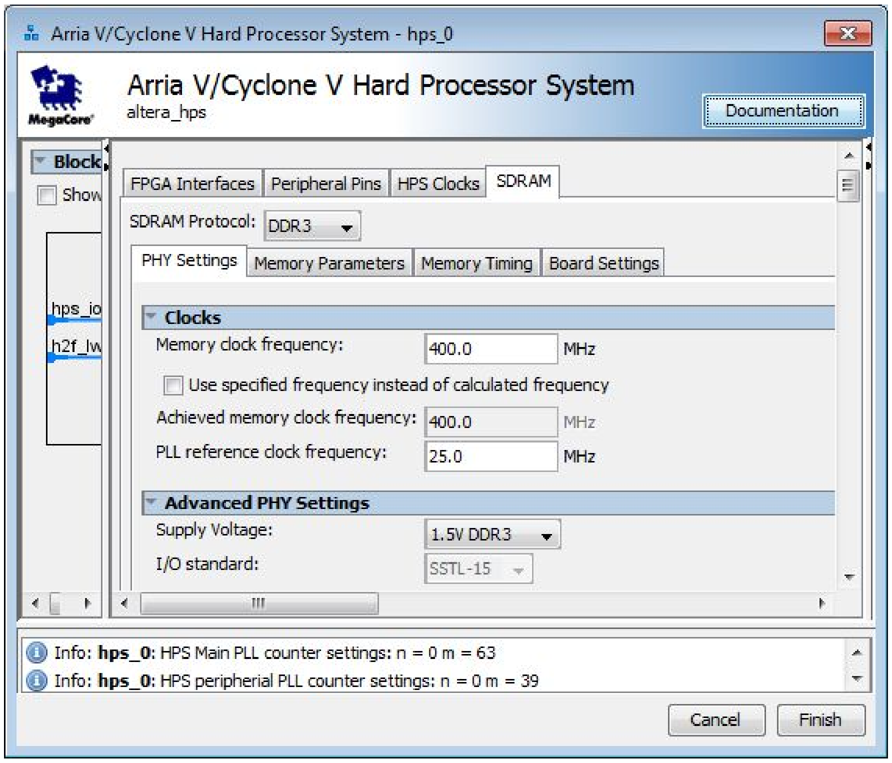
Main PLL Output Clocks - Desired Frequencies

Default MPU clock frequency:	925.0	MHz
☐ Use default MPU clock frequency		
MPU clock frequency:	800.0	MHz
L3 MP clock frequency:	200.0	MHz
L3 SP clock frequency:	100.0	MHz
Debug AT clock frequency:	25.0	MHz
Debug clock frequency:	12.5	MHz
Debug trace clock frequency:	25.0	MHz
L4 MP clock frequency:	100.0	MHz
L4 SP clock frequency:	100.0	MHz
Configuration/HPS-to-FPGA user 0 clock frequency:	100.0	MHz

Peripheral PLL Output Clocks - Desired Frequencies

HPS-to-FPGA User Clocks

Info: **hps_0**: HPS Main PLL counter settings: n = 0 m = 63


Arria V/Cyclone V Hard Processor System - hps_0

Arria V/Cyclone V Hard Processor System
altera_hps

FPGA Interfaces | Peripheral Pins | HPS Clocks | SDRAM

SDRAM Protocol: DDR3

PHY Settings | Memory Parameters | Memory Timing | Board Settings

Clocks

Memory clock frequency: 400.0 MHz

☐ Use specified frequency instead of calculated frequency

Achieved memory clock frequency: 400.0 MHz

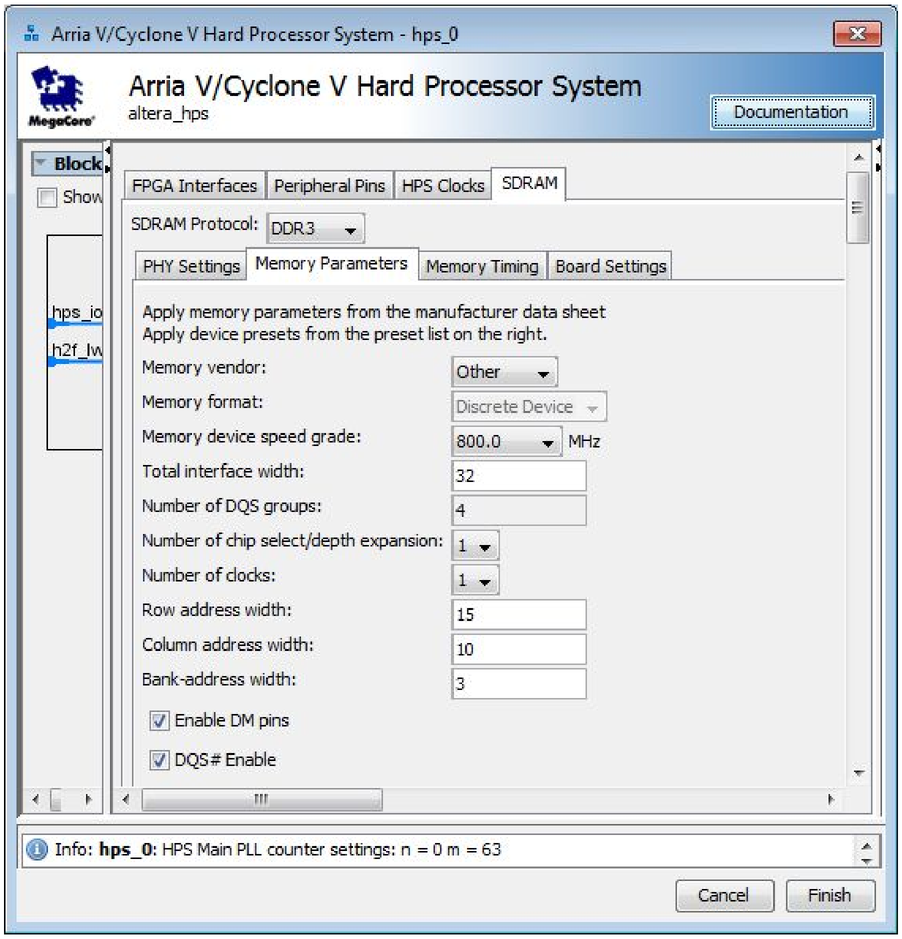
PLL reference clock frequency: 25.0 MHz

Advanced PHY Settings

Supply Voltage: 1.5V DDR3

I/O standard: SSTL-15

Info: **hps_0**: HPS Main PLL counter settings: n = 0 m = 63
Info: **hps_0**: HPS peripherial PLL counter settings: n = 0 m = 39


Arria V/Cyclone V Hard Processor System - hps_0

Arria V/Cyclone V Hard Processor System
altera_hps

FPGA Interfaces | Peripheral Pins | HPS Clocks | SDRAM

SDRAM Protocol: DDR3

PHY Settings | Memory Parameters | Memory Timing | Board Settings

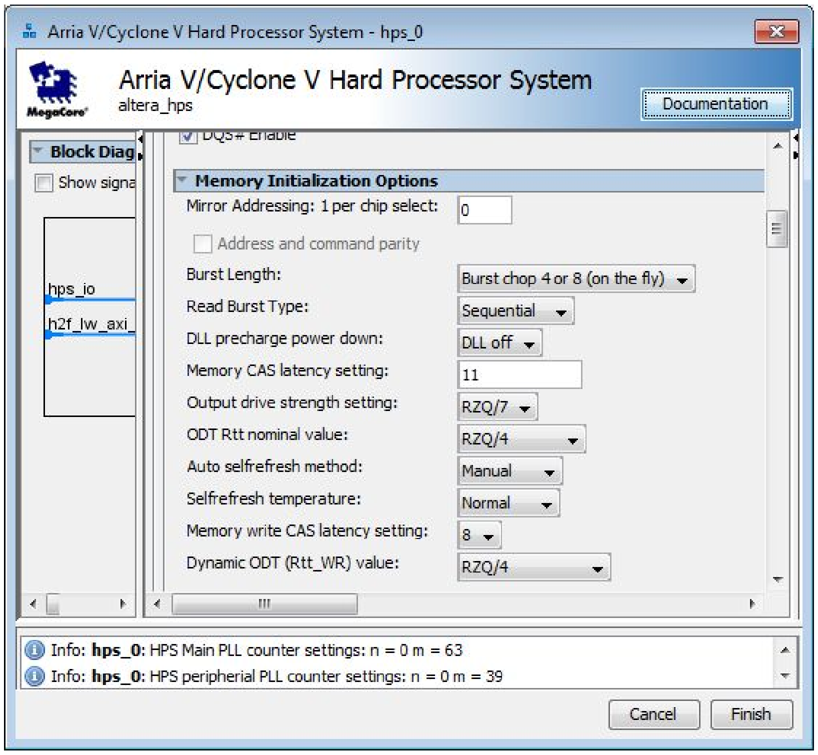
Apply memory parameters from the manufacturer data sheet
Apply device presets from the preset list on the right.

Memory vendor: Other
Memory format: Discrete Device
Memory device speed grade: 800.0 MHz
Total interface width: 32
Number of DQS groups: 4
Number of chip select/depth expansion: 1
Number of clocks: 1
Row address width: 15
Column address width: 10
Bank-address width: 3

☑ Enable DM pins

☑ DQS# Enable

Info: **hps_0**: HPS Main PLL counter settings: n = 0 m = 63


Arria V/Cyclone V Hard Processor System - hps_0
Arria V/Cyclone V Hard Processor System
altera_hps
Documentation
Block Diag
Show signa
hps_io
h2f_lw_axi_
Cancel
Finish

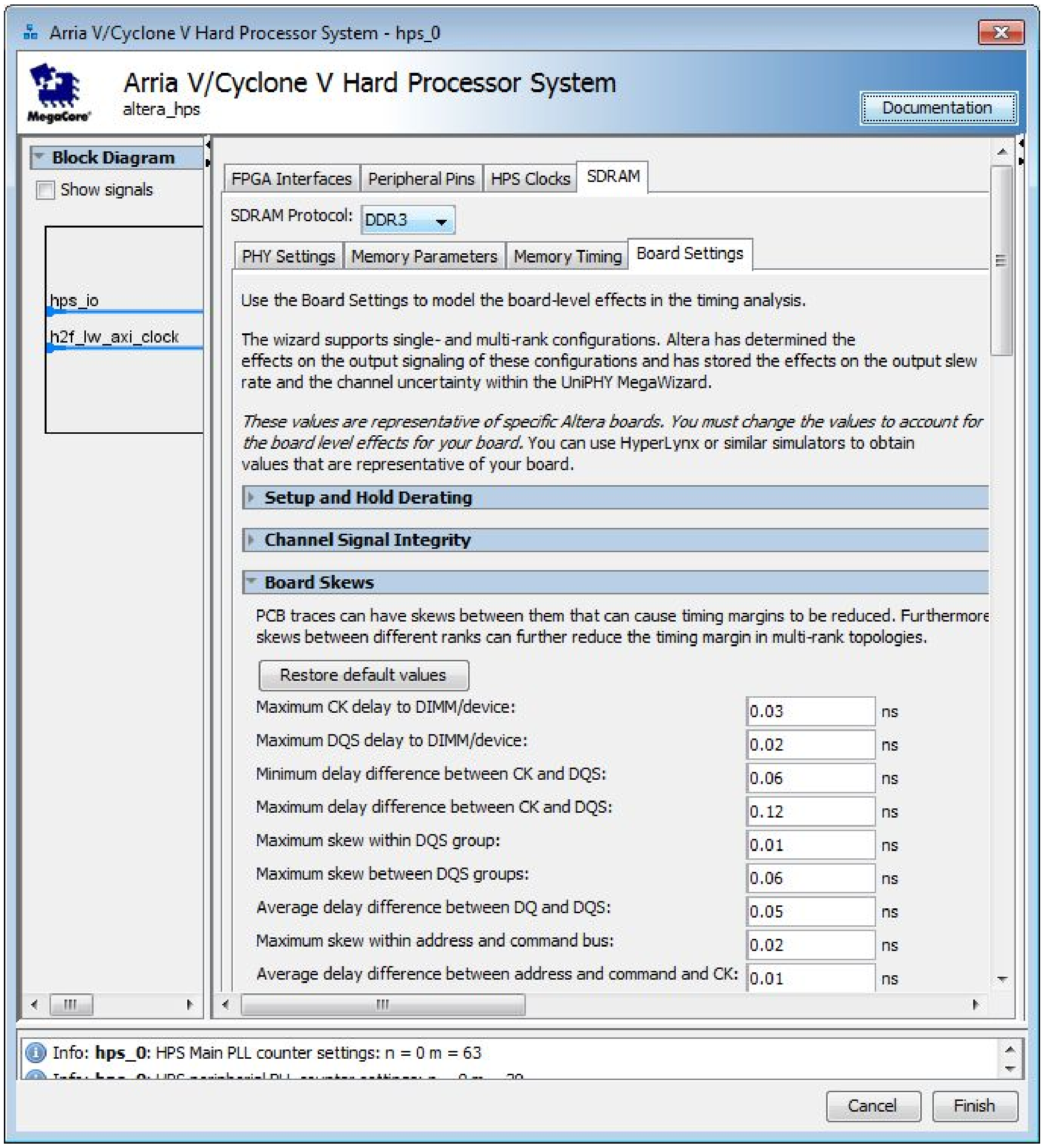
☑ DQS# Enable

Memory Initialization Options

Mirror Addressing: 1 per chip select:	0
☐ Address and command parity	
Burst Length:	Burst chop 4 or 8 (on the fly)
Read Burst Type:	Sequential
DLL precharge power down:	DLL off
Memory CAS latency setting:	11
Output drive strength setting:	RZQ/7
ODT Rtt nominal value:	RZQ/4
Auto selfrefresh method:	Manual
Selfrefresh temperature:	Normal
Memory write CAS latency setting:	8
Dynamic ODT (Rtt_WR) value:	RZQ/4

```
Info: hps_0: HPS Main PLL counter settings: n = 0 m = 63
Info: hps_0: HPS peripherial PLL counter settings: n = 0 m = 39
```

Under Board Settings -> Setup and Hold Derating and Channel Signal Integrity, leave Altera's default settings checked.
The remaining configuration is as follows:



Arria V/Cyclone V Hard Processor System - hps_0

Arria V/Cyclone V Hard Processor System
altera_hps

SDRAM Protocol: DDR3

Use the Board Settings to model the board-level effects in the timing analysis.

The wizard supports single- and multi-rank configurations. Altera has determined the effects on the output signaling of these configurations and has stored the effects on the output slew rate and the channel uncertainty within the UniPHY MegaWizard.

These values are representative of specific Altera boards. You must change the values to account for the board level effects for your board. You can use HyperLynx or similar simulators to obtain values that are representative of your board.

Setup and Hold Derating

Channel Signal Integrity

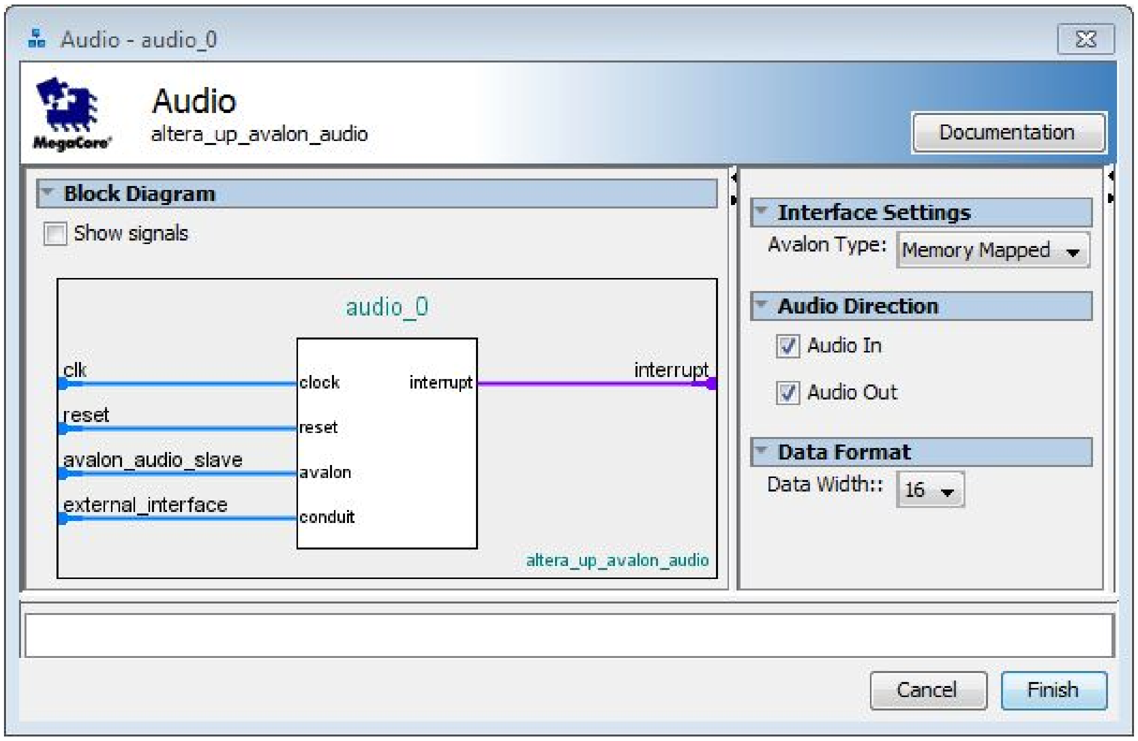
Board Skews

PCB traces can have skews between them that can cause timing margins to be reduced. Furthermore skews between different ranks can further reduce the timing margin in multi-rank topologies.

Maximum CK delay to DIMM/device:	0.03	ns
Maximum DQS delay to DIMM/device:	0.02	ns
Minimum delay difference between CK and DQS:	0.06	ns
Maximum delay difference between CK and DQS:	0.12	ns
Maximum skew within DQS group:	0.01	ns
Maximum skew between DQS groups:	0.06	ns
Average delay difference between DQ and DQS:	0.05	ns
Maximum skew within address and command bus:	0.02	ns
Average delay difference between address and command and CK:	0.01	ns

Info: **hps_0**: HPS Main PLL counter settings: n = 0 m = 63

3. Audio Core (University Program -> Audio and Video -> Audio)



Interface Settings

Avalon Type: Memory Mapped

Audio Direction

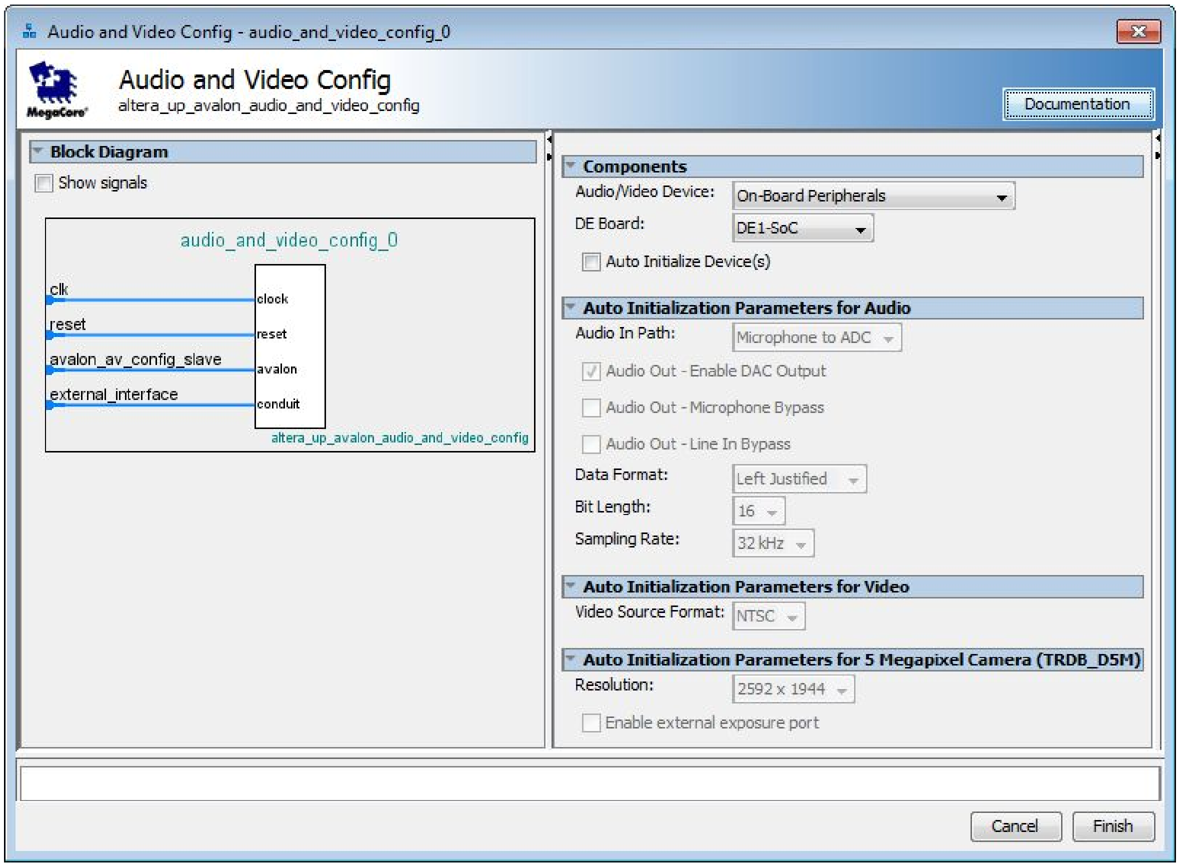
☑ Audio In

☑ Audio Out

Data Format

Data Width:: 16

4. Audio and Video config (University Program -> Audio & Video -> Audio and Video Config)



Components

Audio/Video Device: On-Board Peripherals

DE Board: DE1-SoC

☐ Auto Initialize Device(s)

Auto Initialization Parameters for Audio

Audio In Path: Microphone to ADC

☑ Audio Out - Enable DAC Output

☐ Audio Out - Microphone Bypass

☐ Audio Out - Line In Bypass

Data Format: Left Justified

Bit Length: 16

Sampling Rate: 32 kHz

Auto Initialization Parameters for Video

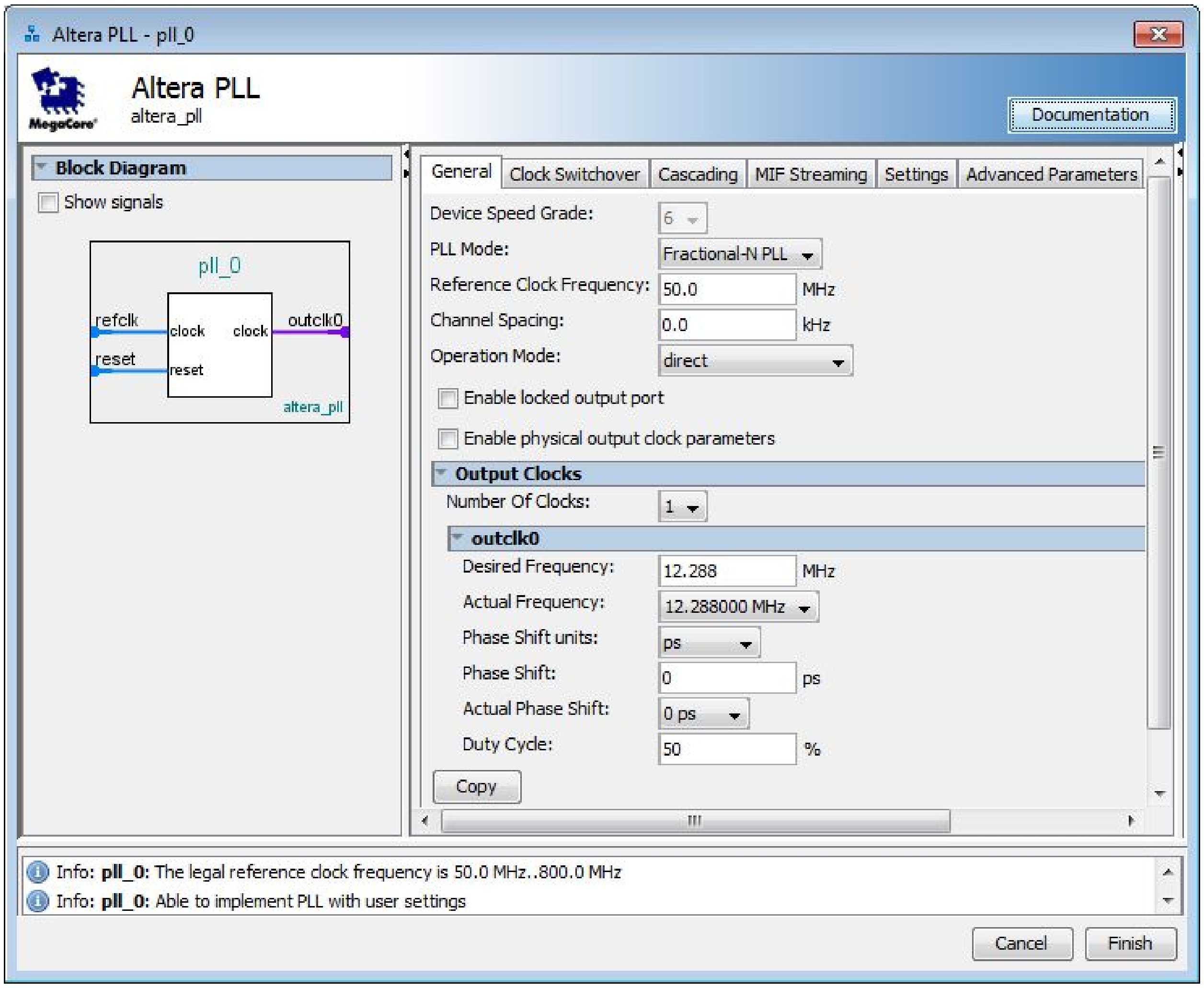
Video Source Format: NTSC

Auto Initialization Parameters for 5 Megapixel Camera (TRDB_D5M)

Resolution: 2592 x 1944

☐ Enable external exposure port

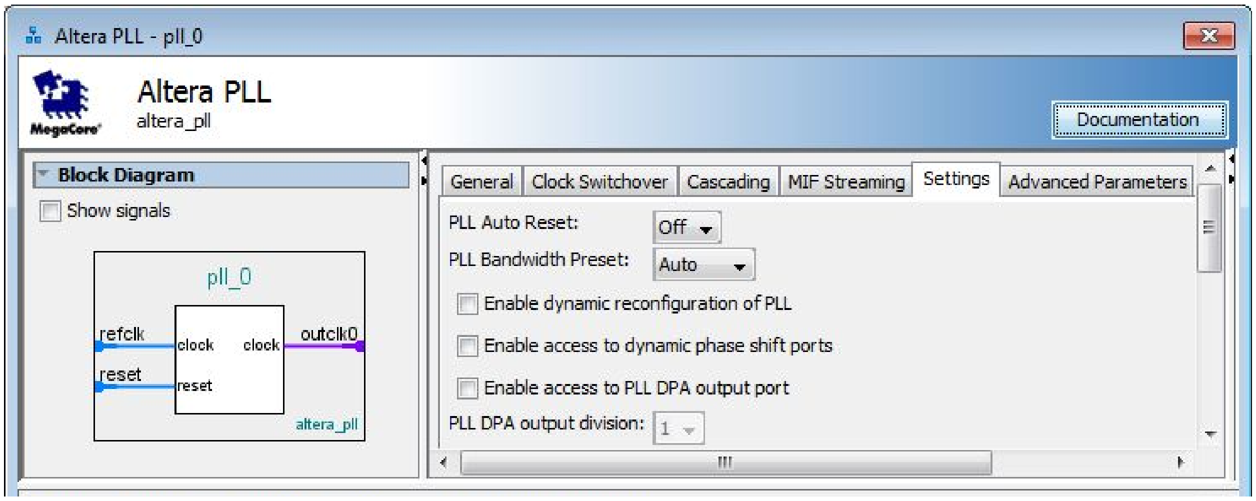
5. Altera PLL (Basic Functions -> Clocks; PLLs and Resets -> PLL -> Altera PLL)



Device Speed Grade: 6
PLL Mode: Fractional-N PLL
Reference Clock Frequency: 50.0 MHz
Channel Spacing: 0.0 kHz
Operation Mode: direct
Enable locked output port
Enable physical output clock parameters
Output Clocks
Number Of Clocks: 1
outclk0
Desired Frequency: 12.288 MHz
Actual Frequency: 12.288000 MHz
Phase Shift units: ps
Phase Shift: 0 ps
Actual Phase Shift: 0 ps
Duty Cycle: 50 %

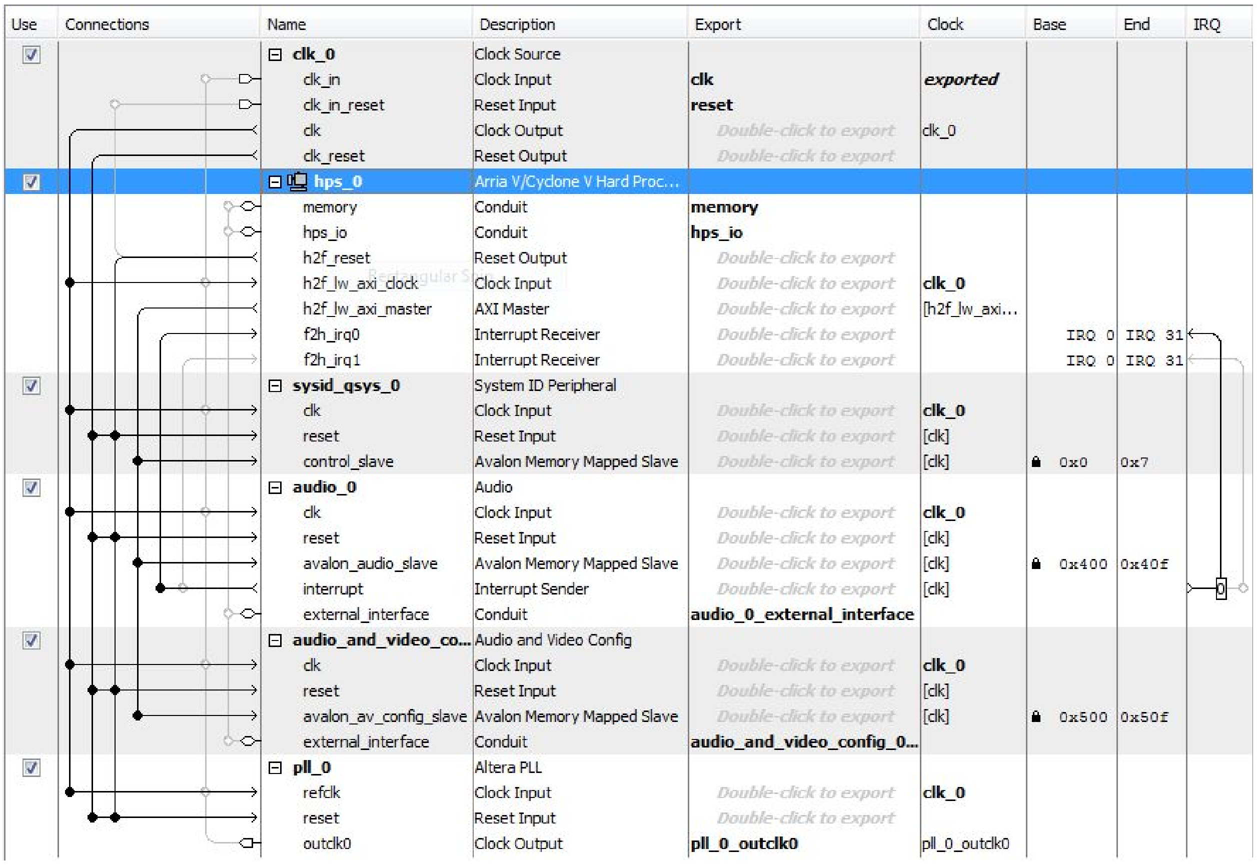
Info: **pll_0:** The legal reference clock frequency is 50.0 MHz..800.0 MHz
Info: **pll_0:** Able to implement PLL with user settings

Under the Clock Switchover, Cascading, and MIF Streaming tabs, nothing needs to be enabled
Finally, the Settings tab is as follows:



PLL Auto Reset: Off
PLL Bandwidth Preset: Auto
Enable dynamic reconfiguration of PLL
Enable access to dynamic phase shift ports
Enable access to PLL DPA output port
PLL DPA output division: 1

Once these components are added, connect them according to the following image and generate the VHDL in Qsys. Be sure to also include the sysid_qsys_0 component with the default configurations of 0x0.



Use	Connections	Name	Description	Export	Clock	Base	End	IRQ
✓		**clk_0**	Clock Source					
		clk_in	Clock Input	**clk**	***exported***			
		clk_in_reset	Reset Input	**reset**				
		clk	Clock Output	*Double-click to export*	clk_0			
		clk_reset	Reset Output	*Double-click to export*				
✓		**hps_0**	Arria V/Cyclone V Hard Proc...					
		memory	Conduit	**memory**				
		hps_io	Conduit	**hps_io**				
		h2f_reset	Reset Output	*Double-click to export*				
		h2f_lw_axi_clock	Clock Input	*Double-click to export*	**clk_0**			
		h2f_lw_axi_master	AXI Master	*Double-click to export*	[h2f_lw_axi...			
		f2h_irq0	Interrupt Receiver	*Double-click to export*		IRQ 0	IRQ 31	
		f2h_irq1	Interrupt Receiver	*Double-click to export*		IRQ 0	IRQ 31	
✓		**sysid_qsys_0**	System ID Peripheral					
		clk	Clock Input	*Double-click to export*	**clk_0**			
		reset	Reset Input	*Double-click to export*	[clk]			
		control_slave	Avalon Memory Mapped Slave	*Double-click to export*	[clk]	0x0	0x7	
✓		**audio_0**	Audio					
		clk	Clock Input	*Double-click to export*	**clk_0**			
		reset	Reset Input	*Double-click to export*	[clk]			
		avalon_audio_slave	Avalon Memory Mapped Slave	*Double-click to export*	[clk]	0x400	0x40f	
		interrupt	Interrupt Sender	*Double-click to export*	[clk]			0
		external_interface	Conduit	**audio_0_external_interface**				
✓		**audio_and_video_co...**	Audio and Video Config					
		clk	Clock Input	*Double-click to export*	**clk_0**			
		reset	Reset Input	*Double-click to export*	[clk]			
		avalon_av_config_slave	Avalon Memory Mapped Slave	*Double-click to export*	[clk]	0x500	0x50f	
		external_interface	Conduit	**audio_and_video_config_0...**				
✓		**pll_0**	Altera PLL					
		refclk	Clock Input	*Double-click to export*	**clk_0**			
		reset	Reset Input	*Double-click to export*				
		outclk0	Clock Output	**pll_0_outclk0**	pll_0_outclk0			

Note you will have four warnings pertaining to interrupts upon successful generation. This is fine and will still provide functional audio.
Also note, the base addresses for the audio components are set at 0x400 and 0x500. You may change these to suit your design as long as they are updated in the audio.h and audio_cfg.h files (discussed later).

Next, ensure your toplevel VHDL file in Quartus is connected as follows. A template is provided in audio.vhd for reference.

A. Declare these pins in the entity (in addition to all of the hps signals):

```
-- I2C Interface
        FPGA_I2C_SCLK                 : out std_logic;
        FPGA_I2C_SDAT                 : inout std_logic := 'X';
-- Audio
        AUD_ADCDAT                            : in std_logic := 'X';
        AUD_ADCLRCK                           : in std_logic := 'X';
        AUD_BCLK                              : in std_logic := 'X';
        AUD_DACDAT                            : out std_logic;
        AUD_DACLRCK                          : in std_logic := 'X';
        AUD_XCK                              : out std_logic
```

B. Add these ports in the architecture's component declaration

```
audio_0_external_interface_ADCDAT     : in    std_logic           := 'X';        -- ADCDAT
audio_0_external_interface_ADCLRCK    : in    std_logic           := 'X';       -- ADCLRCK
audio_0_external_interface_BCLK       : in    std_logic           := 'X';          -- BCLK
audio_0_external_interface_DACDAT     : out   std_logic;                          -- DACDAT
audio_0_external_interface_DACLRCK    : in    std_logic           := 'X';        -- DACLRCK
audio_and_video_config_0_external_interface_SDAT : inout std_logic    := 'X';        -- SDAT
audio_and_video_config_0_external_interface_SCLK : out   std_logic;                  -- SCLK
pll_0_outclk0_clk                    : out   std_logic;                        -- clk
```

C. Port map the pins to the ports in the instantiation:

```
audio_and_video_config_0_external_interface_SDAT => FPGA_I2C_SDAT,
audio_and_video_config_0_external_interface_SCLK => FPGA_I2C_SCLK,
audio_0_external_interface_ADCDAT                => AUD_ADCDAT,
audio_0_external_interface_ADCLRCK               => AUD_ADCLRCK,
audio_0_external_interface_BCLK                  => AUD_BCLK,
audio_0_external_interface_DACDAT                => AUD_DACDAT,
    audio_0_external_interface_DACLRCK               => AUD_DACLRCK,
    pll_0_outclk0_clk                                => AUD_XCK
```

Ensure that the .qip and .tcl scripts have also been added to the project before compiling. Also be sure to run the appropriate .tcl scripts beforehand. Then you may compile the design.

A zipped archive of a template project is included with this report for your reference.

Software Procedure

While only using the LINEOUT to the speakers, the audio core has four writable registers starting at the base address. For a full description of what each register does, see the first reference. In order to send audio out, you must write to both the left and right audio channels, if only one channel is written to and one FIFO buffer remains empty, the codec will not output audio. Although the audio and video config component can be setup in Qsys, this configuration can be overridden by writing to the components registers. These can be written to by the function write_audio_cfg_register(REGISTER ADDRESS, REGISTER VALUE). Below is the full register configuration used in the example code. However, something to note in the 0x8 register, is that bit 6 must be set to 0 as to enable the Slave Mode.

```
write_audio_cfg_register(0x0, 0x17);
write_audio_cfg_register(0x1, 0x17);
write_audio_cfg_register(0x2, 0x7F);
write_audio_cfg_register(0x3, 0x7F);
write_audio_cfg_register(0x4, 0x15);
write_audio_cfg_register(0x5, 0x06);
write_audio_cfg_register(0x6, 0x00);
write_audio_cfg_register(0x7, 0x4D);
write_audio_cfg_register(0x8, 0x18);
write_audio_cfg_register(0x9, 0x01);
```

This should enable all of the correct settings for the audio core and it should be ready to be written to according to the 12.288 MHz clock specified in the Altera PLL.

Finally, to ensure the software can properly communicate with the hardware, specify the base addresses in the audio.h and audio_cfg.h files. Import all of the audio files into your project and use them as follows.

1. Generate a buffer of audio data of type INT32S*
2. Pass the buffer and frequency to the write function using write_audio_data(buffer, freq);

An example is shown in the attached app.c file. Please take note that the SAMPLING_RATE specified in app.c of 32000 Hz is dependent on the 12.288 MHz clock in the PLL. Changing the sampling rate to be say 44100 Hz requires further changes in the hardware to ensure the codec is reading out bytes at the same rate that the software is producing them.

References

[1] University of Toronto. *Digital Embedded Systems Lab: Audio Core.* [Online]. Available:http://www-ug.eecg.utoronto.ca/desl/nios_devices_SoC/dev_audio.html

[2] WOLFSON MICROELECTRONICS plc. (2005). *Portable Internet Audio CODEC with Headphone Driver and Programmable Sample Rates.* [Online]. Available:https://www.rockbox.org/wiki/pub/Main/DataSheets/WM8731_8731L.pdf

Attached Files

audio.c: Provides communication driver for writing to the audio buffers.
audio.h: Header file for audio core driver.
audio_cfg.c: Provides configurations for audio and video cores.
audio_cfg.h: Header file for audio and video core drivers.
app.c: Provides an example main.c and AppTask to produce audio output.
audio.qar: Archived Quartus/Qsys project example to provide functional audio.