Harvard John A. Paulson
School of Engineering
and Applied Sciences

ES100 - Design Review

Audio Processing on FPGAs

Mitchell Sharum

Define

Define

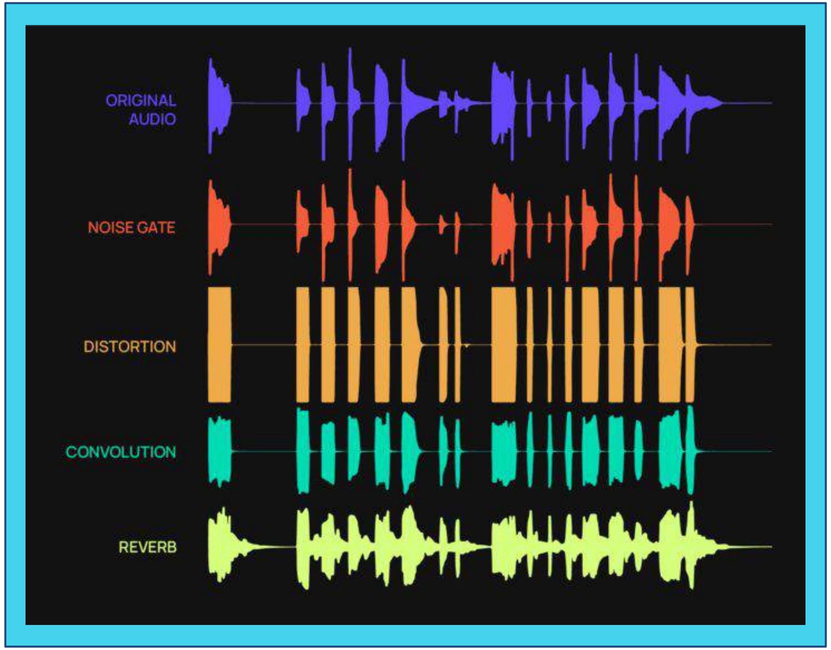
Design

Build

Q&A

Background

- Performing musicians require live processing of many audio effects
- These effects shape the sound of a performance
- These processing devices take in sound signals (represented by Voltages), altering them at the output

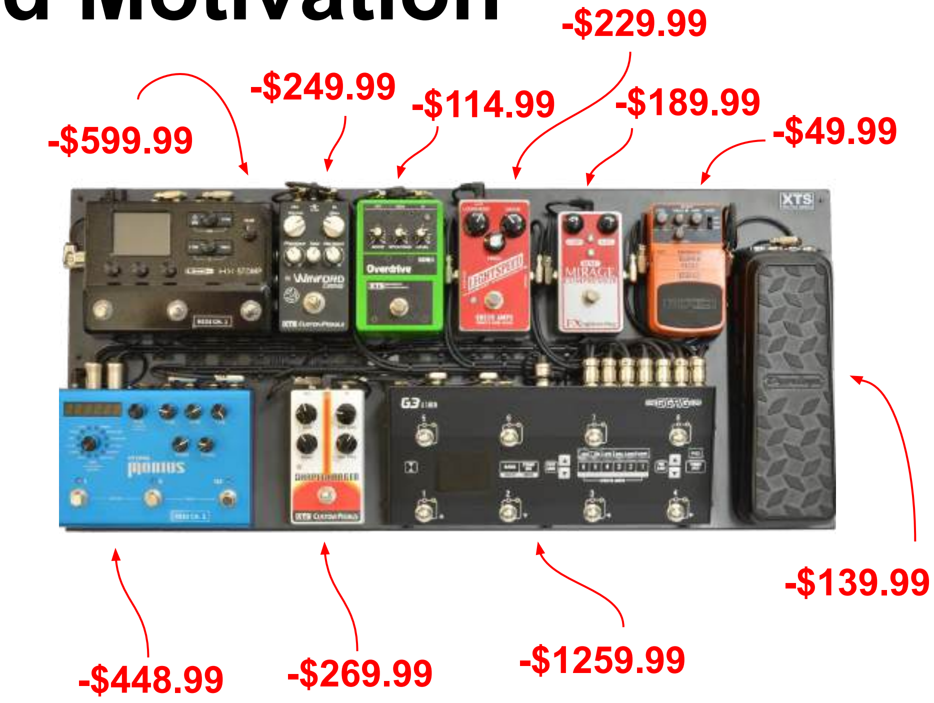


Current Solutions and Motivation

- Arrays of analog and digital pedals are by far the most common live effects (FX) processing solution
- Each discrete pedal typically instantiates one effect
- Pedals are very expensive, which makes effects selection a

Current Solutions and Motivation

- Arrays of analog and digital pedals are by far the most common live effects (FX) processing solution
- Each discrete pedal typically instantiates one effect
- Pedals are very expensive, which makes effects selection a



-$599.99

-$249.99

-$114.99

-$229.99

-$189.99

-$49.99

-$139.99

-$448.99

-$269.99

-$1259.99

Current Solutions and Motivation

- Digital audio workstations (DAWs) offer a great number of digital signal processing (DSP) effects
- These effects are all run on the CPU, which can result in unwanted latency
- An ideal FX processing solution would run at the hardware level, not via an app running on an OS

Problem Statement

Live performance environments often require that musicians process their instruments' electrical signals in order to achieve the expressive effects that shape their sound. Both digital audio workstations and arrays of effects pedals leave higher performance, greater reconfigurability, and lower prices to be desired.

Design

Define
Design
Build
Q&A

Technical Specifications (High Level)

Label	Requirement	Specification	Value	Measurement
LAT	Latency	Max tolerable latency	5 ms [4]	Oscilloscope
NFX	Audio Effects	Min number of supported FX	4	Counting
IGN	Input Gain	Min gain on instrument signal	10 dB	Oscilloscope
SNR	Signal-Noise Ratio	Max tolerable SNR	90 dB [5]	Oscilloscope

Table 1: High Level Technical Specifications

High Level Decisions (Hardware)

Field-Programmable Gate Array

- Very low latency
- Inexpensive at small-scale production
- Enables rapid prototyping and build for complex designs

Application Specific Integrated Circuit

- Extremely low latency
- Price only justifiable at large scale production
- Very long build process
- Highly specialized for efficiency, more design needs

High Level Decisions (Hardware)

Field-Programmable Gate Array

- Very low latency
- Inexpensive at small-scale production
- Enables rapid prototyping and build for complex designs

Application Specific Integrated Circuit

- Extremely low latency
- Price only justifiable at large scale production
- Very long build process
- Highly specialized for efficiency, more design needs

High Level Decisions (Software)

Register-Transfer Level (RTL)

- Hardware description languages (HDL) define electrical circuits
- Offers fine-grained control of all design components
- More complex design process, but allows for more specialized circuits

High-Level Synthesis (HLS)

- Code input generates hardware constructs automatically
- Faster implementation process facilitates ease of design
- Resulting circuits are often less efficient

High Level Decisions (Software)

Register-Transfer Level (RTL)

- Hardware description languages (HDL) define electrical circuits
- Offers fine-grained control of all design components
- More complex design process, but allows for more specialized circuits

High-Level Synthesis (HLS)

- Code input generates hardware constructs automatically
- Faster implementation process facilitates ease of design
- Resulting circuits are often less efficient

vhdl

SystemVerilog

C++

MATLAB®

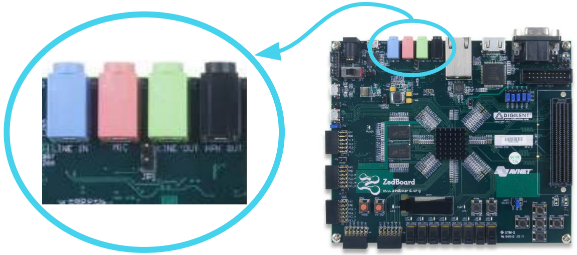
High Level Decisions (I/O Support)

PMOD I2S2 (Peripheral module)

- Peripheral ADC/DAC with 3.5mm audio jack I/O
- Must be synchronized with the host device’s primary clock
- Communicates via I2S protocol

ADAU1761 (Onboard audio codec)

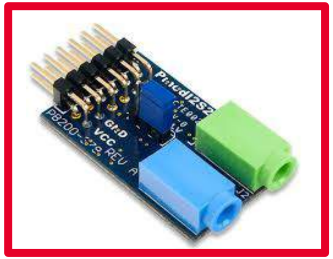
- Integrated codec means lower latency
- The ADAU1761 requires additional configuration via I^{2}C protocol
- Relies on Zynq 7000’s arm core



High Level Decisions (I/O Support)

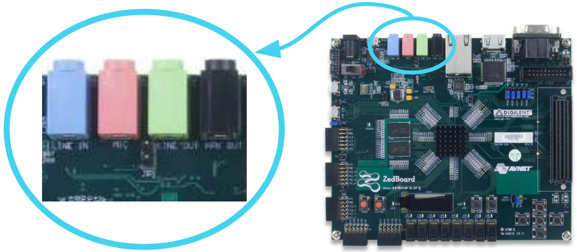
PMOD I2S2 (Peripheral module)

- Peripheral ADC/DAC with 3.5mm audio jack I/O
- Must be synchronized with the host device’s primary clock
- Communicates via I2S protocol



ADAU1761 (Onboard audio codec)

- Integrated codec means lower latency
- The ADAU1761 requires additional configuration via I^{2}C protocol
- Relies on Zynq 7000’s arm core



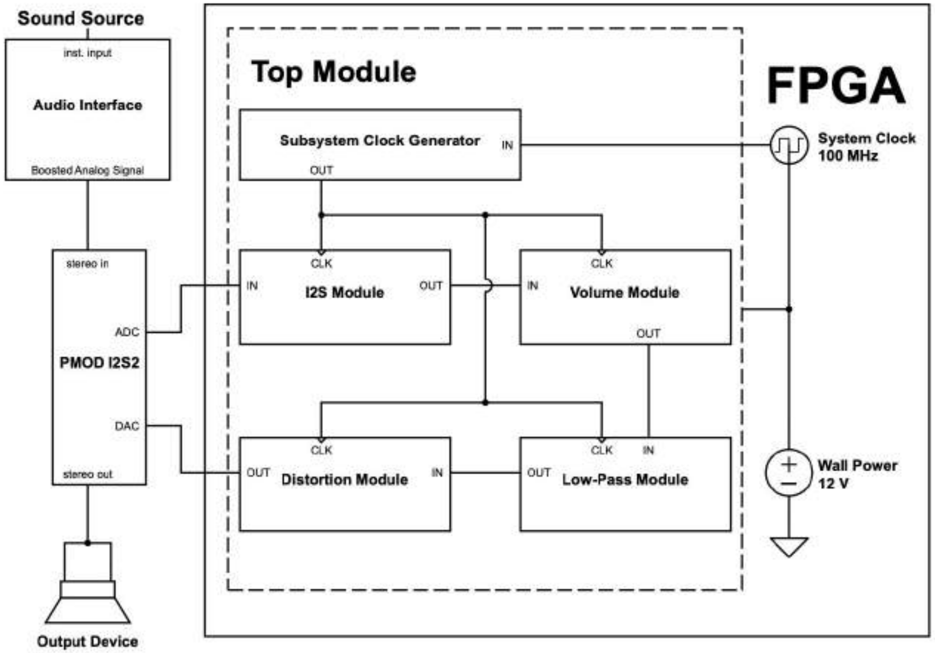
Technical Specifications (FPGA)

Label	Requirement	Specification	Value	Measurement
LUT	Lookup Tables	Max % of logic elements	20%	Vivado
BRM	Block RAM	Max % of on board memory	50%	Vivado
DSP	DSP Blocks	Max % of DSP processors	20%	Vivado

Table 2: FPGA Specific Technical Specifications

System Diagram

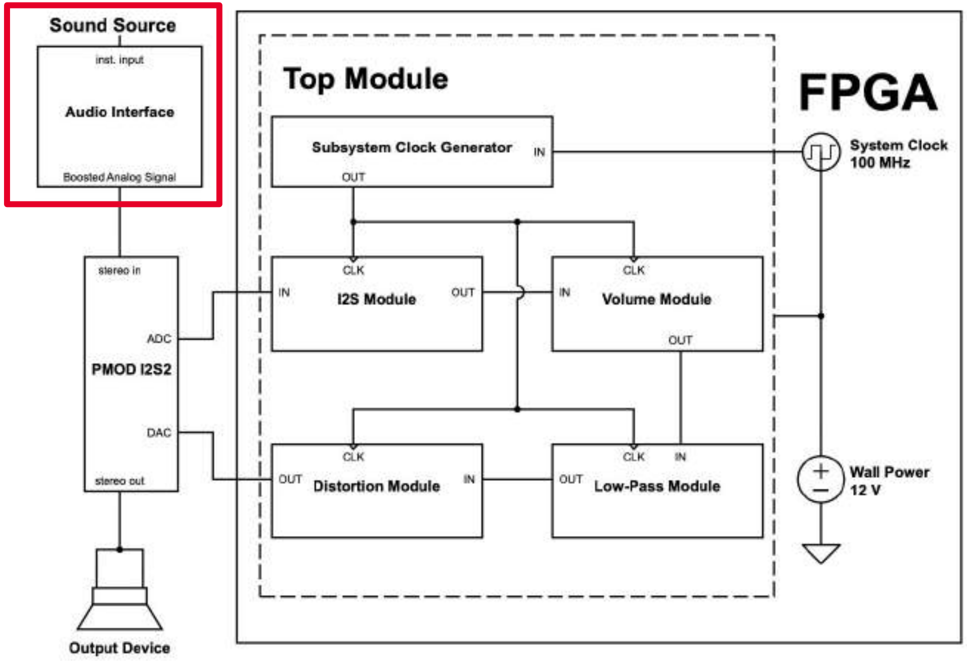
- Audio interface boosts input instrument signal
- PMOD ADC passes in samples of the signal
- I2S frames are processed
- Frames are passed to DAC to form analog wave
- Heard as music from an output device



Harvard John A. Paulsor School of Engineering and Applied Sciences

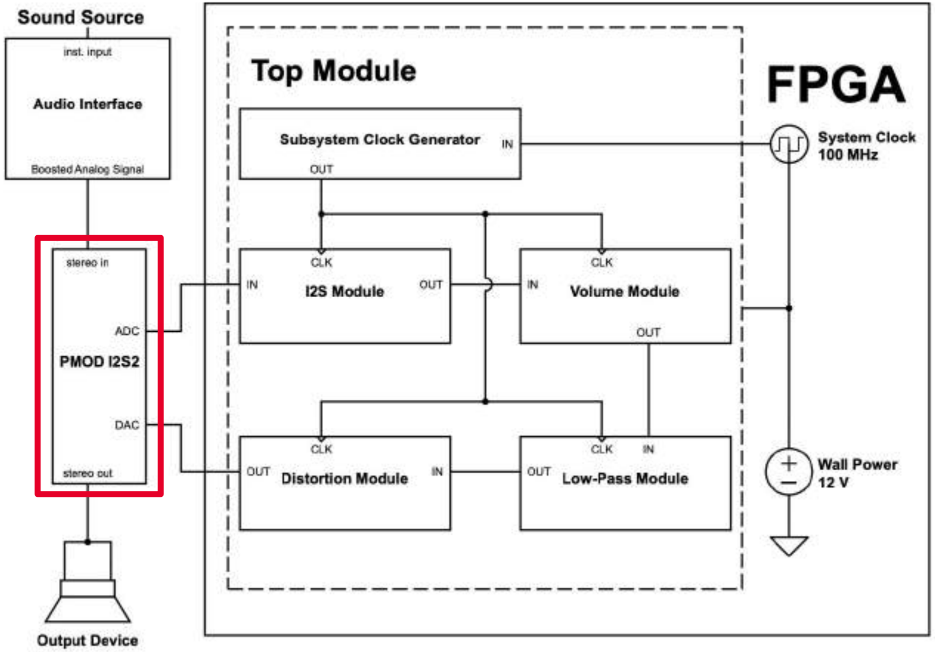
System Diagram

- Audio interface boosts input instrument signal
- PMOD ADC passes in samples of the signal
- I2S frames are processed
- Frames are passed to DAC to form analog wave
- Heard as music from an output device



System Diagram

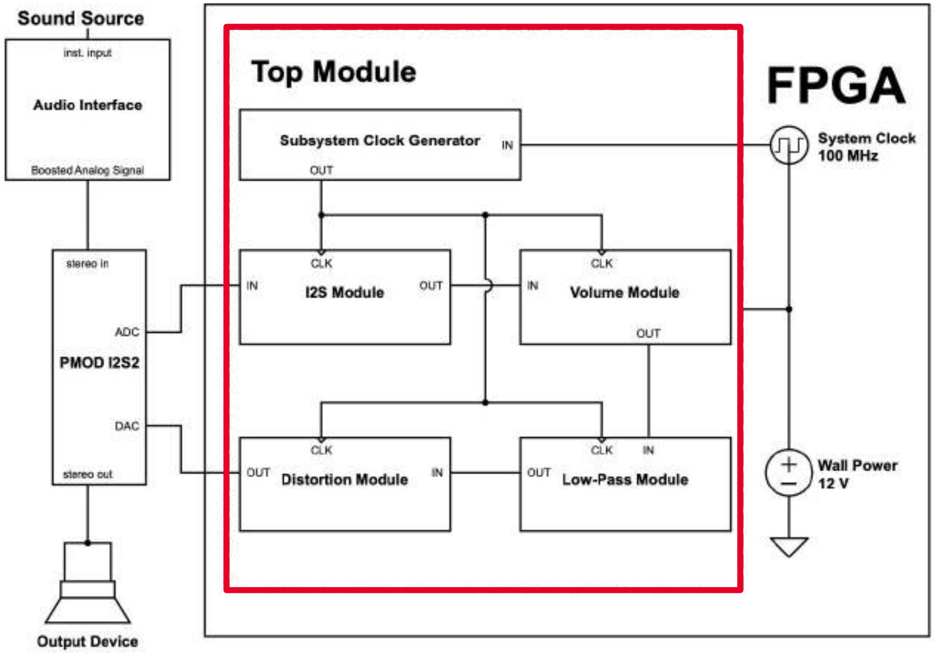
- Audio interface boosts input instrument signal
- PMOD ADC passes in samples of the signal
- I2S frames are processed
- Frames are passed to DAC to form analog wave
- Heard as music from an output device



Harvard John A. Paulson School of Engineering and Applied Sciences

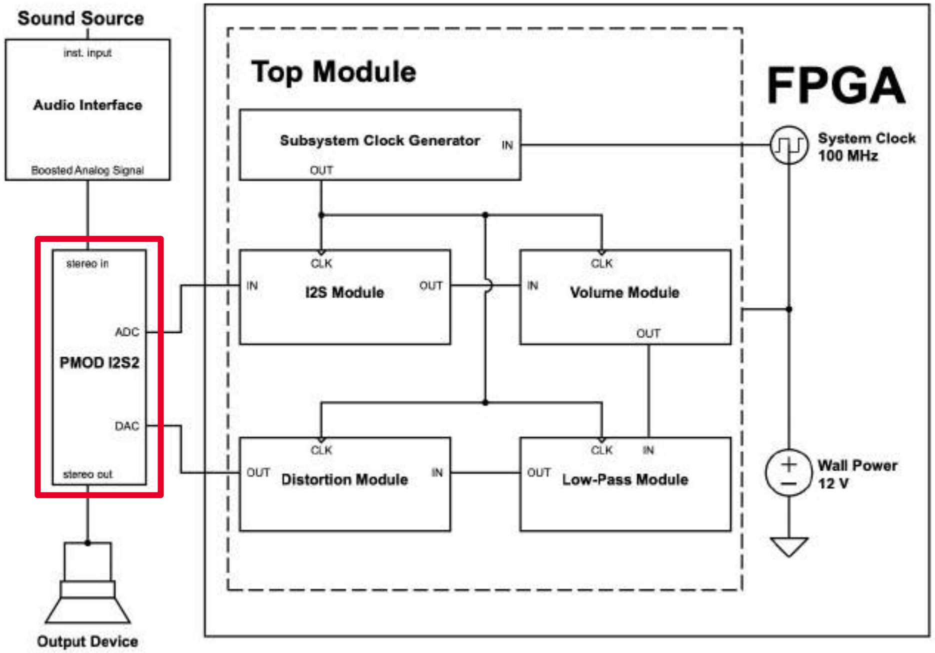
System Diagram

- Audio interface boosts input instrument signal
- PMOD ADC passes in samples of the signal
- I2S frames are processed
- Frames are passed to DAC to form analog wave
- Heard as music from an output device



System Diagram

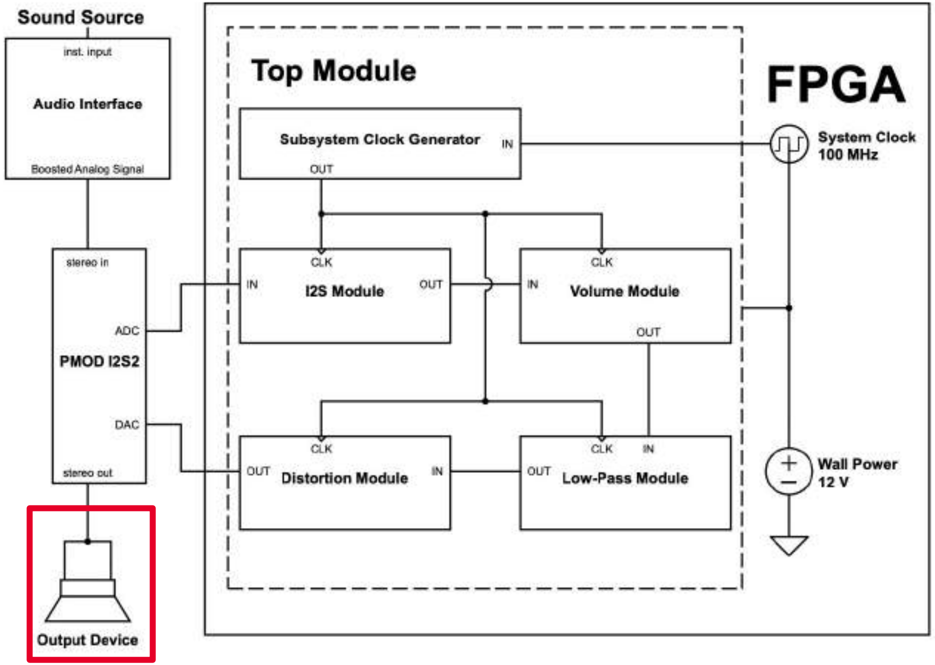
- Audio interface boosts input instrument signal
- PMOD ADC passes in samples of the signal
- I2S frames are processed
- Frames are passed to DAC to form analog wave
- Heard as music from an output device



Harvard John A. Paulson School of Engineering and Applied Sciences

System Diagram

- Audio interface boosts input instrument signal
- PMOD ADC passes in samples of the signal
- I2S frames are processed
- Frames are passed to DAC to form analog wave
- Heard as music from an output device



Build

Define

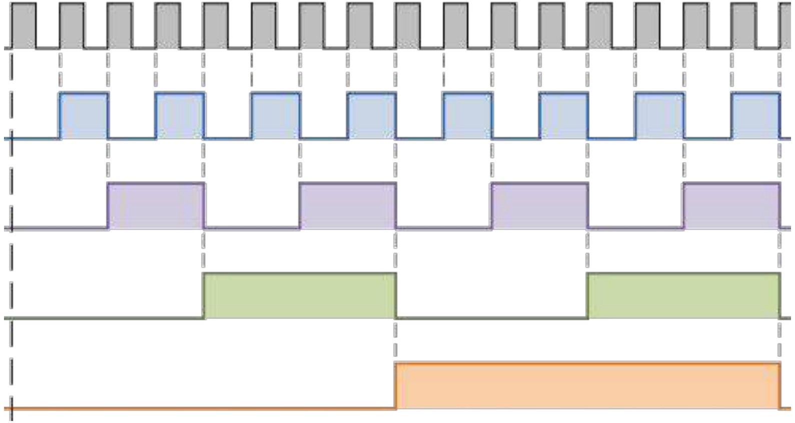
Design

Build

Q&A

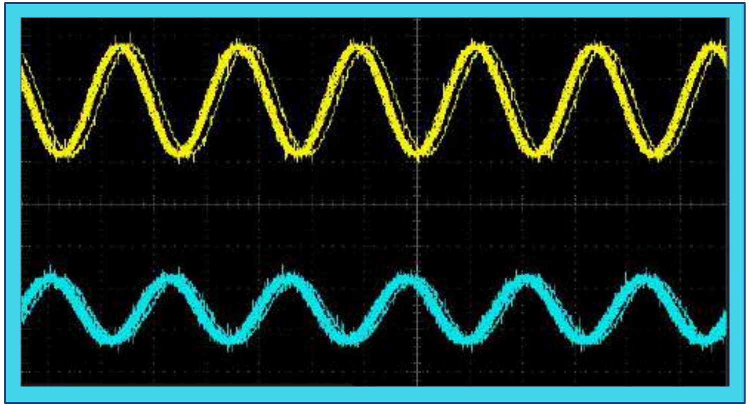
Clock Generation

- The simplest timing solution is to derive all subsystem clocks from the FPGA's 100MHz clock
- This necessitates very specific frequencies be derived from the system clock
- Xilinx provides an intellectual property (IP) block for this express purpose – the clocking wizard
- Configuring the clocking wizard, we can generate a precise enough clock to synchronize the PMOD ADC with the FPGA's internal modules



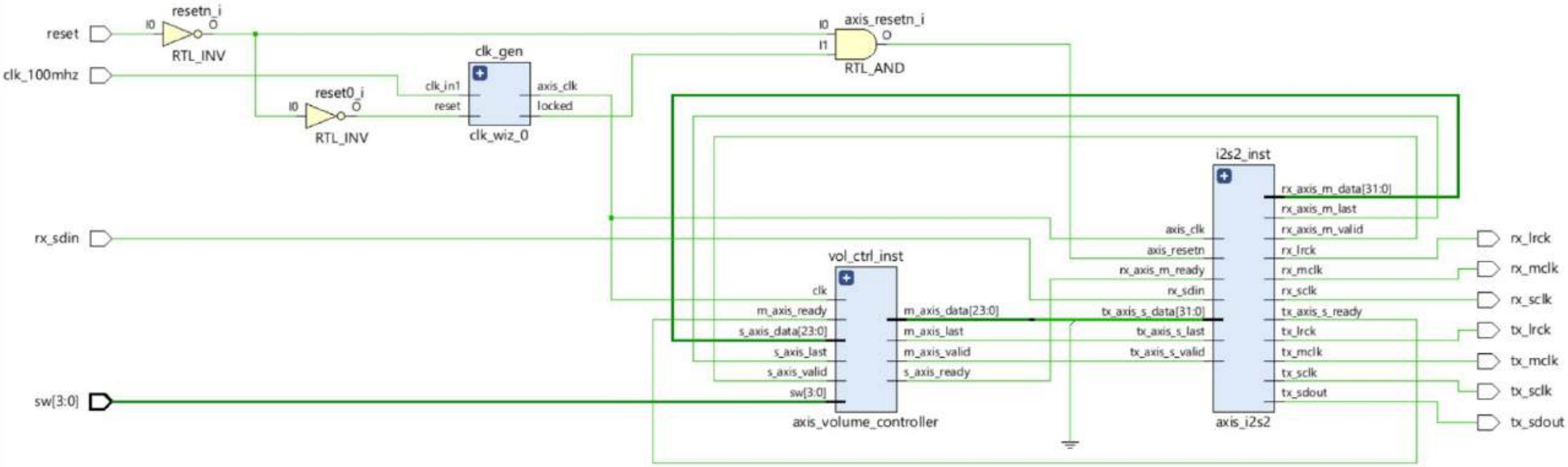
Audio Passthrough (I^{2}S and Volume)

- Designed to take the sound samples from PMOD ADC and deliver them to the DAC
- The clocking wizard IP generates a 22.591 MHz clock from system, which is divided into lrclk (left/right toggle) and sclk (sample clock)
- One sound frame consists of 32 bits per channel, so the bit clock must sample at a rate of 64 times the 44.1 kHz frame rate
- To construct audio frames, the left-right channel toggle inverts every 32 bits



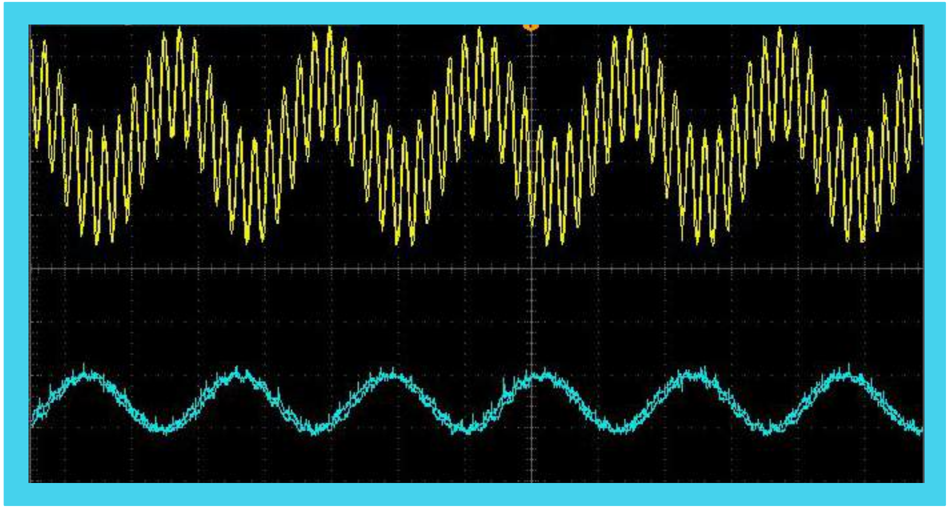
Harvard John A. Paulson School of Engineering and Applied Sciences

Audio Passthrough (Schematic)

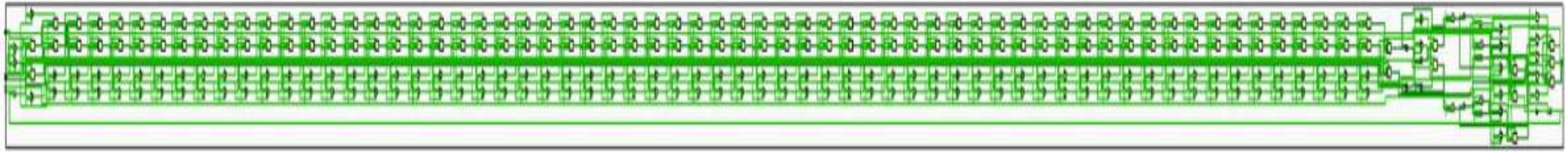


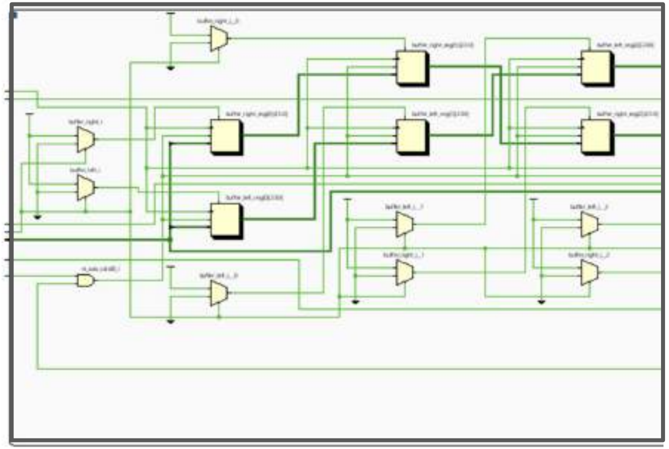
Low-Pass Module

- Because I^{2}S frames represent the amplitude of our digital signal, the we can observe frequency
- Uses AXI-Stream protocol with valid/ready handshaking for input/output
- Modeled as a FSM which maintains a moving window of I^{2}S frames



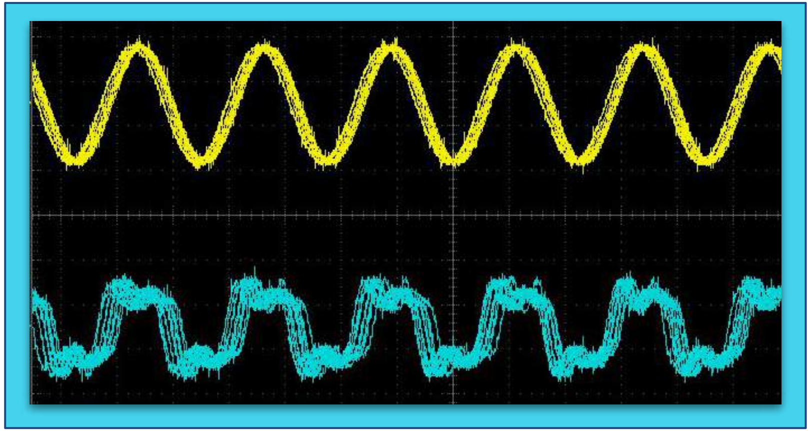
Low-Pass Module (Schematic)





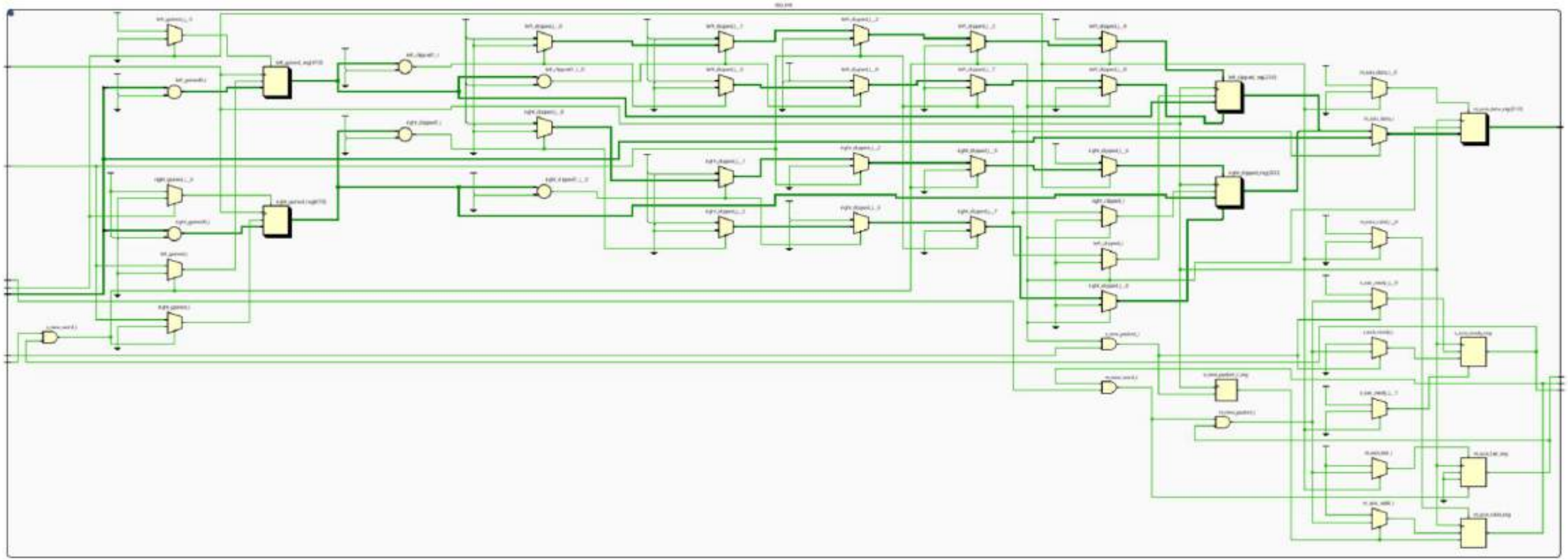
Distortion Module

- Module implementation acts on input frames according to a threshold parameter
- If the recorded amplitude of any sound frame exceeds this threshold, then it is truncated
- This results in flattened wave peaks, which distort the sound at the output



Harvard John A. Paulson School of Engineering and Applied Sciences

Distortion Module (Schematic)



Tech Spec: Latency

- Measured using oscilloscope cursors after probing PMOD input and output
- Measured latency between the peaks of the input and the output
- Several trials run at frequencies of three different orders of magnitude

Frequency	Average Latency
100 Hz	542 μs
1,000 Hz	556 μs
10 kHz	47.2 μs

Table 3: Measured Latency

Tech Spec: Signal-to-noise Ratio

- Signal-to-noise ratio was measured via probing the input and output jack of the PMOD
- Not currently meeting specs, as industry standards hit minimum at 70 dB and a maximum at 110 dB
- Excess noise is likely either in the signal path before entering the FPGA, or due to poor gain staging

$$SNR = 20 \cdot \log_{10}\left(\frac{V_{S+N}}{V_N}\right)$$

Frequency	Average SNR
100 Hz	21 dB
1,000 Hz	35 dB
10 kHz	25 dB

Table 4: Signal-to-noise Measurements

Tech Spec: FPGA Resource Usage

- Resource consumption specifications ensure up to 20 synthesized effects at once
- Resource utilization thus far has not approached the limit
- Implies that by changing the pin constraints on the design, this could be run on cheaper FPGAs

Resource Type	Current Usage	Percentage
Lookup Tables	290 / 53200	0.5%
Block RAM	0 / 140	0.0%
DSP Blocks	0 / 220	0.0%

Table 5: Resource Usage on Zynq7000

References

1. "DAW Latency and CPU." EDMProd. Accessed: Sept. 1, 2024. https://www.edmprod.com/daw-latency-and-cpu/

2. "Inspiring Ways to Use Logic Pro for Live Performance and DJing." LogicXX. Accessed: Sept. 1, 2024. https://logicxx.com/blogs/news/inspiring-ways-to-use-logic-pro-for-live-performance-and-djing

3. "Advice on the Best Setup for Live Performance Using Logic." Logic Pro Help. Accessed: Sept. 1, 2024. https://www.logicprohelp.com/forums/topic/152400-advice-on-the-best-setup-for-live-performance-using-logic/

4. "ASIC vs FPGA: What's the Difference?" ASIC North. Accessed: Sept. 1, 2024.https://www.asicnorth.com/blog/asic-vs-fpga-difference/

References

5. "Synergy Core." Antelope Audio. Accessed: Sept. 1, 2024.https://en.antelopeaudio.com/synergy-core/
6. J. Lipshy. "Development of a Digital Sound Processor Using FPGAs." California Polytechnic State University, San Luis Obispo, 2013. https://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=1326&context=cpesp
7. G. T. Metcalf et al. "Low Power Neural Architecture Search for TinyML Applications." IEEE Journal of Emerging and Selected Topics in Circuits and Systems, vol. 12, no. 2, pp. 255-266, June 2022.
8. S. Wang et al. "High-Fidelity FPGA-Based Emulation of Power Systems with Modular Multilevel Converters." IEEE Access, vol. 9, pp. 100771-100784, 2021.

References

9. R. Y. Litovsky, H. S. Colburn, W. A. Yost, and S. J. Guzman. "The Precedence Effect." Journal of the Acoustical Society of America, vol. 106, no. 4, pp. 1633-1654, 1999.

10. C. Dragoi, C. Anghel, C. Stanciu, and C. Paleologu. "Efficient FPGA Implementation of Classic Audio Effects." Proceedings of the 2021 IEEE International Symposium on Signals, Circuits and Systems (ISSCS), 2021.

Questions?

Define

Design

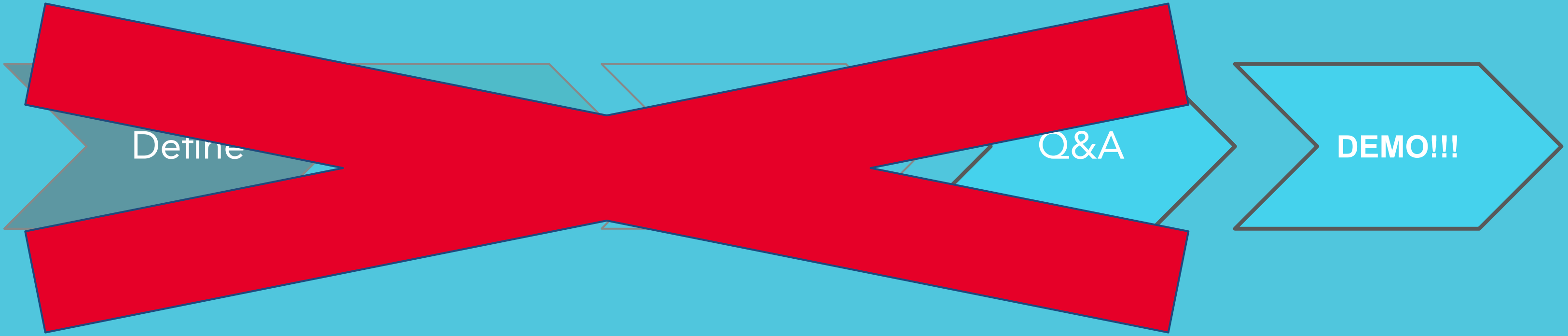
Build

Q&A

My Questions for Further Work

- Currently working on delay effect, which uses a FIFO buffer created from BRAM with depth defined by the desired delay time, it is giving me problems
- The distortion module is functional, but also distorts noise, turning what should be silence into very, very loud buzzing. Any ideas for a possible noise gate solution?
- Buffering with parallel sound effects modules into a signal sum module, or modules in series?

Demo!!!



Define

Q&A

DEMO!!!

Harvard John A. Paulson
School of Engineering
and Applied Sciences

Appendix

High Level Decisions (Justification)

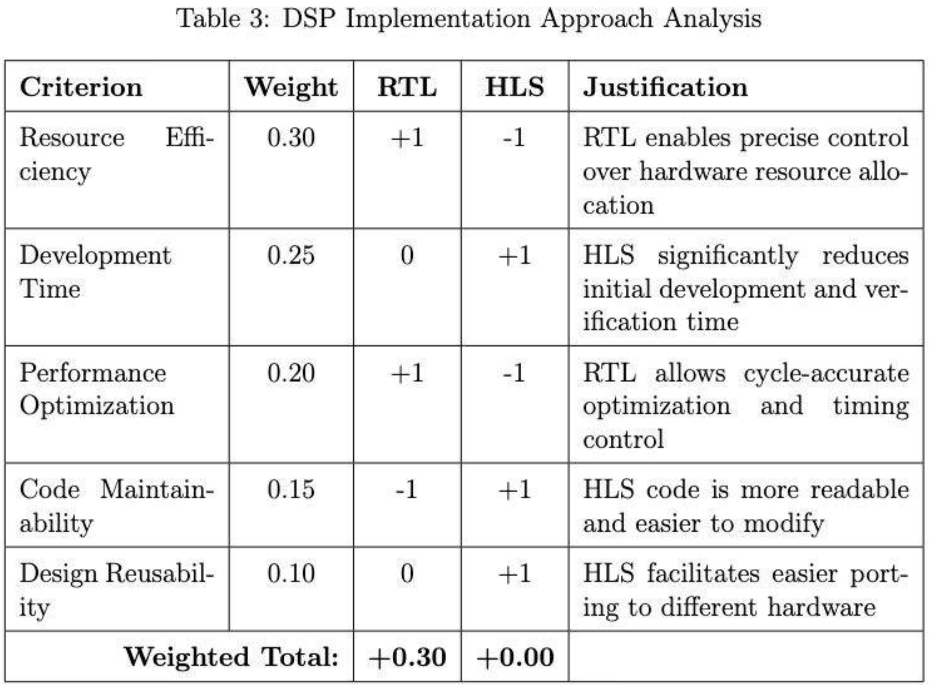
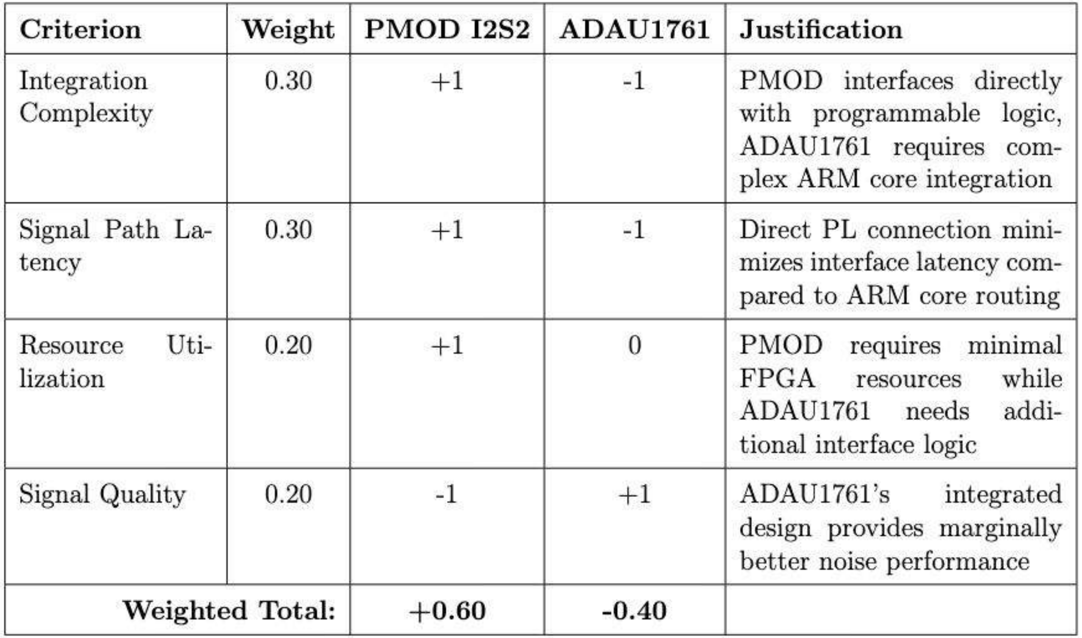


Table 3: DSP Implementation Approach Analysis

Criterion	Weight	RTL	HLS	Justification
Resource Efficiency	0.30	+1	-1	RTL enables precise control over hardware resource allocation
Development Time	0.25	0	+1	HLS significantly reduces initial development and verification time
Performance Optimization	0.20	+1	-1	RTL allows cycle-accurate optimization and timing control
Code Maintainability	0.15	-1	+1	HLS code is more readable and easier to modify
Design Reusability	0.10	0	+1	HLS facilitates easier porting to different hardware
Weighted Total:		**+0.30**	**+0.00**	

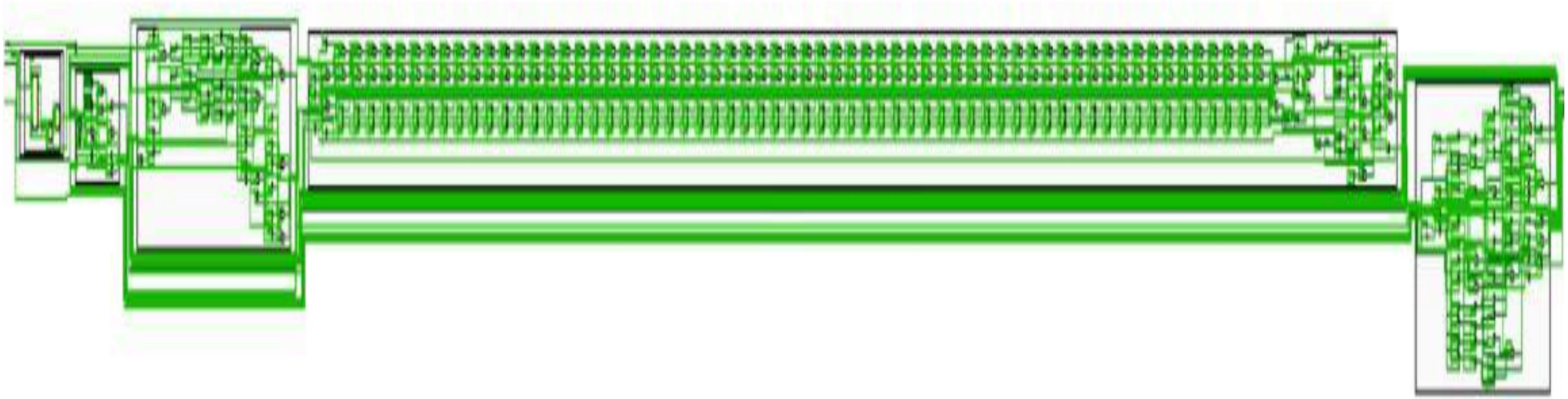
High Level Decisions (Justification)

Table 2: Audio Codec Selection Analysis



Criterion	**Weight**	**PMOD I2S2**	**ADAU1761**	**Justification**
Integration Complexity	0.30	+1	-1	PMOD interfaces directly with programmable logic, ADAU1761 requires complex ARM core integration
Signal Path Latency	0.30	+1	-1	Direct PL connection minimizes interface latency compared to ARM core routing
Resource Utilization	0.20	+1	0	PMOD requires minimal FPGA resources while ADAU1761 needs additional interface logic
Signal Quality	0.20	-1	+1	ADAU1761's integrated design provides marginally better noise performance
Weighted Total:		**+0.60**	**-0.40**	

Full System View



Harvard John A. Paulson
School of Engineering
and Applied Sciences

WHERE
SCIENCE
AND
ENGINEERING
CONVERGE