

Class D Audio Amplifier Design with Power Supply Noise Cancellation

by

Jing Bai

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved November 2015 by the
Graduate Supervisory Committee:

Bertan Bakkaloglu, Chair
Hongjiang Song
Yu Cao

ARIZONA STATE UNIVERSITY

December 2015

ABSTRACT

In this thesis, a digital input class D audio amplifier system which has the ability to reject the power supply noise and nonlinearity of the output stage is presented. The main digital class D feed-forward path is using the fully-digital sigma-delta PWM open-loop topology. Feedback loop is used to suppress the power supply noise and harmonic distortions. The design is using global foundry 0.18 μ m technology.

Based on simulation, the power supply rejection at 200Hz is about -49dB with 81dB dynamic range and -70dB THD+N. The full scale output power can reach as high as 27mW and still keep minimum -68dB THD+N. The system efficiency at full scale is about 82%.

ACKNOWLEDGMENTS

I would like to thank Professor Bertan Bakkaloglu for advising this project. I would also like to thank Professor Yu Cao and Hongjiang Song as committee members for my defense.

TABLE OF CONTENTS

	Page
LIST OF TABLES.....	vi
LIST OF FIGURES.....	vii
CHAPTER	
1 INTRODUCTION	1
1.1 Class D Audio Amplifier Basics	1
1.2 Objectives.....	3
1.3 Outline	4
2 SYSTEM ARCHITECTURE	6
2.1 Digital Class D Amplifier Architecture Review	6
2.1.1 Open-loop Architecture	6
2.1.2 Closed-loop Architectures	10
2.2 PWM Modulation Scheme	13
2.2.1 Fundamental Concepts of Pulse Width Modulation (PWM) ...	13
2.2.2 Signal Nonlinearity and Optimization.....	16
2.3 Power Supply Noise.....	20
2.4 Proposed Error Feedback Closed-loop Architecture	21
3 SYSTEM DESIGN	24
3.1 System Budget	24
3.2 Top-level System Model.....	25
3.3 Digital Class D Modulator.....	26
3.3.1 Delta-sigma Noise-shaper.....	26

CHAPTER	Page
3.3.2 Triangle-wave Generator	32
3.3.3 Comparator.....	33
3.3.4 Simulation Results for Digital Class D Modulator	33
3.4 Continuous-Time Delta-Sigma Analog-to-Digital Converter.....	35
3.5 Noise Budget.....	42
3.5.1 Subtractor-gain Noise	42
3.5.2 Continuous-time Delta-sigma ADC Integrator Noise.....	43
4 DIGITAL IMPLEMENTATION- FPGA BASED EMBEDDED SYSTEM DESIGN	
.....	44
4.1 Atlys Spartan®-6 FPGA Development Kit.....	45
4.2 Embedded System Design	47
4.3 Microblaze Processor Based Embedded System	48
4.4 Microblaze Processor Based Class D System Digital Design.....	50
4.4.1 Digital Class D Modulator HDL Code Generation	50
4.4.2 Embedded System Design Using Xilinx EDK	52
4.4.3 Final Results.....	53
5 ANALOG IMPLEMENTATION - SCHEMATIC DESIGN	56
5.1 Class D Amplifier System Top-level Schematic and Test Bench....	56
5.2 Continuous-time Delta-sigma ADC (CTADC)	57
5.2.1 ADC Top.....	57
5.2.2 Integrator Amplifier	61
5.2.3 Current Steering DAC	68

CHAPTER	Page
5.2.4 2-bit Quanziter	69
5.2.5 Current Bias Generation Block	69
5.3 Switching Power Stage	73
5.4 Subtractor-gain	76
5.5 Speaker Load Model	76
5.6 Miscellanea	78
5.6.1 Delay	78
5.6.2 Output Filter	78
5.6.3 PCM Generation Block	78
5.6.4 Data Write	79
6 FINAL RESULTS	80
REFERENCES.....	85
APPENDIX	
A MATLAB CODE OF THE 3RD ORDER DELTA-SIGMA MODULATOR.....	87
B MATLAB CODE OF THE 4TH ORDER CONTINUOUS-TIME DELTA-SIGMA ADC.....	89
C DIGITAL CLASS D MODULATOR VERILOG CODE GENERATED BY MATLAB.....	92
D PCM GEN VERILOGA CODE.....	100
E DATA WRITE VERILOGA CODE.....	103

LIST OF TABLES

Table	Page
3.1: The System Design Spec Target of the Proposed Class D Audio Amplifier.....	24
3.2: The Specifications of the Desired Delta-Sigma Noise-Shaper.	27
3.3: The Specifications of the Desired Continuous-Time Delta-Sigma ADC.....	35
3.4: Laplace and Z-Transfer Pairs.	37
3.5: Subtractor-Gain Stage Noise Estimation.	43
4.1: Clock Setup for FPGA.	52
5.1: Continuous-Time Delta-Sigma ADC First Integrator Noise Simulation Results. ...	67
6.1: Supply Noise Rejection Over Feedback Delay Time (Delay From Point a to B)...	80
6.2: System Power Without Signal	82
6.3: System Efficiency With 32ohm Load.....	82
6.4: Class D Amplifier System Spec Summary.....	83

LIST OF FIGURES

Figure	Page
1.1: The Basic Principle of a Class D Audio Amplifier.	2
1.2: PWM Waveforms for the Class D Audio Amplifier.	2
1.3: Power Efficiency of Class A, Class B and Class D Amplifiers.	3
2.1: The Block Diagram of a Direct Digital PWM Class D Amplifier.	7
2.2: The Block Diagram of a Pulse-Width-Density (PDM) Class D Amplifier.	8
2.3: $\Delta \Sigma$ Modulator for Class D Amplifier.....	8
2.4: A Simple Diagram of the Delta-Sigma PWM Class D Amplifier.	9
2.5: Basic Diagram of a Closed Loop Digital Input Class D Amplifier With DAC.....	10
2.6: A Fixed-Carrier Based Closed-Loop Second-Order PWM Control Loop.....	11
2.7: Power Supply Sensing Topology.	12
2.8: A Block Diagram of the Digital Audio Amplifier With Error Correction.....	13
2.9: PWM Generation schemes.[5]	15
2.10: Input and Carrier signals.[5].....	16
2.11: Power Spectrum of the Analog PWM Signal	17
2.12: Power Spectrum of the Digital PWM signal.[5].....	18
2.13: The Power Spectrum of Symmetrical Regular Sampled PWM.[5]	19
2.14: The Spectrum of the Asymmetrical Regular Sampled PWM signal.[5].....	20
2.15: Frequency Mixing of the Noise With input.[5]	21
2.16: Time Domain Error Signal.	22
2.17: Block Diagram of the Proposed Closed Loop Digital Class D Amplifier.....	23
3.1: MATLAB Simulink Top-Level Model of the Class D Amplifier System.	25

Figure	Page
3.2: MATLAB Simulink Sub-Level Model of Digital Class D Modulator.	26
3.3: A General Diagram of the CIFB Architecture.	27
3.4: The Pole-Zero Plot of the NTF in Z Domain.	29
3.5: The Frequency Response of the NTF.	29
3.6: The Calculated SNR for the NTF With 6-Bit Quantizer.	30
3.7: The MATLAB Simulink Model for the Delta- Sigma Modulator.	32
3.8: The MATLAB Simulink Model for the Triangle-Wave Generator.	32
3.9: The MATLAB Simulink Model for the Comparator.	33
3.10: Frequency Spectrum of PWM Output of Digital Class D Modulator.	34
3.11: Zoom in of the Frequency Spectrum of the PWM Output.	34
3.12: A Diagram of the CRFB Architecture Used in the Design.	36
3.13: The Pole-Zero Plot of the NTF in Z Domain.	38
3.14: The Frequency Response of the NTF.	39
3.15: MATLAB Simulink Model for Continuous-Time Delta-Sigma ADC.	40
3.16: The Power Spectrum of the ADC Output With 5KHz Sine-Wave Input in Audio Band.	40
3.17: SNR vs. Input Amplitude for the CTADC.	41
4.1: The Atlys Spartan 6 FPGA Development board.[20].	45
4.2: Xilinx Spartan-6 Atlys Development Board With Its peripherals.[20].	46
4.3: The Embedded System Design Flow.	47
4.4: The Microblaze Processor Based Embedded System and Peripherals.	49
4.5: The MATLAB Simulink Model Used for the Embedded Design.	50

Figure	Page
4.6: The MATLAB Model of the Decimator.	51
4.7: 3rd Order Sinc Filter With Decimation Factor of 16.....	51
4.8: 3rd Order Sinc Filter With Decimation Factor of 8.....	51
4.9: The Power Spectrum of the PWM Output After Decimator With MATLAB Simulation.....	54
4.10: The Power Spectrum of the PWM Output of HDL Code.	54
4.11: The Power Spectrum of the PWM Output After Decimator of FPGA.	55
5.1: Class D Amplifier System Top-Level Schematic With Test Bench.	59
5.2: Continuous-Time Delta-Sigma ADC Top Schematic.	60
5.3: Integrator Amplifier Top Schematic.	62
5.4: Folded-Cascode Class AB Amplifier Schematic.....	62
5.5: CMFB Amplifier Schematic.	63
5.6: Amplifier Bias Schematic.....	63
5.7: The Open Loop Response of the Integrator Amplifier.....	65
5.8: The Closed-Loop Response of the Integrator.	66
5.9: Current Steering DAC Schematic.	68
5.10: 2bit Quantizer Top-Level Schematic.	70
5.11: Latch Comparator Schematic.	71
5.12: Clock Gen Schematic.	71
5.13: Decoder Schematic.	72
5.14: Bias Gen Schematic.....	72
5.15: The Switching Power Stage Top Schematic.....	74

Figure	Page
5.15: The Switching Power Stage Top Schematic.....	74
5.16: The PWM Dead-Time Gen Schematic.	74
5.17: PMOS Gate Pre-Driver Schematic.....	75
5.18: NMOS Gate Pre-Driver Schematic.....	75
5.19: Subtractor and Gain Schematic.	77
5.20: Load Model Schematic.....	77
5.21: Delay Schematic.	78
5.22: The PWM Output Filter Schematic.....	79
6.1: Power Supply Rejection vs Supply Noise Frequency	80
6.2: Class D Amplifier System Showing Feedback Delay Point a to B.....	81
6.3: System THD+H vs Input Amplitude (V_{peak})	81

CHAPTER 1

INTRODUCTION

1.1 Class D Audio Amplifier Basics

The class D amplifier is also called switching amplifier. This type of amplifier only operates on “on” and “off” states, so, ideally, the power loss is very small.[1] Compared to class A, class B and other linear amplifiers, class D amplifier is usually used in the audio system because of its high efficiency. Based on the different type of input signal, class D audio amplifier can be digital input or analog input. In this thesis, the audio system is based on the digital input type of class D audio amplifier. For simplicity, in this part, the analog input type class D amplifier is used for background explanation.

The class D audio amplifier working scheme is very similar to the switching power supply, such as buck/boost DC to DC convertor. Fig 1-1 shows the basic principle of a class D audio amplifier.[2] The system consists of a PWM (pulse width modulation) modulator which is used to generate a switching control signal and a class D output driver which is a power stage. The PWM modulator will generate a high frequency triangle or a saw-tooth waveform, usually 500KHz to 1MHz. This signal is compared with the audio input signal with a frequency ranging from 20Hz to 20kHz typically. Fig 1-2 shows the waveforms of the pulse width modulation scheme.[2] The output of the comparator is called PWM signal output. This PWM signal is then used to drive the class D output driver creating an amplified PWM signal. With the class D output driver, the class D audio amplifier can deliver large power to the load. Finally, a low pass filter is applied to the output to filter out the out of audio band.[1]

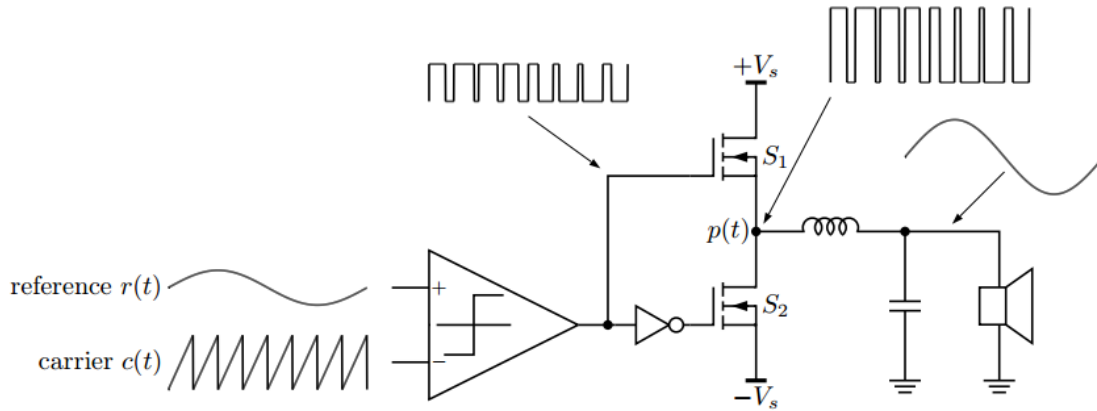


Figure 1.1: The basic principle of a class D audio amplifier.

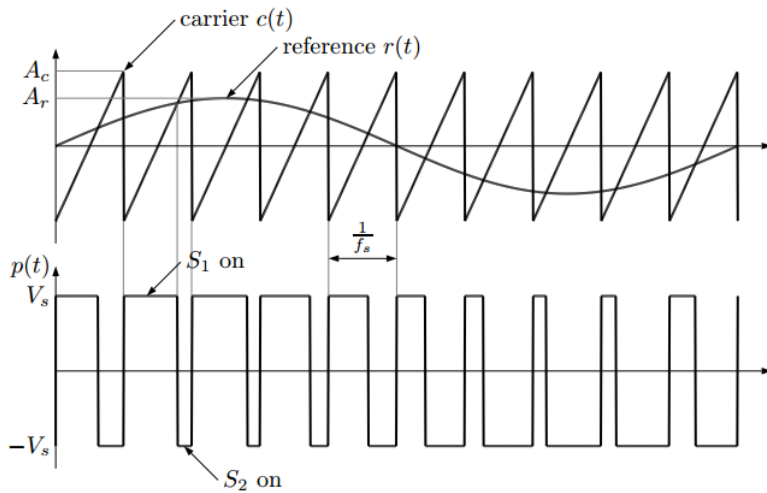


Figure 1.2: PWM waveforms for the class D audio amplifier.

Class D amplifier has many advantages, such as high efficiency, smaller dimension and higher output power. Fig 1-3 demonstrates the efficiency comparison between class A, class B and class D amplifiers.[3] In this plot, AD199x chips are used to compare with ideal class A and class B amplifiers. However, it also has some drawbacks:

1. The distortion of the output signal. The nonlinearity is mainly caused by dead-time, limited on resistance, limited slew-rate and other non-ideal switching behaviors of the class D driver stage.
2. No power supply noise rejection. The power supply noise on the class D driver stage is directly modulated with the PWM signal and appears at the driver output.
3. Electromagnetic interference (EMI) noise. The radiated EMI is mainly caused by the high switching rate of the class D driver stage. This rate is defined by the PWM carrier frequency.

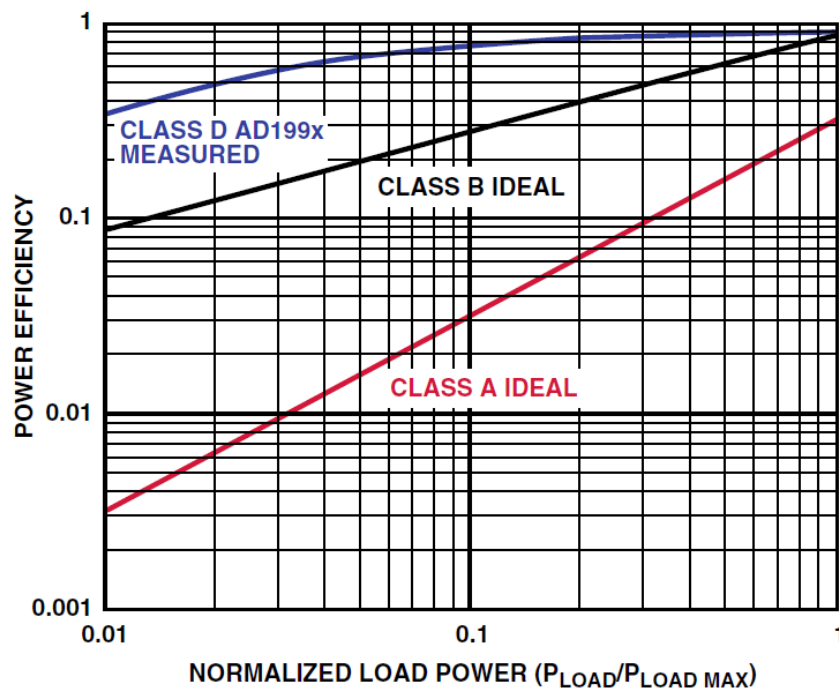


Figure 1.3: Power efficiency of Class A, Class B and Class D amplifiers.

1.2 Objectives

The objective of this thesis is to design a digital input class D audio system which has the ability to reject the power supply noise and nonlinearity of the output stage. The analog portion design is using global foundry 0.18 μ m technology. The digital portion design is RTL code only and verified in the FPGA. Top-level digital-analog-mixed-signal simulation is needed to verify the performance of the whole system.

1. Better power supply noise rejection (PSR) in the audio band compared to open-loop class D audio system.
2. Better total harmonic distortion and noise (THD+N) in the audio band compared to open-loop class D audio system.
3. The class D audio system should have $>70\text{dB}$ THD+N and $>80\text{dB}$ dynamic range at typical corner.
4. Better efficiency than class A, class AB amplifier. $>80\%$ total efficiency at full scale.
5. Load condition: 32ohm load headphone or earbud.

1.3 Outline

Chapter 2 covers the system architecture of the proposed class D audio amplifier. This portion reviews power supply rejection scheme including open-loop and close-loop topology. Then, the proposed architecture is described. Special PWM modulation scheme is also used to reduce the EMI noise.

Chapter 3 talks about the system design of the proposed class D audio amplifier. Matlab simulink is used to model the system level class D amplifier. Nonlinearity and

other system concern, such as analog delay, are considered in the system model. This portion also covers the noise budget of the whole class D system.

Chapter 4 discusses the digital implementation and verification of the digital design. Atlys Spartan®-6 FPGA Development Kit is used for a listening demo.

Chapter 5 covers the analog design portion which includes a continuous-time sigma-delta analog-to-digital convertor (CTSDADC), class D output driver stage, subtractor-gain stage and other miscellaneous blocks.

Chapter 6 shows the system simulation results and spec table.

CHAPTER 2

SYSTEM ARCHITECTURE

2.1 Digital Class D Amplifier Architecture Review

2.1.1 Open-loop Architecture

A typical digital input class D amplifier includes the open-loop topology which uses a digital PWM (DPWM) modulator to drive a switching output stage without any feedback. Open-loop solution allows fully-digital implementations, which can be manufactured by the most advanced technology and give the advantage in the cost and size.

1. Direct digital PWM class D amplifier

Fully digital open-loop class D topology can be separated to direct digital PWM conversion and sigma-delta PWM conversion. The direct digital PWM conversion approach usually involves a sampling step and a PWM conversion step which is also called uniformly-sampled pulse-width-modulation (UPWM). Fig 2-1 shows a block diagram of a direct digital PWM class D amplifier.[4] However, because of the sampling and conversion, the digital PWM modulator will introduce harmonic distortions in many modulation schemes.[5] Different modulation approaches will be discussed in later part of this chapter. In addition, direct digital PWM conversion usually requires high speed for digital DSP core. For example, for a typical 16bit 44.1kHz PCM audio input, to implement a saw-tooth comparison, the digital core need to run at 2.89GHz ($=216 \times 44.1\text{kHz}$). For a triangle comparison, the digital core need to run at 5.78GHz ($=2 \times 216 \times 44.1\text{kHz}$). PWM switching frequency also produces EMI.

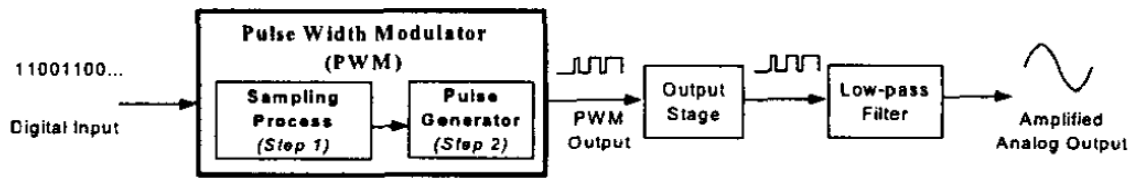


Figure 2.1: The block diagram of a direct digital PWM class D amplifier.

2. Pulse-width-density (PDM) class D amplifier

Instead of PWM controller, pulse density modulation (PDM) solution is used in many places to solve the distortion and EMI issues.[6] The PDM modulator usually includes an interpolation oversampling and delta-sigma modulation. Fig 2-2 shows a block diagram of a PDM class D amplifier.[7] The modulator is a 1-bit sigma-delta modulator generating a PDM signal driving a switching output stage. However, this architecture has the trade-off between power efficiency and SNR. To achieve sufficient audio band SNR, the oversampling ratio need to be at least 64. With high oversampling ratio, switching frequency of the output stage will be also high which cause low power efficiency. For example, with oversampling ratio of 64, the switching frequency at output will be $44.1\text{kHz} \times 64 = 2.822\text{MHz}$. To overcome this disadvantage, high order modulator is used to lower the oversampling ratio. Fig2-3 shows a sigma-delta PWM with a 1-bit seventh-order feedforward architecture.[7] However, the high order 1-bit sigma-delta will have stability issue and complicated implementation.

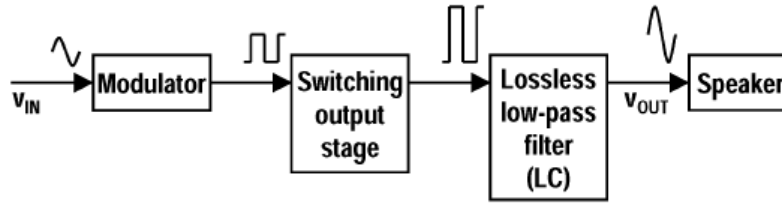


Figure 2.2: The block diagram of a Pulse-width-density (PDM) class D amplifier.

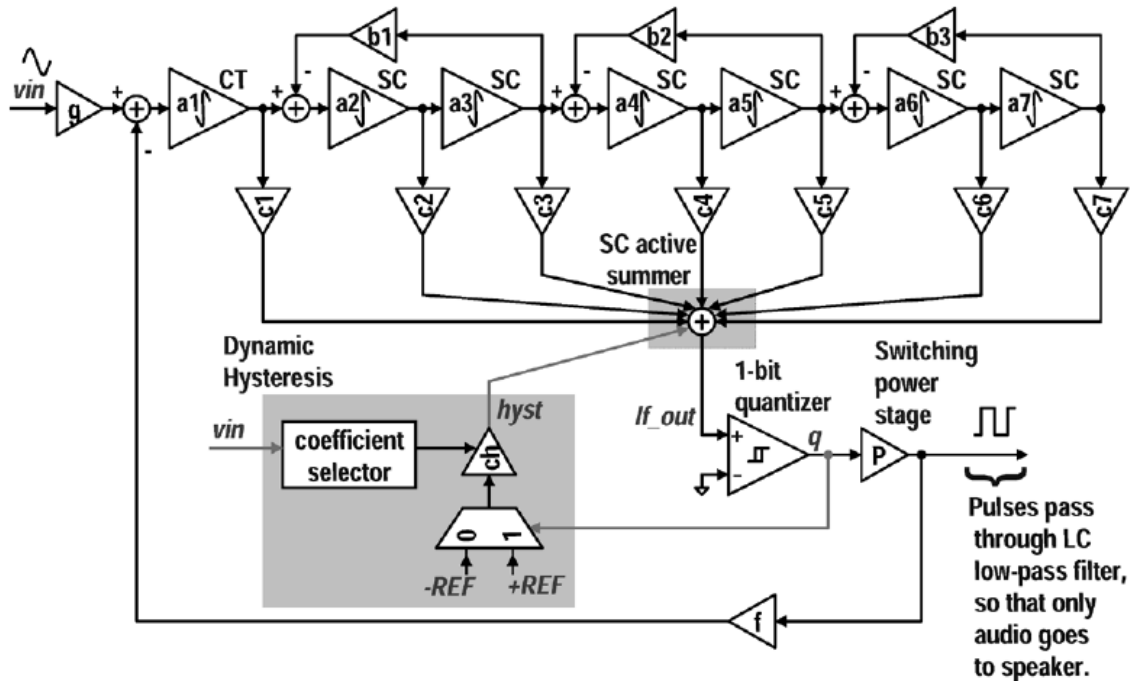


Figure 2.3: $\Delta\Sigma$ modulator for class D amplifier.

3. Delta-sigma PWM class D amplifier

Combining both PWM and PDM solution, sigma-Delta PWM conversion approach is introduced to overcome the speed and power efficiency issue. Sigma-delta PWM solution involves an oversampling block, multi-bit sigma-delta modulator, a PWM modulator and a switching output stage.[8] Fig2-4 indicates a simple diagram of the sigma-delta PWM class D amplifier.

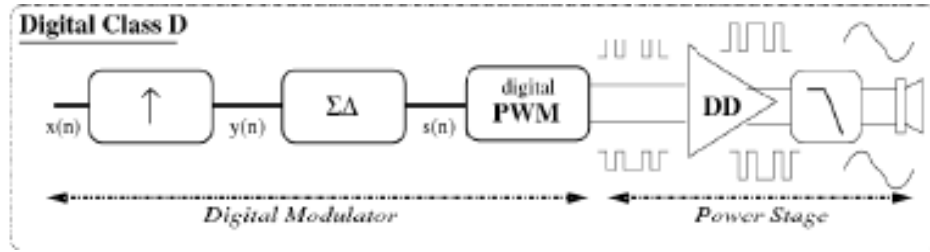


Figure 2.4: A simple diagram of the delta-sigma PWM class D amplifier.

Compared to the same order 1-bit sigma-delta modulator, the multi-bit sigma-delta modulator gives higher SNR, lower oversampling ratio and more stability. Lower number of bits compared to PCM signal also leads to lower PWM computation speed in the digital PWM block. For example, to achieve the same performance, an 8-bit sigma-delta with oversampling rate (OSR) of 8 is used. Then, the DPWM speed will be 90.3MHz ($=28 \times 44.1\text{kHz} \times 8$) for a saw-tooth computation and 180.6MHz ($=2 \times 28 \times 44.1\text{kHz} \times 8$) for a triangle computation. Digital PWM modulator moves the switching frequency down from a few MHz to within 1MHz.

Although the full-digital solution is attractive, the open-loop topology has some drawbacks:

1. Total harmonic distortion (THD) is worse because of the modulation method and nonlinearity of the output stage.
2. Power supply rejection ratio (PSRR) is poor because of the nonregulated supply at the output stage.

2.1.2 Closed-loop Architectures

An easy way to improve THD and PSRR is to apply feedback to suppress the distortions. So, closed loop architecture is developed to implement in the digital class D amplifier.[9][10]

1. DAC based analog closed-loop class D amplifier

Closed-loop architecture is widely used to suppress and compensate the harmonic distortion and power supply noise in the analog input class D amplifier. To make use of the advantage of the analog closed-loop topology in the digital input case, a high resolution DAC is commonly added in front of the analog class D amplifier. Fig 2-5 shows a basic diagram of a closed loop digital input class D amplifier with DAC. [9]

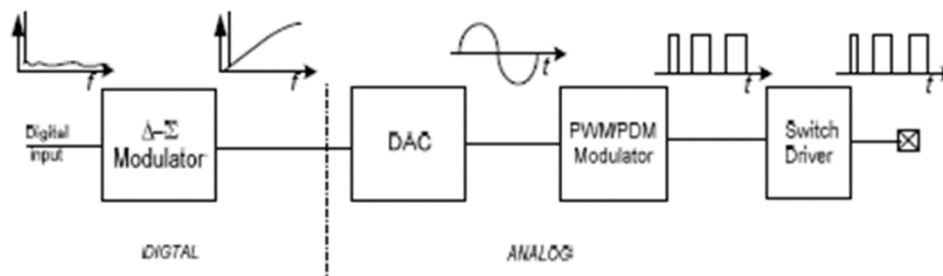


Figure 2.5: a basic diagram of a closed loop digital input class D amplifier with DAC.

The analog class D amplifier can be delay based, hysteretic based or fixed carrier based closed loop architecture.[8] Fig 2-6 shows a fixed-carrier based closed-loop second-order PWM control loop.[12] The controller consists of two gm-C integrators which realize a second-order transfer function and a comparator which drives the output

stage. A few hundred kHz triangle reference is served as a reference before the comparator to generate the PWM signal. The controller drives a class D output driver stage which can amplify the PWM and provide high power. The class D output is fed back to the input of the controller.

The DAC based architecture is widely used in several industry products because of the easy configuration, high reliability and good THD and PSRR. However, this architecture requires a high performance DAC which consumes high power and large area. It will not suppress the THD produced by inherent PWM nonlinearities.

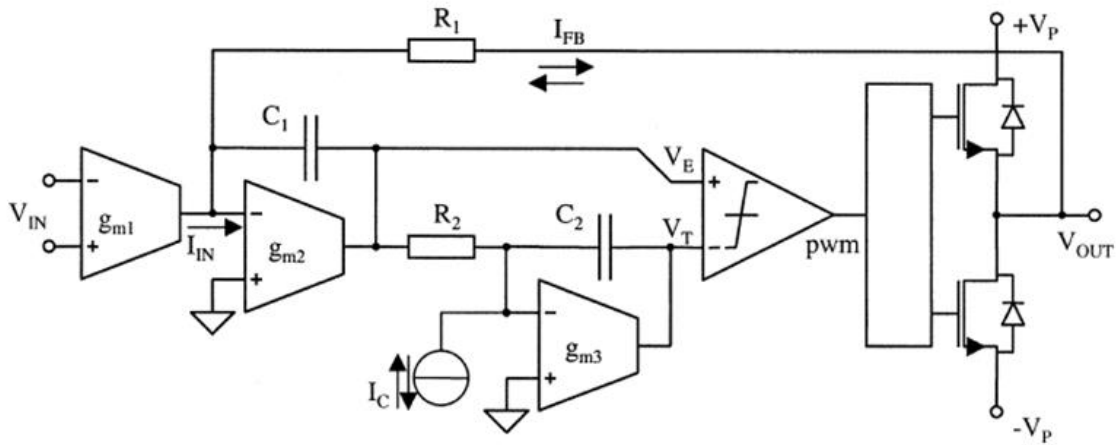


Figure 2.6: a fixed-carrier based closed-loop second-order PWM control loop.

2. Power supply sensing topology

To eliminate the high-performance DAC, some attempts have been made to improve power supply rejection by sensing the supply voltage and using the feedback to adjust digital modulator, as proposed in [11]. As it shown in Fig 2-7, two ADCs are used to sense the positive and negative supply of the switching output stage. The noise

information is feedback to the noise shaper, and the corrected digital PWM signal is finally driving the switching output stage. The PSRR of the system is highly depending on the ADC resolution, loop delay and noise shaper feedback coefficient.

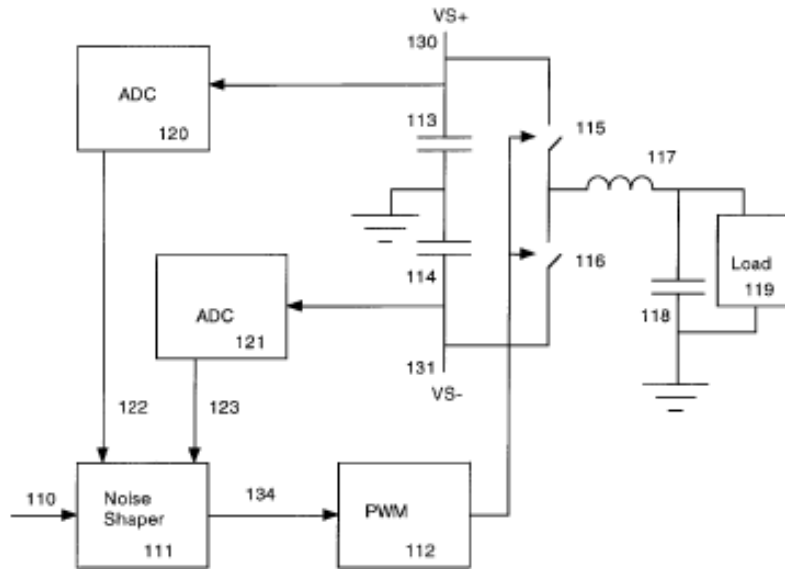


Figure 2.7: Power supply sensing topology.

The power supply sensing topology will give very good PSRR, because it corrects the supply noise directly. However, this architecture alone will not suppress the THD produced by digital PWM modulation scheme and power-stage non-idealities.

3. Error correction topology

Fig. 2-7 shows a block diagram of the digital audio amplifier with error correction.¹² A corrected PWM signal which drives the power stage is generated using the difference between the digital PWM signal and the output signal. The difference represents the error which is amplified using an integrating error amplifier. A small (6–7 bits) flash ADC is used to convert this error signal to the digital domain. Both the rising

and falling edges of the PWM signal will be corrected based on the output of the ADC. The corrected PWM signal is then fed to the switching output stage.

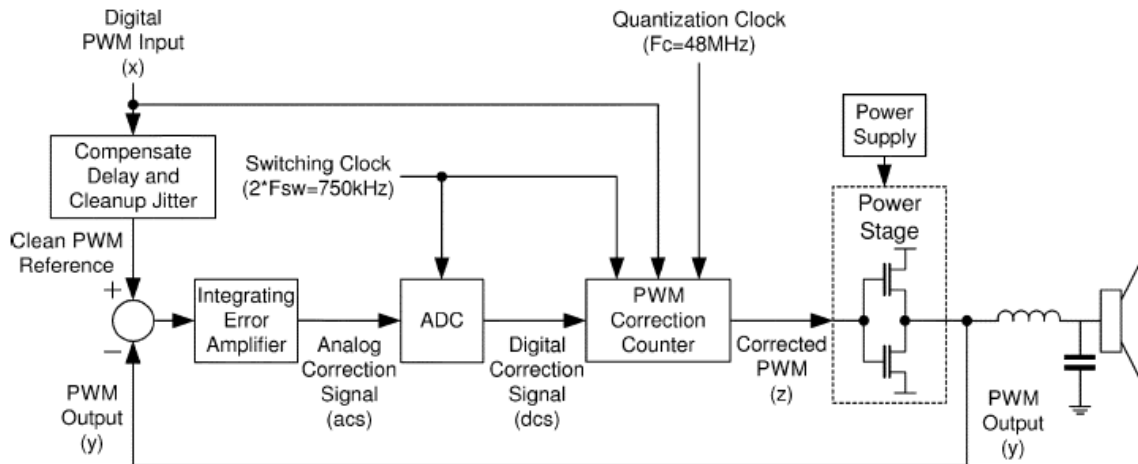


Figure 2.8: a block diagram of the digital audio amplifier with error correction.

The error correction topology is based on a good PCM to PWM Conversion to reduce the distortion caused by the digital PWM modulation. A sufficiently fast and high resolution ADC is needed such that the corrected PWM signal can accurately propagate to the input of the switching output stage before the next PWM edge.

2.2 PWM Modulation Scheme

2.2.1 Fundamental Concepts of Pulse Width Modulation (PWM)

In principle, PWM modulation schemes is used to create a series of pulses with different width which in average or after filtering have the same fundamental frequency

behavior as the target reference waveform in the frequency band we care about. The main issue of the PWM signal is that it contains undesired harmonic tones which should be suppressed. So, the primary aim of any PWM scheme is to calculate the converter switch high/low width within a fixed period of time which creates the desired voltage. The second aim of the PWM scheme is to minimize the undesired harmonic distortion, switching losses when determine the switching signal.

There are several kinds of PWM modulation schemes:[4]

1. Naturally sampled PWM: Switching at the intersection of a high frequency carrier and a target input waveform.
2. Regular sampled PWM: Switching at the intersection of a high-frequency carrier and a sampled input waveform.
3. Direct PWM: Switching so that the integrated area of the target input waveform over the PWM period is the same as the integrated area of the switched output after conversion.

The first modulation scheme is used in analog input class D amplifiers. As it shown in fig 2-8, the target analog input waveform is compared with the carrier in continuous time and switching signal feeds to the switching output stage directly.

The major limitation of the naturally sampled PWM is that it cannot be implemented in a digital modulation scheme, because the digital modulator input is sampled at the beginning of each PWM period and the real intersection of the digital input and the carrier needs complex algorithms and calculation. Hence, for the digital input class D amplifier, the regular sampled PWM scheme is widely used to overcome the limitation of the modulation schemes. As it shown in fig2-8, the carrier can be triangle carrier or saw-tooth carrier. For triangle carrier, comparison is happening at both

falling and rising edge of the carrier, so both edges are moving in each PWM period. While for saw-tooth carrier, comparison occurs at only rising edge of the carrier and reset at the falling edge, hence only one edge is moving and the other edge is always reset at the end of the PWM cycle.

For triangle carrier, sampling can be symmetrical, where the input is sampled at the beginning of each PWM cycle and held constant for the entire cycle, or asymmetrical, where the input is sampled twice at the beginning of rising or falling of the carrier. Fig 2-8 shows the symmetrical and asymmetrical sampling for triangle carrier.

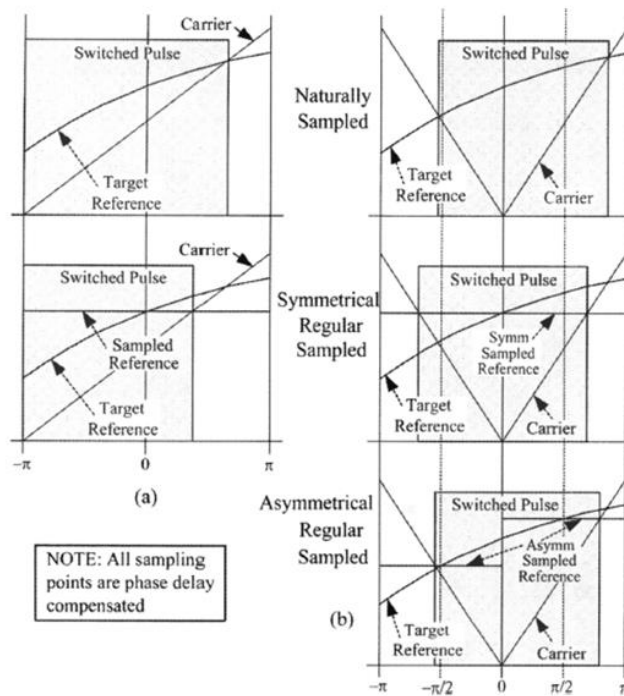


Figure 2.9: PWM generation schemes.[5]

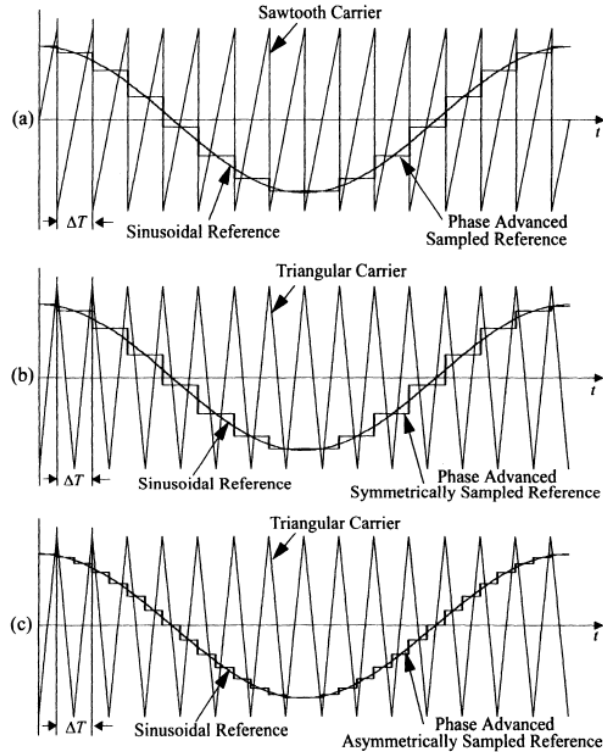


Figure 2.10: input and carrier signals.[5]

2.2.2 Signal Nonlinearity and Optimization

As mentioned before, in both analog and digital implementation, PWM signal has harmonics and nonlinearity.

1. Analog PWM signal

For analog PWM, or naturally sampled PWM, there is no harmonic distortion in the signal band, but inter-modulation component around carrier frequency. Figure 2-11(a) shows the power spectrum of the analog PWM signal.

$$x(t) = M \sin(\omega_s t) + \sum_{m=1}^{\infty} \frac{4}{m\pi} \sin(m\pi\gamma) \cos(m\omega_c t)$$

Equation 2-1

$$\gamma = (1 + M \sin(\omega_s t)) / 2$$

Equation 2-2

γ : Duty cycle

ω_c : Carrier frequency

ω_s : Input signal frequency

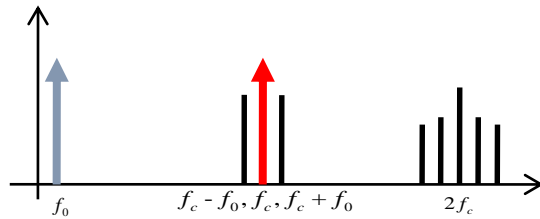
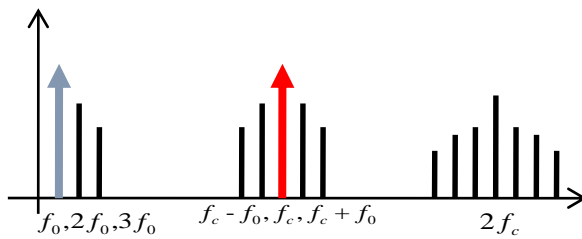


Figure 2.11: a. Power spectrum of the analog PWM signal



b. Power spectrum of the digital PWM signal

2. Digital PWM signal

For digital PWM signal, there are signal harmonics caused by sampling and more inter-modulation distortion (IMD) component at higher frequency. Figure 2-10 shows the power spectrum of the digital PWM signal. From the diagram, beside the inter-modulation component around the carrier frequency, harmonic distortion also shows in the signal band. The in-band distortion is the component that needs to be suppressed.

a. Saw-tooth carrier regular sampled PWM.

Fig 2-11(a) shows the power spectrum of a trailing edge saw-tooth carrier regular sampled PWM. Compared this plot with the power spectrum of the naturally sampled

PWM shown in Fig 2-9, Fig 2-11 shows more in-band harmonic distortions as while as distortions around the carrier frequency.

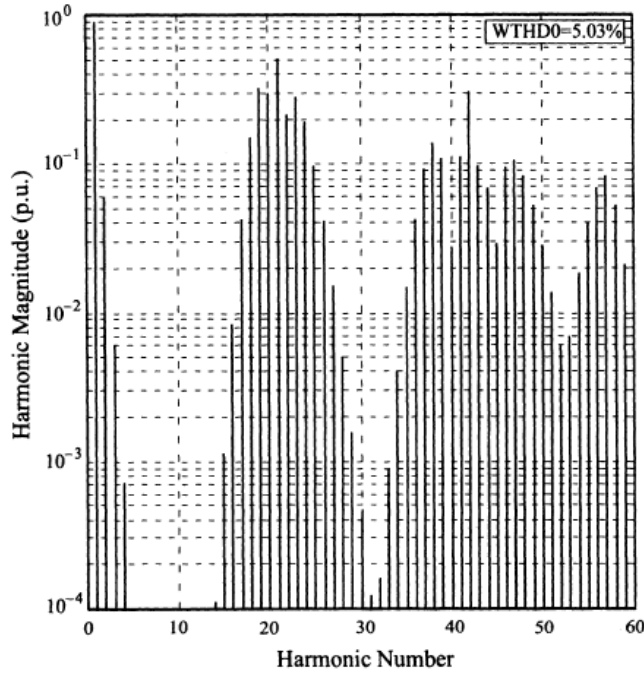


Figure 2.12: Power spectrum of the digital PWM signal.[5]

b. Symmetrical regular sampled PWM

With a triangular carrier, a similar approach can be adopted for symmetrical regular sampled PWM. Fig 2-12 shows the power spectrum of symmetrical regular sampled PWM. Compared to both naturally sampled PWM and saw-tooth regular sampled PWM, the power spectrum of symmetrical regular sampled PWM has significant differences. In the baseband, the harmonic distortion still exist, but the magnitude of harmonic components produced by the symmetrical regular sampling process has rolled off much more rapidly than the saw-tooth regular sampling. In the carrier band, the odd

sideband harmonics around the odd carrier multiples and the even sideband harmonics around the even carrier multiples are significantly reduced.

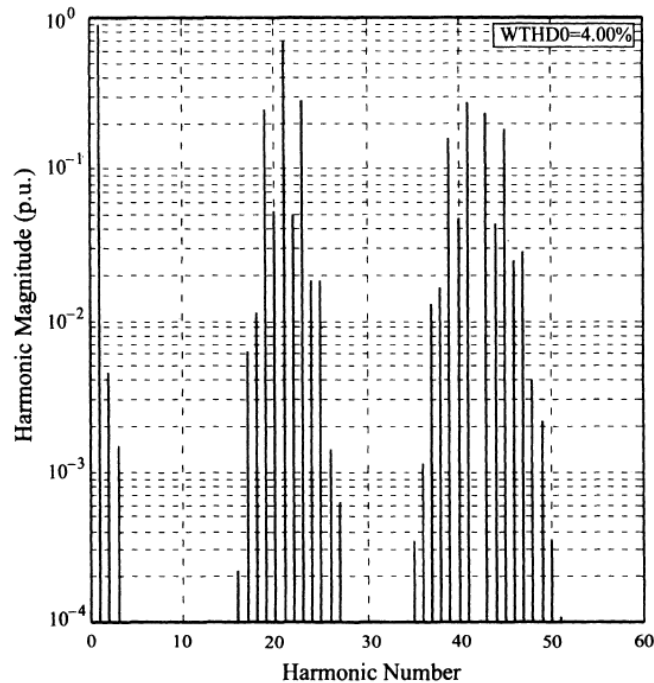


Figure 2.13: the power spectrum of symmetrical regular sampled PWM.[5]

c. Asymmetrical regular sampled PWM

Asymmetrical sampling method is used to optimize the nonlinearity. The regular sampling method is using a sawtooth carrier. The harmonics of the sawtooth carrier is affected by the carrier ratio. The asymmetrical sampling method is using a triangular carrier. Figure 2-13 shows the frequency domain spectrum of the asymmetrical regular sampled PWM signal. Compared to the sampling scheme mentioned above, some harmonics are completely eliminated by this method. In the base band, the even order harmonics is eliminated. In the carrier band, the odd harmonic sideband components

around odd carrier frequency multiples, and even harmonic sideband components around even carrier frequency multiples, are completely eliminated.

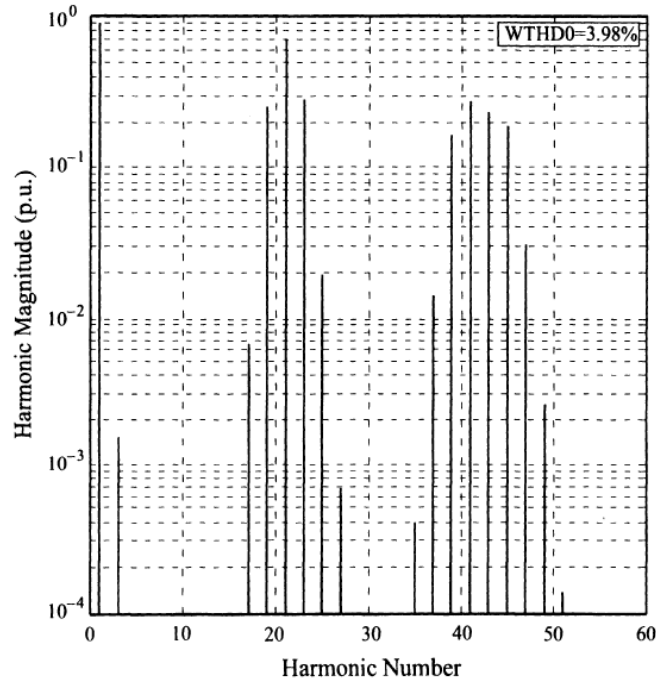


Figure 2.14: the frequency domain spectrum of the asymmetrical regular sampled PWM signal.[5]

2.3 Power Supply Noise

As mentioned in the first section of this chapter, the open loop topology is vulnerable to the power supply noise because any noise on the power stage will be directly mixed to the output. The following equation 2-4 shows all the nonlinear parts caused by the power supply noise.

$$x(t) = M \sin(\omega_s t) + N \sin(\omega_n t) + \frac{MN}{2} \cos(\omega_s \pm \omega_n)t + \varepsilon \quad \text{Equation 2-3}$$

ε includes the negligible DC tone, higher order harmonics.

ω_s is the input signal frequency.

ω_n is the power supply noise frequency.

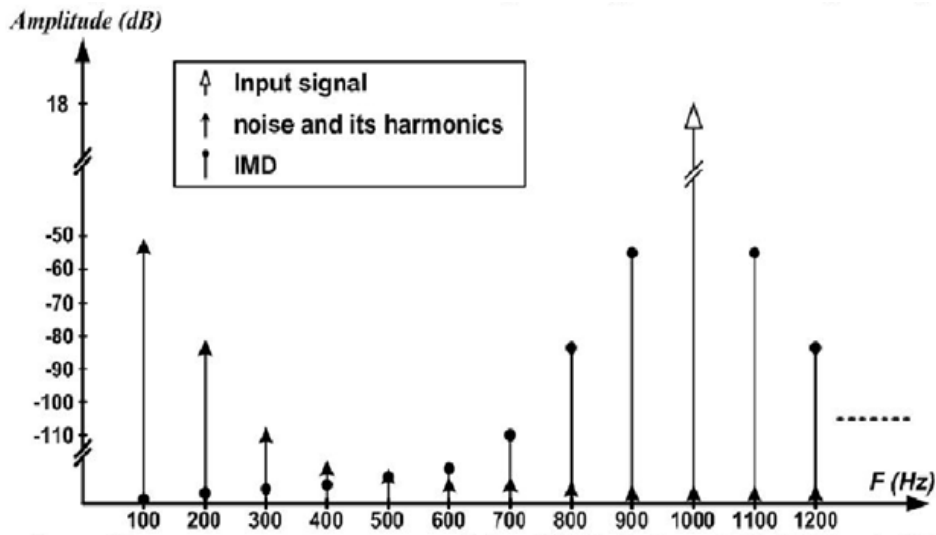


Figure 2.15: frequency mixing of the noise with input.[5]

2.4 Proposed Error Feedback Closed-loop Architecture

In this thesis, a truly digital input class D amplifier is proposed which eliminates the high performance DAC. The main digital class D feed-forward path is using the fully-digital sigma-delta PWM open-loop topology. By using this topology, it will reduce:

1. The design complexation of the sigma-delta modulator,
2. The speed of the digital PWM modulator and
3. The switching loss of the switching output stage.

Feedback loop is used to suppress the power supply noise and harmonic distortions. However, instead of closing the loop at the output of digital PWM modulator, this thesis feeds the output directly to the PCM input which not only suppresses the distortion in the analog portion, but also compensates the harmonics caused by the digital PWM modulator.

Fig 2-17 shows the block diagram of the proposed closed loop digital class D amplifier. The noise shaper is a 3rd order sigma-delta digital modulator with 6 bits quantizer and $OSR = 32$. CIFB architecture is used to achieve good performance. Digital PWM modulator is using asymmetrical sampling process with a triangular carrier to reduce the number of harmonic distortions caused by the modulation scheme. [4] The close loop class D architecture is normally used to reduce the power supply noise of the power stage. The error feedback scheme is used in the proposed architecture. The error signal is shown in fig2-16. The error signal is a switching signal with the same frequency and duty cycle as the switching output and the amplitude of power supply noise. The error signal cannot be feedback to input directly because the high frequency components in the switching signal will fold back to audio band during sampling. The most widely used method is adding an antialiasing filter in front of an ADC.[15] However, the antialiasing filter will add delay to the system, which will degrade the PSR performance. It will also add complexity in the design. Continuous-time ADC has the inherent filtering effect because of the integrator and R C filter. The error signal is fed into a 4th order continuous-time sigma-delta ADC. The 2bit output is directly feedback to the upsampled input.

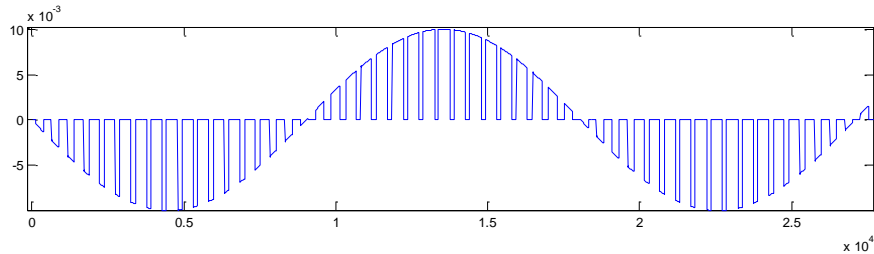


Figure 2.16: time domain error signal.

As it shown in fig 2-17, there are several noise sources in this proposed architecture.

E_q : quantization noise from the sigma-delta digital modulator.

E_{pwm} : harmonic distortion caused by the asymmetrical sampling process.

E_p : power supply noise and harmonic distortion from the switching output stage.

E_{adc} : quantization noise from the feedback ADC.

The follow equation 2-5 shows the transfer function of proposed closed-loop architecture.

$$Y(z) = \frac{H(z)}{1 + H(z)} (X(z) + E_{ADC}) + \frac{1}{1 + H(z)} (E_q + E_p) + E_{PWM} \quad \text{Equation 2-4}$$

Based on the equation, the E_p and E_q will be shaped and canceled by the feedback loop. However, E_{PWM} , E_{adc} cannot be attenuated by the feedback loop. Therefore, a good performance continuous-time sigma-delta ADC is needed to have a good noise performance. And a good digital PWM modulation scheme is need to reduce the in-band harmonic distortion.

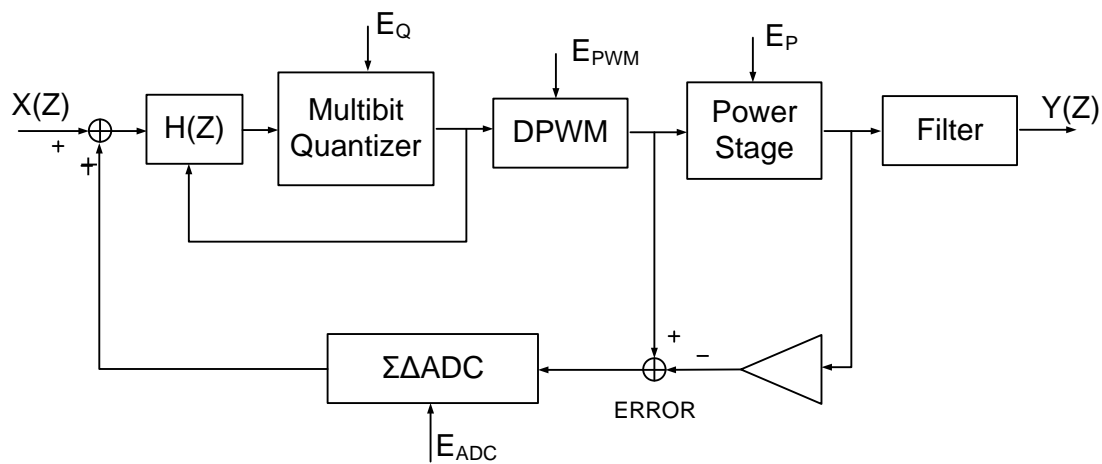


Figure 2.17: the block diagram of the proposed closed loop digital class D amplifier.

CHAPTER 3
SYSTEM DESIGN

In this chapter, the system design of the class D amplifier procedure is presented in details. The major tool used for the system design is MATLAB Simulink. For sigma-delta modulator design, the delta-sigma toolbox by Richard Schreier is used to generate noise transfer function (NTF) and signal transfer function (STF).

3.1 System Budget

Table 3.1: The system design spec target of the proposed class D audio amplifier.

Specification	Range
Input signal bandwidth	20~20 kHz
Input signal type	PCM
Minimum SNR	80 dB
digital delta-sigma modulator sampling frequency	$44.1\text{kHz} \times 32 = 1.4112 \text{ MHz}$
Digital triangle frequency	$44.1\text{kHz} \times 32/2 = 705.6 \text{ KHz}$
PWM switching frequency	$44.1\text{kHz} \times 32/2 = 705.6 \text{ KHz}$
Feedback ADC sampling frequency	$44.1\text{kHz} \times 32 = 1.4112 \text{ MHz}$

Table 3-1 shows the system design spec target of the proposed class D audio amplifier. Before starting the system design, the major system specs, i.e., signal-to-noise ratio (SNR), total-harmonic-distortion (THD), should be defined to have a good ideal of the design target. In most of the audio products, around 80dB SNR is needed for a good

audio quality. Therefore, for digital noise shaper and feedback continuous-time delta-sigma ADC, at least 80dB SNR is needed for SNR. Considering 6dB design margin, design target for both modulator and ADC are 86dB SNR.

3.2 Top-level System Model

MATLAB Simulink is used for the system top-level design. The top-level system includes an audio input, a digital class D modulator, a switching power stage, gain stages, a subtractor and a continuous delta-sigma ADC. Figure 3-1 shows the MATLAB Simulink class D amplifier system top-level model. The following parts will talk about each block in detail.

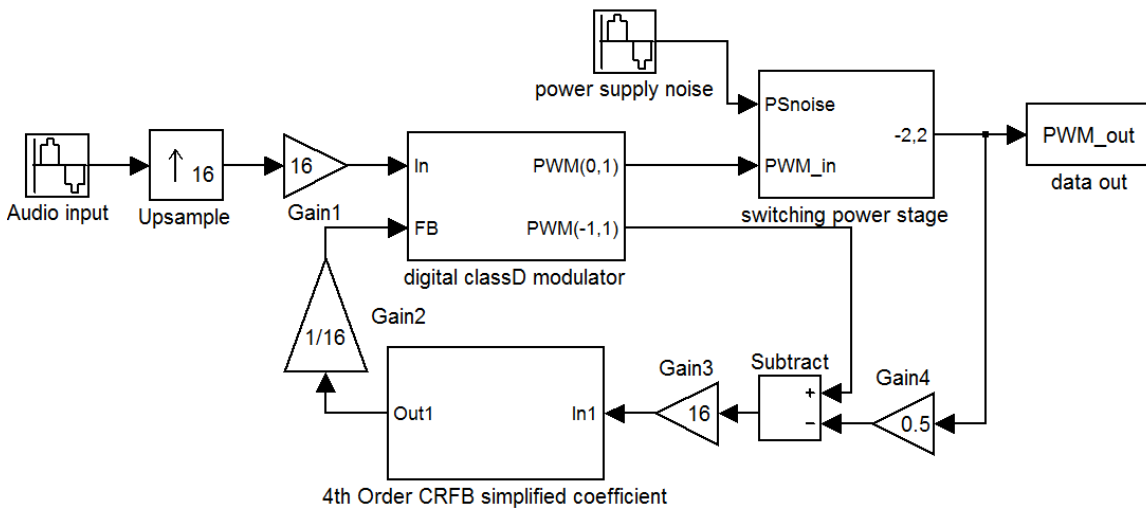


Figure 3.1: MATLAB Simulink top-level model of the class D amplifier system.

3.3 Digital Class D Modulator

The digital class D modulator mainly includes a delta-sigma modulator, a triangle wave generator and a comparator. Figure 3-2 shows the model of the digital class D modulator. This block is a purely digital block which will be implemented and verified in the FPGA. After finishing the MATLAB modeling, by using specific blocks which are compatible with Verilog/VHDL generation tool embedded in the MATLAB, Verilog/VHDL RTL code can be automatically generated using the MATLAB. By adding peripherals such as FIFO read/write function to the core RTL code, the generated code can be used by the FPGA.

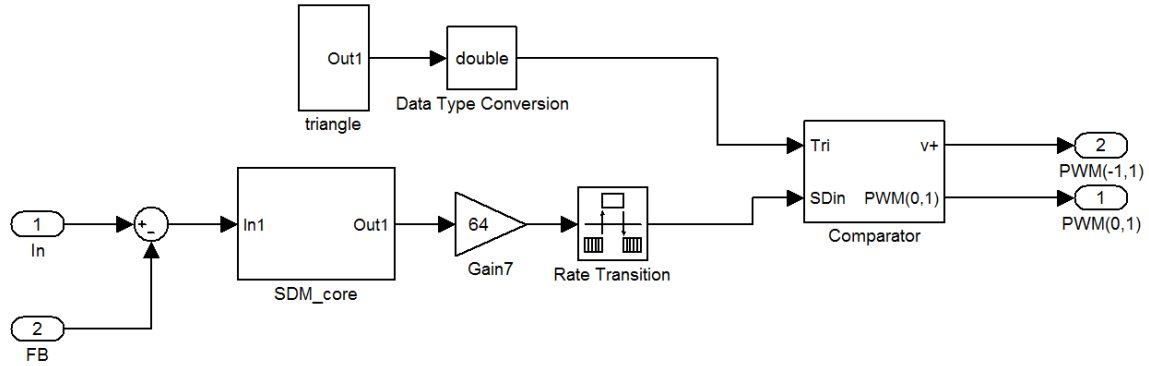


Figure 3.2: MATLAB Simulink sub-level model of the digital class D modulator.

3.3.1 Delta-sigma Noise-shaper

Table 3-2 shows the specifications of the desired delta-sigma noise-shaper. The SNR target for the delta-sigma noise-shaper is 86dB. To meet the design target, a 3rd

order sigma-delta digital modulator with $OSR = 32$ is used to meet the SNR requirement. Cascade of Integrators with Distributed Feedback (CIFB) architecture is used to achieve good performance. Figure 3-3 shows a general diagram of the CIFB architecture. Eq 3-1 and 3-2 is a general expression of the noise transfer function (NTF) for the CIFB architecture. Eq 3-3 is a general expression of the signal transfer function (STF) for the CIFB architecture.

Table 3.2: the specifications of the desired delta-sigma noise-shaper.

Specification	Range
Signal Bandwidth	20~20 kHz
Oversampling ratio	32
Sampling frequency	44.1kHz X 32 = 1.4112 MHz
Minimum SNR	86 dB

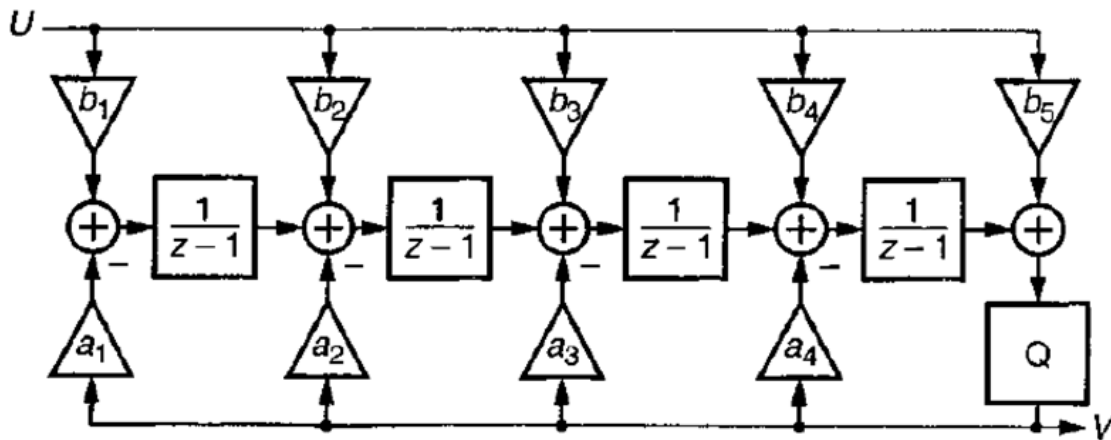


Figure 3.3: A general diagram of the CIFB architecture.

By using delta-sigma toolbox, the NTF can be generated for the desired performance based on the order, OSR and desired out of band noise. “synthesizeNTF” is the function to generate the NTF for a delta-sigma modulator. [14]

`ntf = synthesizeNTF(order, OSR,opt,H_inf,f0)`

order: the order of the NTF.

OSR: oversampling ration.

Opt: 0: puts all NTF zeros at DC.

1: optimizes the NTF zeros.

2: for even order modulators, puts two zeros at DC, but optimizes the rest.

H_inf: the maximum out of band gain of the NTF.

F0: the normalized center frequency for bandpass modulator.

With the help of delta-sigma toolbox, the NTF and STF are obtained as shown in eq. 3-4, 3-5 respectively. Figure 3-4 shows the pole-zero plot of the NTF in z domain. Fig 3-5 shows the frequency response of the NTF.

$$H_{ntf} = \frac{1.3304(z^2 - 1.443z + 0.564)}{(z-1)^3} \quad \text{Equation 3-1}$$

$$H_{stf} = \frac{1.33(z^2 - 1.919z + 0.7504)}{z^3 - 3z^2 + 3z - 1} \quad \text{Equation 3-2}$$

After obtaining the initial NTF, quantizer level needs to be decided based on the SNR spec. The delta-sigma toolbox can help to calculate the SNR of the NTF with quantizer levels. “simulateSNR” is the function to simulate a modulator with various sine wave amplitudes and calculate the SNR for each input.

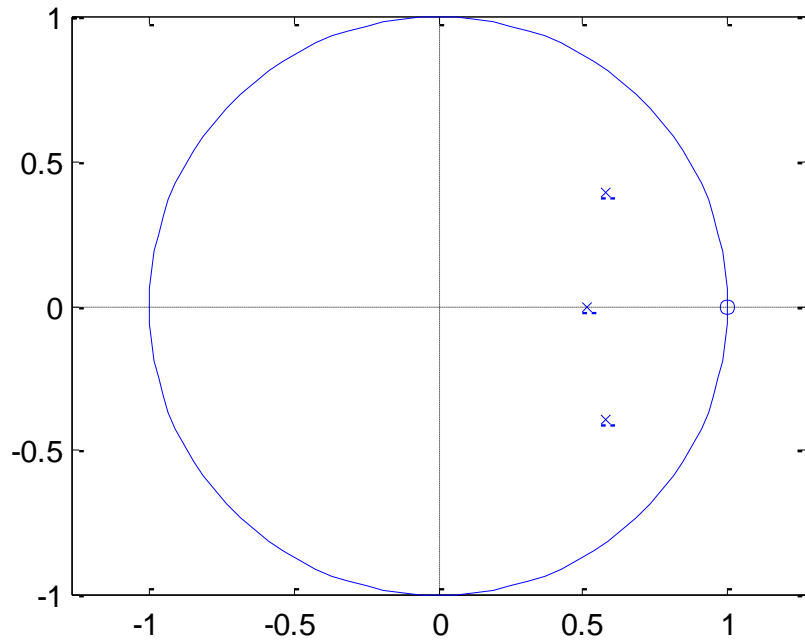


Figure 3.4: the pole-zero plot of the NTF in z domain.

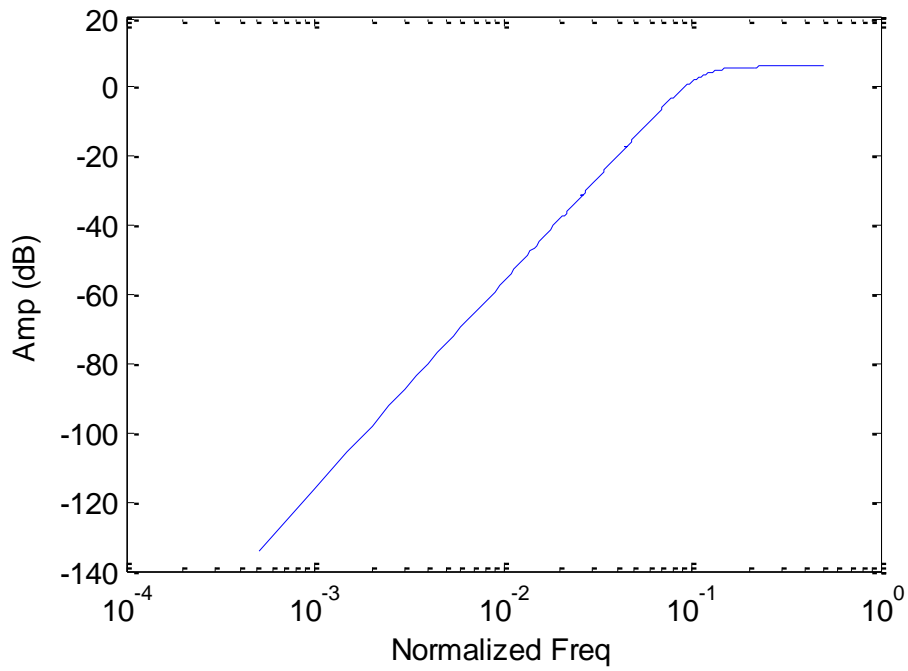


Figure 3.5: the frequency response of the NTF.

`[snr,amp] = simulateSNR(ntf,osr,amp,f0,nlev,f,k)`

`ntf`: noise transfer function.

`osr`: oversampling ration.

`amp`: a row vector listing the amplitudes to use. 0dB means a full-scale sine wave peak value is `nlv-1`.

`f0`: center frequency of the modulator.

`nlev`: number of quantizer levels.

`f`: normalized test sine wave frequency. Default `f=1/(4*osr)`.

`k`: 2^k number of FFT points.

Fig3-6 shows the calculated SNR for the NTF shown in eq3-4 and 6-bit quanziter. 105dB SNR and 100dB dynamic range is obtained.

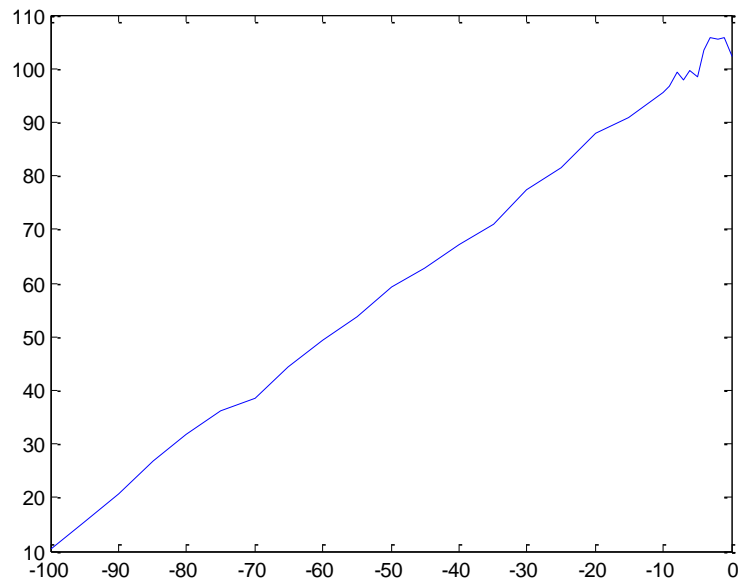


Figure 3.6: the calculated SNR for the NTF shown in eq3-4 with 6-bit quanziter.

From the NTF generated by the delta-sigma toolbox and the architecture chosen to use, the coefficient for the CIFB architecture can be find out by comparing two NTFs. Delta-sigma toolbox can also be used to help finding out the coefficient for some basic topologies. “realizeNTF” is the function to convert a NTF into a set of coefficients for a particular modulator topology. [14]

$$[a,g,b,c] = \text{realizeNTF}(\text{ntf},\text{form},\text{stf})$$

ntf: noise transfer function

Form: CRFB Cascade of resonators, feedback form.

CRFF Cascade of resonators, feedforward form.

CIFB Cascade of integrators, feedback form.

CIFF Cascade of integrators, feedforward form.

CRFBD CRFB with delaying quantizer.

CRFFD CRFF with delaying quantizer.

stf: a zpk signal transfer function. Default is 1.

The coefficient for the CIFB topology chosen for the 3rd order delta-sigma modulator is showing below:

$$a = 0.1617 \quad 0.7417 \quad 1.3304; b = 0.1617 \quad 0.7417 \quad 1.3304 \quad 1.0000; c = 1 \quad 1$$

1.

Fig3-7 shows the MATLAB Simulink model for the delta- sigma modulator. 6-bit quantizer is used for better stability and SNR.

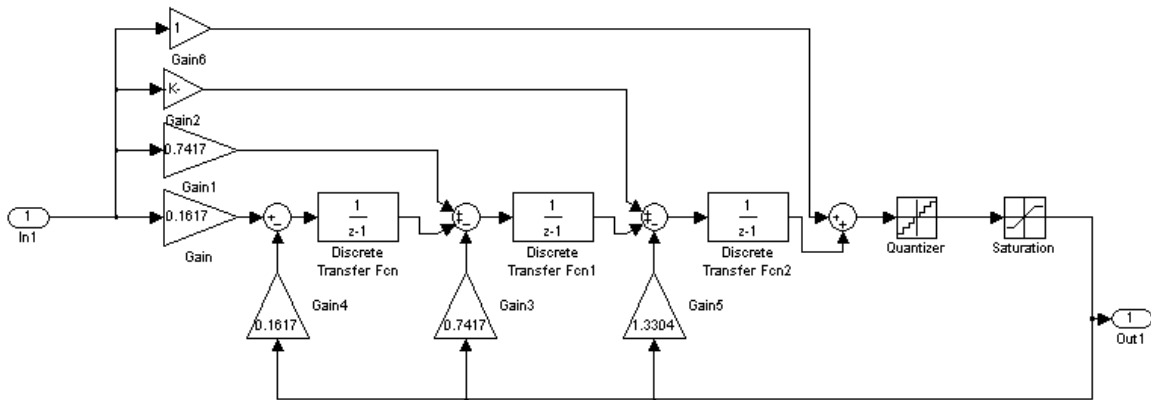


Figure 3.7: the MATLAB Simulink model for the delta- sigma modulator.

3.3.2 Triangle-wave Generator

The triangle-wave generated by this block is a 6-bit 64 levels triangle-wave running up-and-down. The frequency of the triangle-wave is $44.1\text{kHz} \times 32/2 = 705.6$ KHz. The generator block is running at $44.1\text{kHz} \times 32 \times 2^6 = 90.3168\text{MHz}$.

A 7-bits free-running counter block is used to model the triangle wave. This block overflows back to zero, after it reaches the maximum value 127. Figure 3-8 shows the MATLAB Simulink model for the triangle-wave generator.

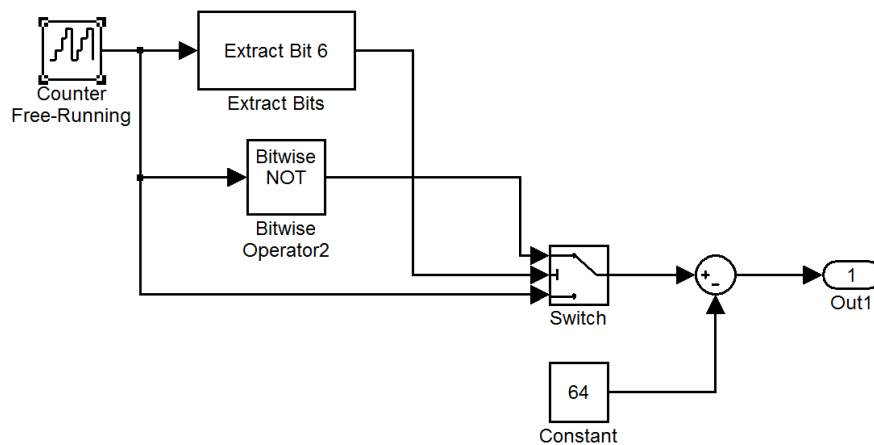


Figure 3.8: the MATLAB Simulink model for the triangle-wave generator.

3.3.3 Comparator

A subtractor and a compare to zero block are used to model the comparator. The delta-sigma modulator output is subtracted from the triangle-wave. If the result is equal or larger than zero, the output will be one. Figure 3-9 shows the MATLAB Simulink model for the comparator.

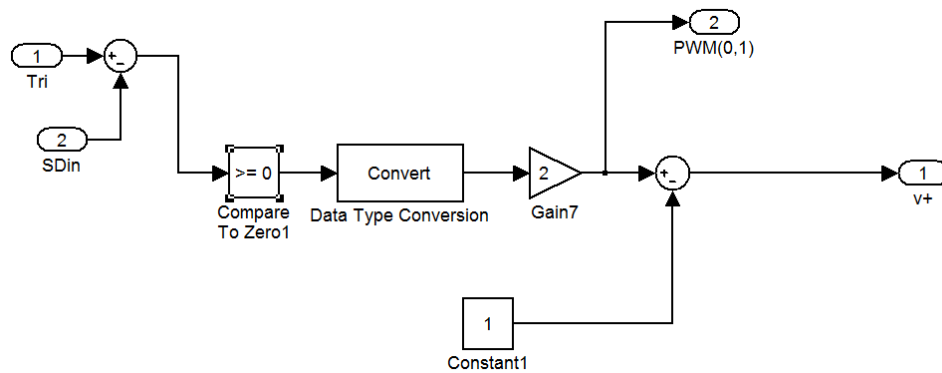


Figure 3.9: the MATLAB Simulink model for the comparator.

3.3.4 Simulation Results for Digital Class D Modulator

Figure 3-10 and 3-11 shows the frequency spectrum of the PWM output of the digital class D modulator.

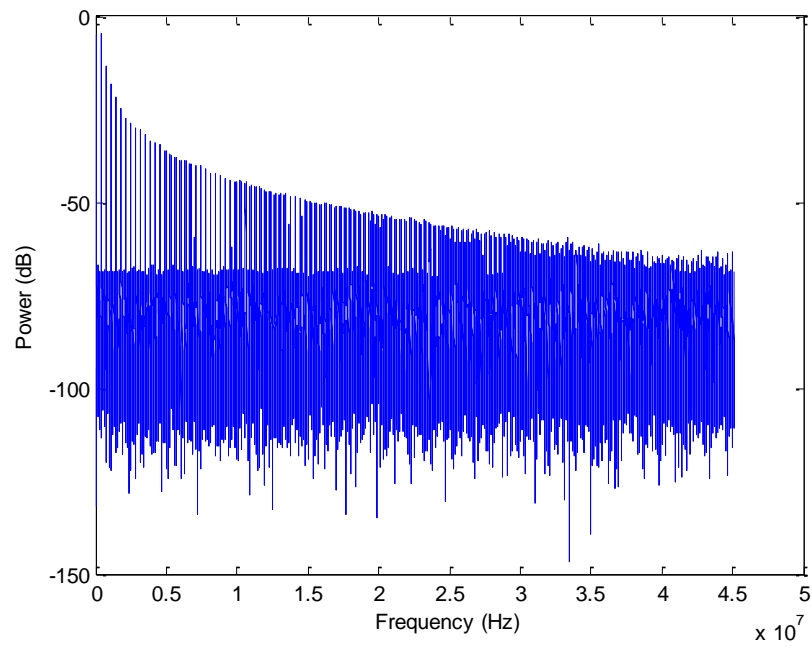


Figure 3.10: the frequency spectrum of the PWM output of the digital class D modulator.

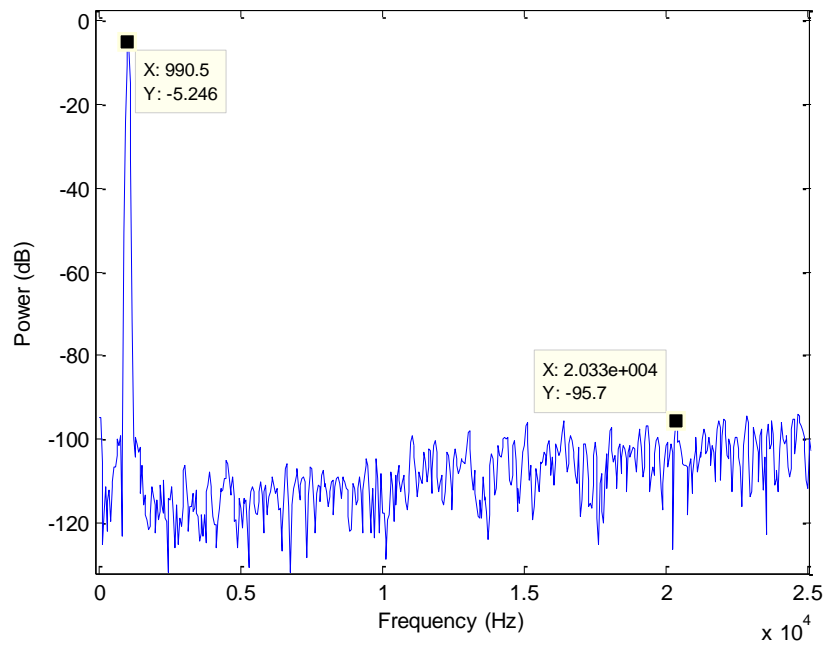


Figure 3.11: Zoom in of the frequency spectrum of the PWM output in the audio band.

3.4 Continuous-Time Delta-Sigma Analog-to-Digital Convertor (CTDSADC)

Table 3.3: the specifications of the desired continuous-time delta-sigma analog-to-digital convertor.

Specification	Range
Signal Bandwidth	20~20 kHz
Oversampling ratio	32
Sampling frequency	44.1kHz X 32 = 1.4112 MHz
Input voltage range	-500mV to 500mV
Minimum SNR	86 dB

Table 3-3 shows the specifications of the desired continuous-time delta-sigma analog-to-digital convertor. Continuous-time ADC has the inherent filtering effect which can eliminate the use of the antialiasing filter in front of an ADC. To meet the 86dB SNR design target for ADC, a 4th order sigma-delta analog ADC with $OSR = 32$ is used. Cascade of Resonators with Distributed Feedback (CRFB) architecture is used because the resonator can separate the zero locations from DC to the frequency we desired. Fig 3-12 shows a diagram of the CRFB architecture used in the design. Eq 3-3 is the expression of the noise transfer function (NTF) for the proposed CRFB architecture in s domain. Eq 3-4 is the expression of the signal transfer function (STF) for the proposed CRFB architecture in s domain.

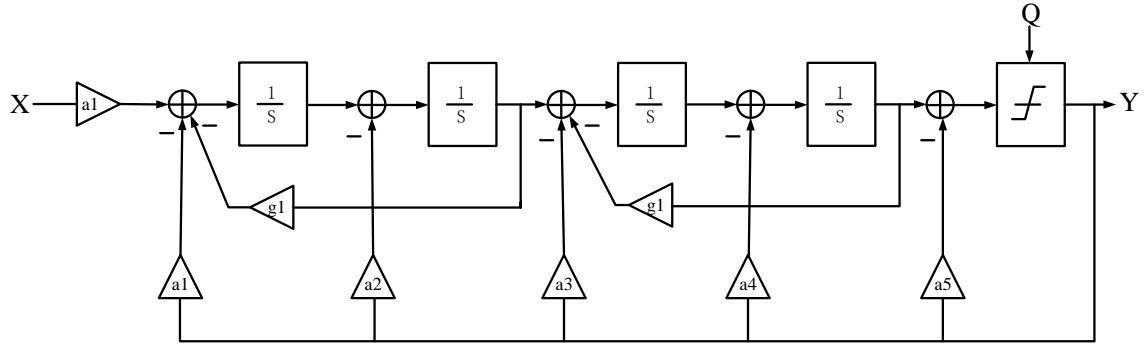


Figure 3.12: a diagram of the CRFB architecture used in the design.

$$H_{ntf} = \frac{(1+a_5) \cdot (s^2+g_1) \cdot (s^2+g_2)}{s^4+a_4s^3+(g_1+g_2+a_3)s^2+(a_2+a_4g_1)s+(g_1g_2+g_3g_1+a_1)} \quad \text{Equation 3-3}$$

$$H_{stf} = \frac{a_1}{s^4+a_4s^3+(g_1+g_2+a_3)s^2+(a_2+a_4g_1)s+(g_1g_2+g_3g_1+a_1)} \quad \text{Equation 3-4}$$

To find the noise transfer function for a continuous time modulator, a functionally equivalent discrete time modulator prototype is selected first and then transfer the discrete time modulator into a continuous time modulator with a same NTF. The impulse response method is used to convert the DT system to a CT system. Basically, the impulse response of the DT modulator should be equivalent to the impulse response of the CT modulator. Table 3-4 can be used to convert the z domain function to s domain function.

Table 3.4: Laplace and Z-Transfer Pairs.

$f(t)$	$F(s)$	$f(k)$	$F(z)$
$u(t)$, unit step	$\frac{1}{s}$,	$u(k)$ unit step	$\frac{z}{z-1}$
$tu(t)$	$\frac{1}{s^2}$	$kTu(k)$	$\frac{Tz}{(z-1)^2}$
$e^{-at}u(t)$	$\frac{1}{s+a}$	$(e^{-aT})^k u(k)$	$\frac{z}{z-e^{-aT}}$
$te^{-at}u(t)$	$\frac{1}{(s+a)^2}$	$kT(e^{-aT})^k u(k)$	$\frac{Tze^{-aT}}{(z-e^{-aT})^2}$
$\sin(\omega t)u(t)$	$\frac{\omega}{s^2 + \omega^2}$	$\sin(k\omega T)u(k)$	$\frac{z \sin \omega T}{z^2 - 2z \cos \omega T + 1}$
$\cos(\omega t)u(t)$	$\frac{s}{s^2 + \omega^2}$	$\cos(k\omega T)u(k)$	$\frac{z(z - \cos \omega T)}{z^2 - 2z \cos \omega T + 1}$

The MATLAB with delta-sigma toolbox will help discrete to continuous time transfer. First of all, the NTF and STF is obtained using “synthesizeNTF” function as shown in eq. 3-8, 3-9 respectively. The order is set to 4, osr is 32, zeros are optimizes for the NTF, the maximum out of band gain is 2.5. Figure 3-13 shows the pole-zero plot of the NTF in z domain.

$$H_{ntf} = \frac{(z^2 - 1.999z + 1)(z^2 - 1.993z + 1)}{(z^2 - 1.023z + 0.2791)(z^2 - 1.204z + 0.5708)} \quad \text{Equation 3-5}$$

$$H_{stf} = \frac{1.7651(z - 0.6855)(z^2 - 1.525z + 0.6948)}{(z^2 - 1.999z + 1)(z^2 - 1.993z + 1)} \quad \text{Equation 3-6}$$

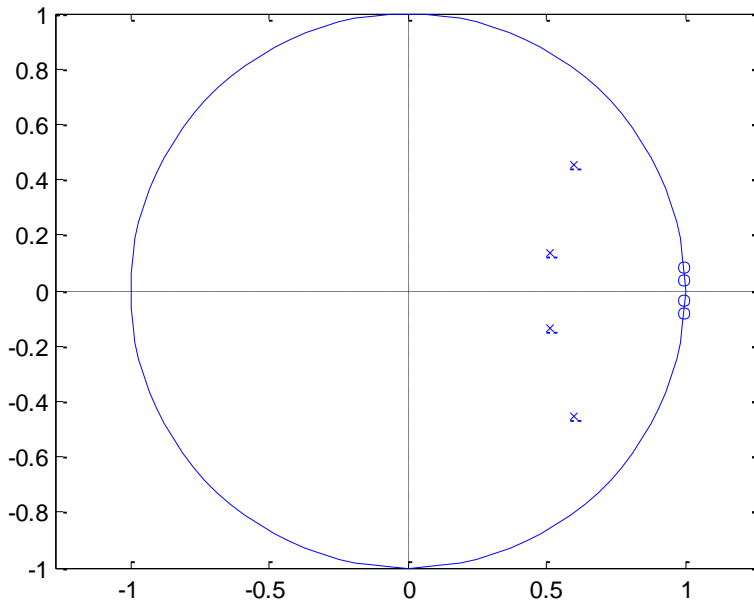


Figure 3-13: the pole-zero plot of the NTF in z domain.

MATLAB signal processing toolbox is used to convert the z domain NTF to s domain NTF. “d2c” function is used to convert discrete-time system to continuous-time system. Eq 3-10 and 3-11 represent the STF and NTF of the equivalent continuous-time delta-sigma ADC respectively. Figure 3-14 shows the frequency response of the NTF.

$$H_{\text{stf}} = \frac{1.2343(s+0.3744)(s^2+0.3687s+0.2037)}{(s^2+0.001114)(s^2+0.007147)} \quad \text{Equation 3-7}$$

$$H_{\text{ntf}} = \frac{(s^2+0.001114)(s^2+0.007147)}{(s^2+0.9026s+0.251)(s^2+0.3317s+0.3751)} \quad \text{Equation 3-8}$$

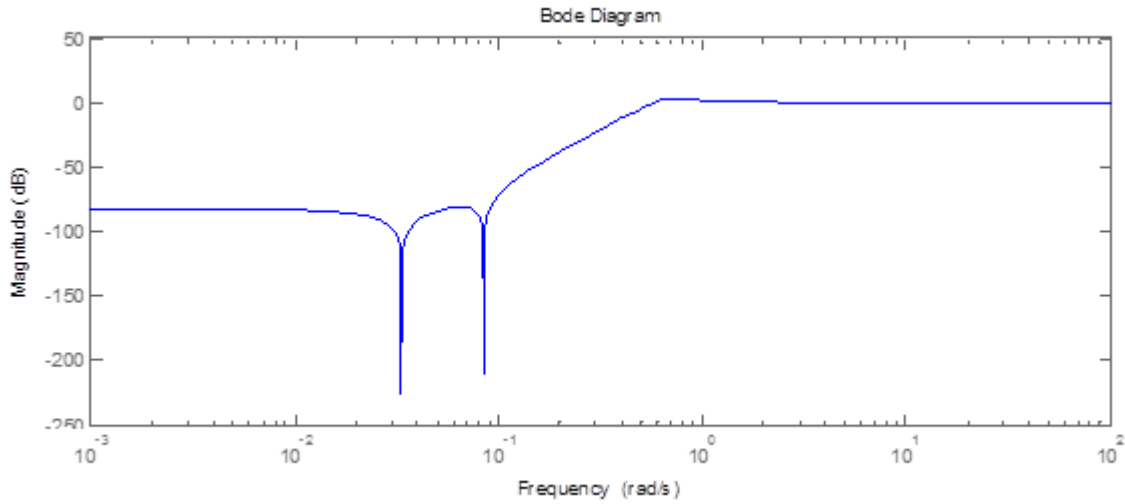


Figure 3-14: the frequency response of the NTF.

Comparing the obtained NTF in Eq3-7 and NTF of the proposed CRFB architecture in eq 3-3, the coefficients in eq3-3 can be calculated which is shown below:

$$g1 = 0.0011; g2 = 0.0071; a4 = 1.2317; a3 = 0.9164; a2 = 0.4195; a1 = 0.0931.$$

Considering the circuit implementation of the coefficients, a simplified set of coefficients is used for better R and C matching. The coefficients are now set to:

$$g1 = 1/256; g2 = 1/32; a4 = 1.25; a3 = 1; a2 = 0.5; a1 = 0.1.$$

Figure 3-15 shows the MATLAB Simulink model for the continuous-time delta-sigma ADC. 2-bit quantizer is used for better stability and SNR.

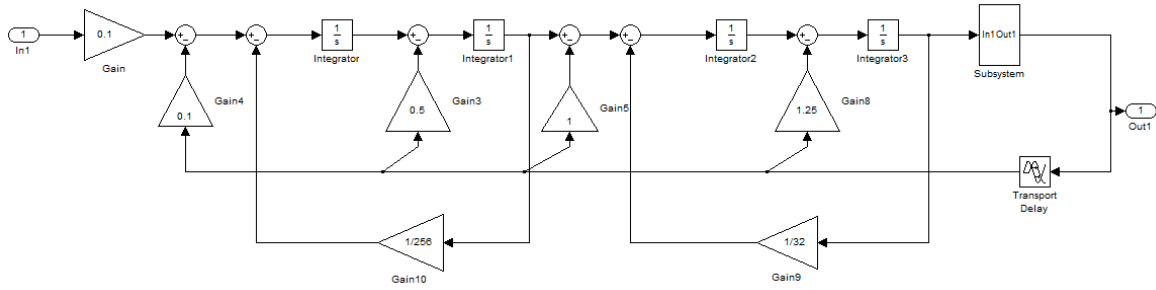


Figure 3.15: the MATLAB Simulink model for the continuous-time delta-sigma ADC.

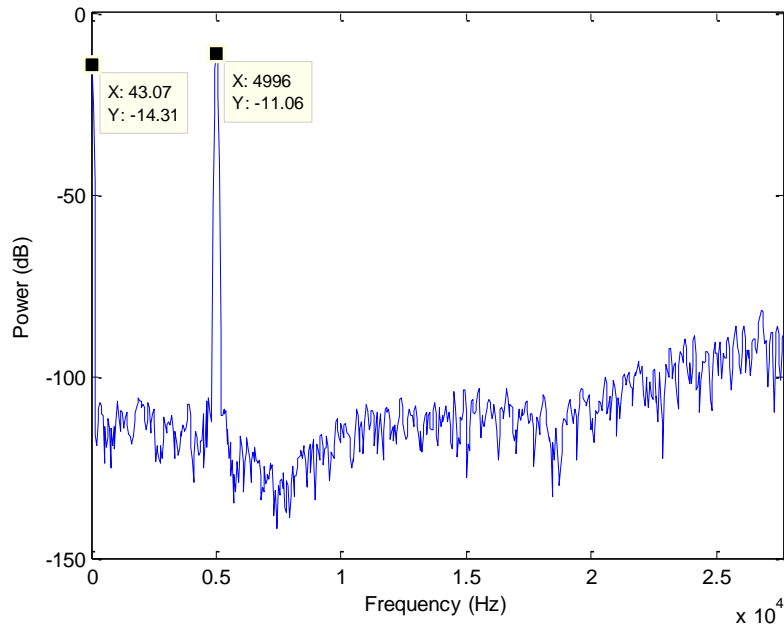


Figure 3.16: the power spectrum of the ADC output with 5KHz sine-wave input in audio band.

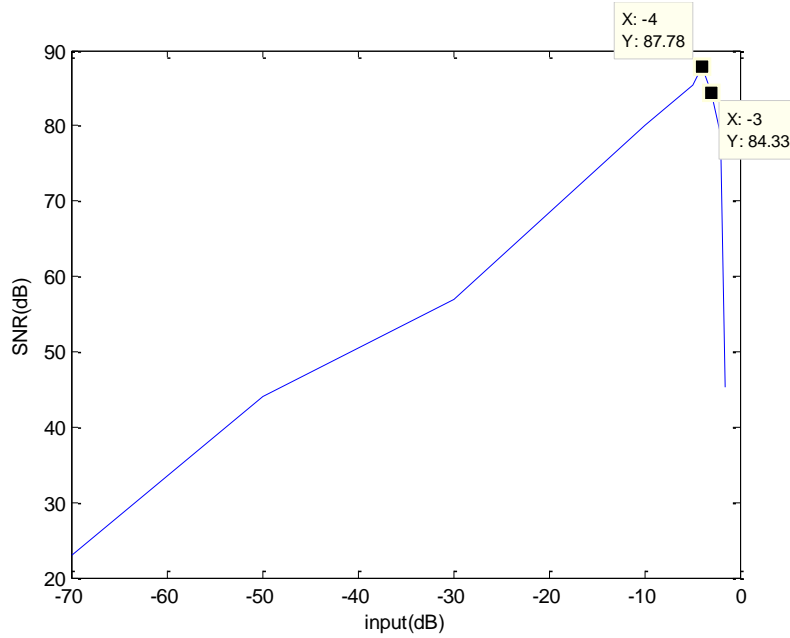


Figure 3.17: SNR vs. Input amplitude for the CTADC.

CT ADC input signal range

The input signal range is estimated based on the power supply noise amplitude. Assuming maximum power supply noise amplitude is 80mV (160mV peak-peak) and attenuator is $\frac{1}{2}$, the feedback error signal is 40mV (80mV peak-peak). Gain stage before CT ADC is 24dB (X16) which means amplified feedback signal amplitude is 640mV (1.28V peak-peak). Assuming the gain error of the attenuator is $\pm 5\%$, and analog power supply is 3V, then the maximum offset is $\pm 75\text{mV}$. With the number above, the final CT ADC input range is $\pm 715\text{mV}$ which is about -3dB.

3.5 Noise Budget

Based on the matlab simulink model, the total gain of the subtractor-gain block is $16 \cdot V_{ddd}/V_{dda} \approx 13.33$. In this design, subtractor-gain block has two stages. The first stage is a subtractor with gain of 4. The second stage is a gain stage with gain of 3.33.

To meet the system noise requirement -80 dB, the input referred noise of the feedback analog system should meet -102.5dB noise requirement. This noise includes the noise of subtractor-gain and continuous-time-delta-sigma-ADC.

$$P_{n_total} = P_{n_sub} + P_{n_ADC}/Gain_{sub}$$

Equation 3-9

Considering the dominant noise will be the subtractor-gain block, the total input referred noise of this block should be at least <-103dB (<7uVrms) which gives 35uVrms noise budget for ADC block.

Total input referred noise by calculation: 6.7uVrms (-103.5dB)

Total input referred noise by simulation: 5.8uVrms (-104.7dB)

3.5.1 Subtractor-gain Noise

$$P_{n_in} = P_{n_1st} + P_{n_2nd}/Gain_1^2$$

Equation 3-10

$$P_{n_1st} = 2P_{n_Ri} + P_{n_amp} (1 + Gain_1)^2/Gain_1^2 + 2 P_{n_Rf}/Gain_1^2$$

Equation 3-11

$$P_{n_2nd} = 2P_{n_Ri} + P_{n_amp} (1 + Gain_2)^2/Gain_2^2 + 2 P_{n_Rf}/Gain_2^2$$

Equation 3-12

Table 3.5: Subtractor-gain stage noise estimation.

Block name	Input referred noise	RMS noise (Vrms)
Subtractor_gain stage 1	Vn_Ri	1.82
	Vn_Rf	3.64
	Vn_amp	3.5
	total	5.23
Gain stage 2	Vn_Ri	1.82
	Vn_Rf	2.99
	Vn_amp	3.5
	total	1.4
total input referred noise	total	6.625
Simulation results	total	5.64

3.5.2 Continuous-time Delta-sigma ADC Integrator Noise

The noise of CTSDADC is dominated by the noise of the first integrator. The noise of the rest stages will be divided by the gain of the first integrator. For simplicity, we only consider first stage integrator in the noise calculation. Based on Eq. 3-9, our design target is $35\mu\text{Vrms}$. The detailed noise simulation results will be shown in Chapter 5.

CHAPTER 4

DIGITAL IMPLEMENTATION- FPGA BASED EMBEDDED SYSTEM DESIGN

In this chapter, the digital implementation and verification of the digital class D modulator proposed in Chapter 3.1 is presented by using the FPGA based embedded system design. By using the FPGA based embedded system, an MP3 type of music playback system is implemented, whose audio processing core is the digital class D modulator presented in Chapter 3.1. The digital class D modulator core includes a delta-sigma modulator, a triangle wave generator, a comparator and decimators. A memory is needed to store the audio signal or music. PLB bus is used to read the data from memory to the processor core. The output of the digital class D modulator processing core is a PWM signal, which needs to be filtered down to the audio frequency with decimators. The decimated output data is sent to a speaker or an audio playing chip which can play the music through the speaker.

Digilent Atlys Spartan®-6 FPGA Development Kit is used to prototype the embedded system which represents the digital portion of the class D system. Using the development board, the digital design can be verified by looking at the output of the FPGA. With the peripherals of the development board, the music can be stored in the onboard memory. Digilent A97 chip is also on the development board which can be used to drive the onboard speaker. A listening demo can be used as a direct way to verify the digital design. Fig 4-1 shows the block diagram of the embedded system implemented on the

4.1 Atlys Spartan®-6 FPGA Development Kit

The Atlys development board is based on a Xilinx Spartan 6 LX45 FPGA. It's a ready-to-use digital circuit development platform. Figure 4-1 shows a photo of the Atlys Spartan 6 FPGA development board.

The Atlys development board consists of a collection of on-board high-end peripherals. The Atlys board includes 128Mbyte DDR2 memory array, audio, USB ports, Gbit Ethernet and HDMI Video which make it an ideal host for complete digital embedded systems. Fig4-2 shows a block diagram of Atlys development board with its peripherals. Xilinx's MicroBlaze can be used as an embedded processor for the Atlys development board. Atlys development board is fully compatible with all Xilinx CAD tools, including ChipScope, EDK, and the free WebPack.

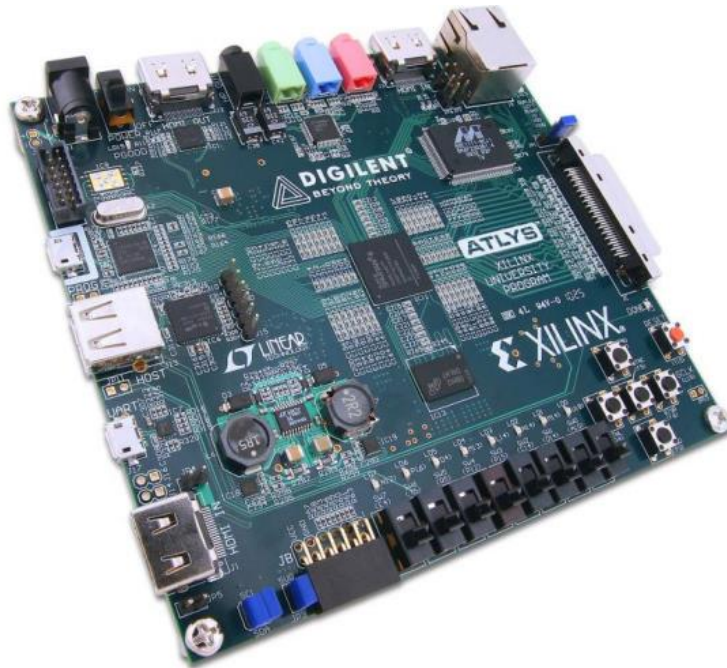


Figure 4.1: The Atlys Spartan 6 FPGA development board.[20]

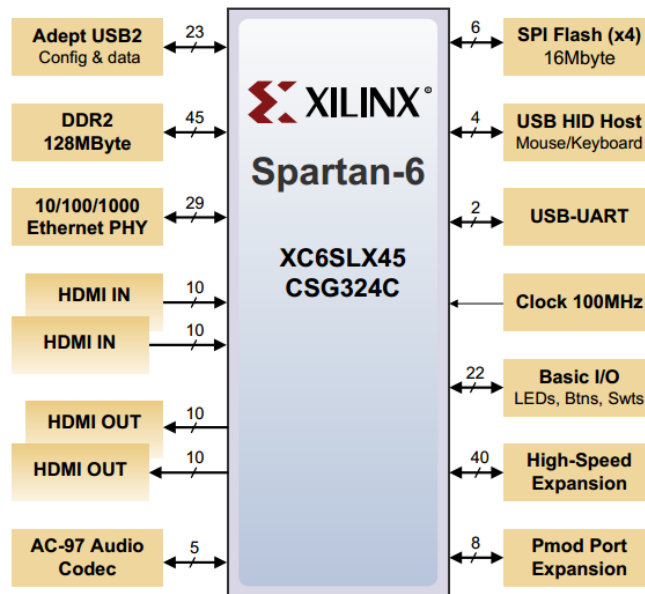


Figure 4.2: Xilinx Spartan-6 based Atlys development board with its peripherals.[20]

The Spartan-6 LX45 has a lot of advantages. As it mentioned on the reference manual, it offers:

1. 6822 slices each containing four 6-input LUTs and eight flip-flops.
2. 2.1Mbits of fast block RAM.
3. 4 clock tiles (8 DCMs & 4 PLLs).
4. 6 phased-locked loops.
5. 58 DSP slices.
6. 500MHz+ clock speeds.

4.2 Embedded System Design

An embedded system is a digital system with at least one processor that implements a hardware function that is a part or all of the digital system. The processor that is used in an embedded system is an embedded processor.

The embedded system design flow consists of the following steps: modeling, refining, hardware-software partitioning, scheduling, and mapping.

Embedded system design can be broken down into two main parts, hardware and software. The hardware aspect of the design is implemented using hardware packages, hardware description language programs, and/or gates. The software aspect deals with the high level C or C++ program that performs the sequence of steps necessary for the system to operate as specified.

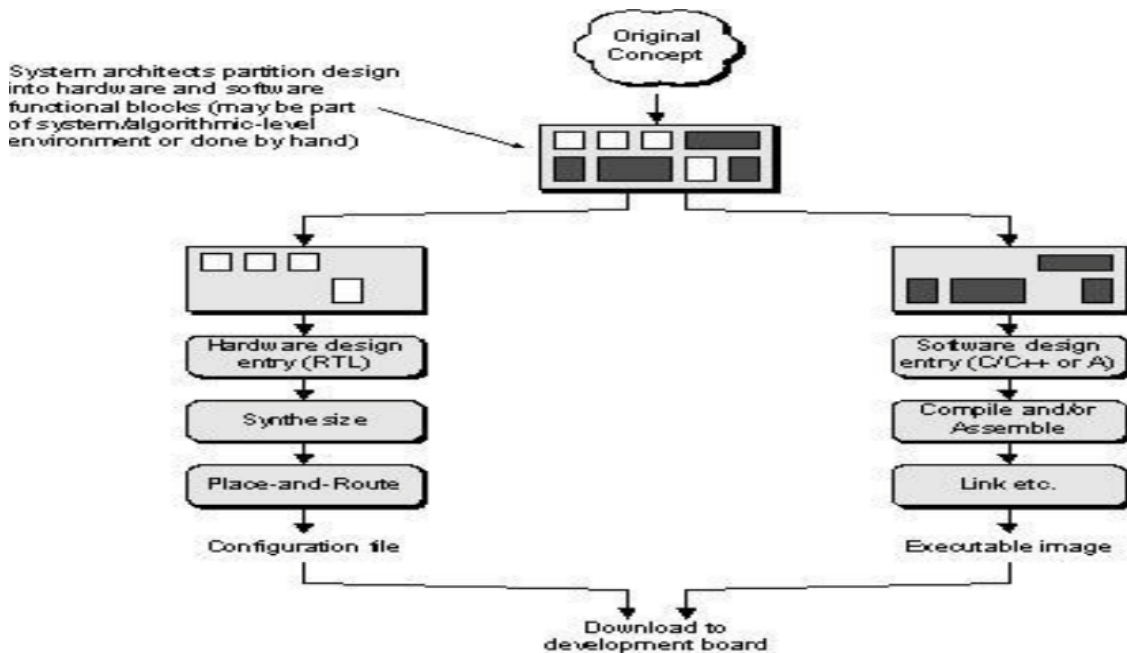


Figure 4.3: The embedded system design flow.

An embedded system contains many elements including peripherals, processor, memory, software and algorithms.

4.3 Microblaze Processor Based Embedded System

Atlys FPGA Development board provided all the elements that an embedded system needs. Xilinx Platform Studio (XPS) is used to create a processor system targeting the Atlys FPGA development board. The Base System Builder (BSB) can be used to quickly create a MicroBlaze system in XPS. By using Xilinx IP cores available in the Embedded Development Kit (EDK), the hardware part of the embedded system can be implemented. Fig4-4 shows a Microblaze processor based embedded system and the peripherals. Using the Xilinx EDK tool, a custom peripheral can be added to the microblaze processor system which implements the user defined function. The detailed instruction can be find in Rapid Prototyping of Embedded System Using FPGA[21].

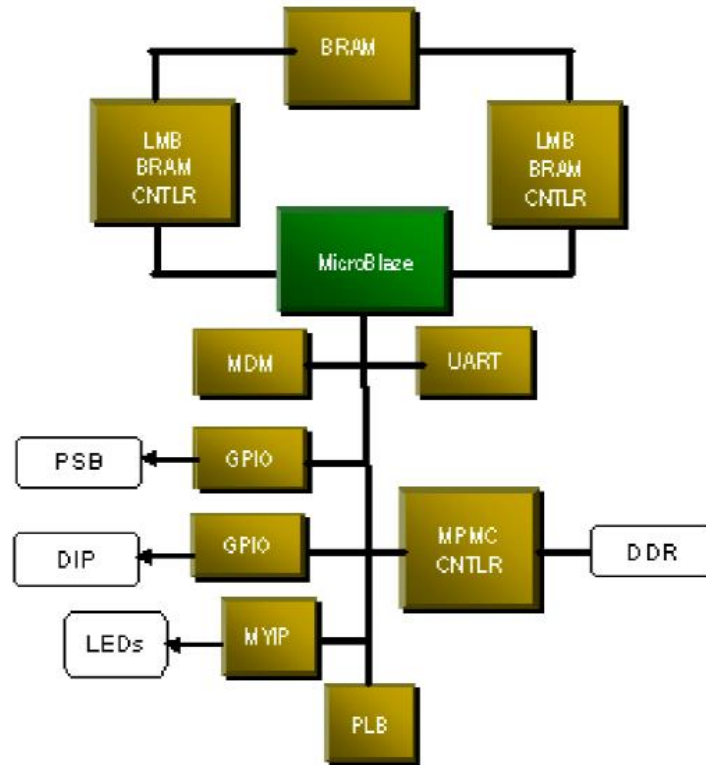


Figure 4.4: The Microblaze processor based embedded system and the peripherals.

XPS runs the following tools to generate the downloadable bitstream:

1. Platgento generate the netlists of the processor system
2. Libgento generate libraries and drivers
3. GNU Compiler to generate the executable
4. ISE tools in xflowmode to generate the bitstream
5. BitInitto update the bitstream with the TestAppexecutable
6. iMPACT to download the bitstream to the target hardware platform

4.4 Microblaze Processor Based Class D System Digital Design

4.4.1 Digital Class D Modulator HDL Code Generation

The class D digital system in the Microblaze processor includes an SDM_core, a triangle wave generator, a comparator and three 3rd order sinc filters. Figure 4-5 shows the MATLAB Simulink model used for the embedded design. Since the onboard audio controller only receive audio band frequency signal, a decimator is added compared to the model shown in Figure 3-2.

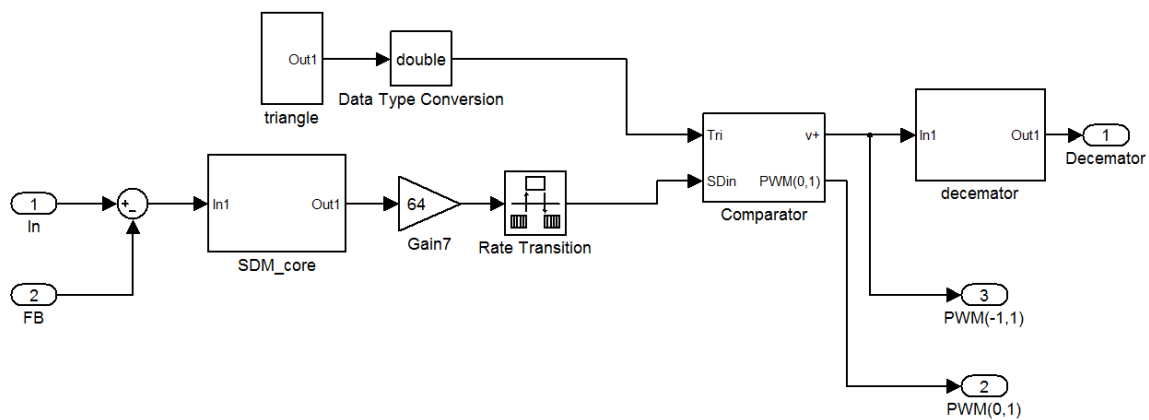


Figure 4.5: the MATLAB Simulink model used for the embedded design.

The details of each block have been presented in Chapter 3. For the decimator, three 3rd order sinc filters are used to filter out the out-of-band noise and downsample the output to 44.1KHz. Figure 4-6 shows the MATLAB model of the decimator. The decimation factor of 16 is used for the first and second sinc filters. The decimation factor of 8 is used for the third sinc filter.

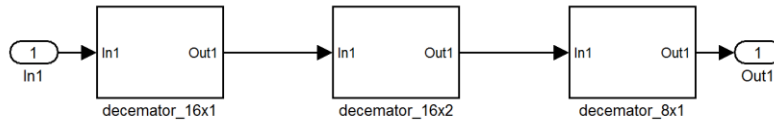


Figure 4.6: the MATLAB model of the decimator.

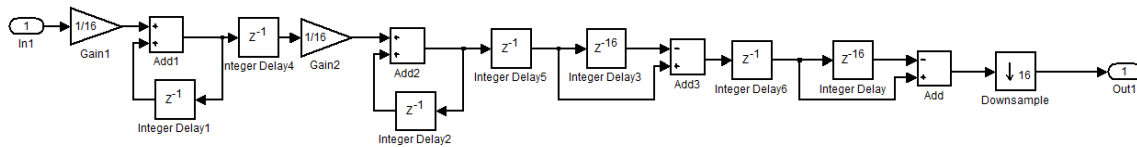


Figure 4.7: 3rd order sinc filter with decimation factor of 16.

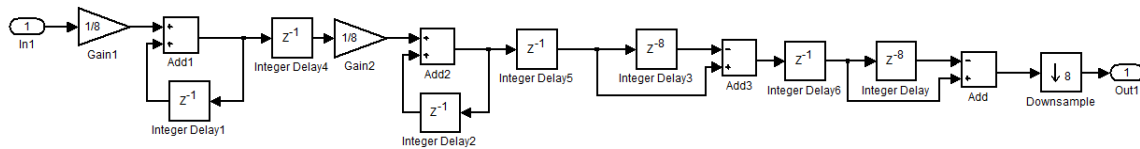


Figure 4.8: 3rd order sinc filter with decimation factor of 8.

After completing the model for the digital class D modulator, MATLAB has functions to automatically generate the HDL code from Simulink model:

1. hdlsetup: Set Simulink model parameters for HDL code generation.
2. Makehdl: Generate HDL code for a Simulink model or subsystem.

The HDL code generated by the MATLAB is shown in the Appendix.

4.4.2 Embedded System Design Using Xilinx EDK

The Rapid Prototyping of Embedded System Using FPGA[21] on MIT website gives a detailed instruction on how to setup Xilinx EDK based the system requirements.

The main requirement for the design shows in the following portion.

1. Clocks of the FPGA

Data transfer handshake is using an onboard generated low frequency and then divided down by 4 times. because the frequency needed is below the FPGA PLL low boundary. SDM_core, decimator and comparator is using an onboard generated 50MHz clock. Timing_Controller.v generates the low frequency clock for decimator and SDM_core. Table 4-1 shows a detailed clock setup for FPGA.

Table 4.1: Clock setup for FPGA.

Block name	Frequency
PLL input	100MHz
Microblaze frequency	50MHz
Counter frequency:	50MHz
PLL output to SDM	3.125MHz (min freq from PLL)
Upsampled data reading freq	3.125MHz/8 (generated in the usr_logic.v)
Decimated output	50MHz/16/16/8

2. FPGA input/output data

The upsampling and AAF is done by matlab. Processed data named, `input_x8_nofirst8bit.bin` and `input_x8_withfirst8bit.bin`, is the input to the FPGA memory. FPGA output is saved under `...\SDM\workspace`, but its sampling frequency is x8. downsampling of 8 times needed if want to play.

3. FPGA software programming file (.elf file)

For SDK, .elf file under the main directory is the updated software. input audio file is `input_x8_nofirst8bit.bin`. If using the one under `...\data\xx.elf`, `input_x8_nofirst8bit.bin` should be used and FPGA board cannot play the audio.

4.4.3 Final Results

Figure 4-9, 4-10, 4-11 shows comparison of the power spectrum of the final PWM output with MATLAB simulation, HDL code ISE simulation and real FPGA output respectively. The MATLAB simulation output is the decimator output which includes three 3rd order sinc filters. Compared to raw PWM output shown in Figure 3-10 and 3-11, the maximum SNR is similar. The HDL code generated by MATLAB is verified using ISE first. Finally, the real FPGA output data is exported.

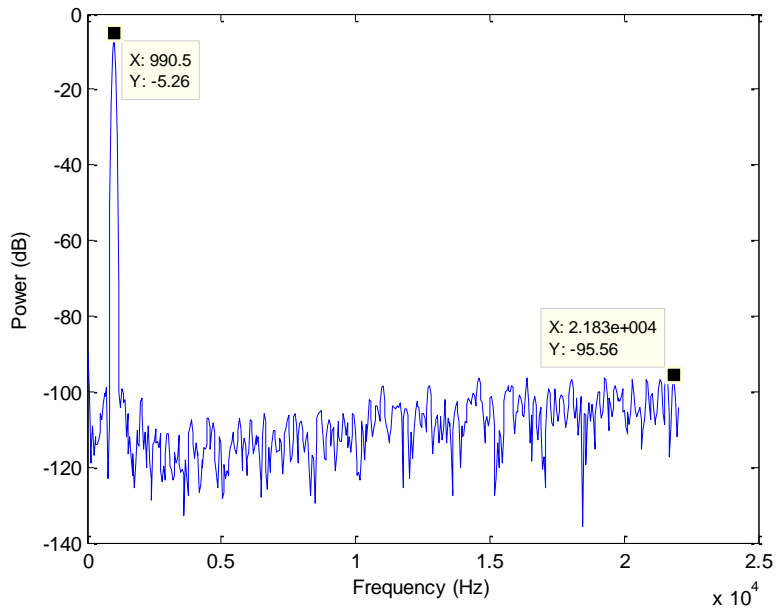


Figure 4.9: the power spectrum of the PWM output after decimator with MATLAB simulation.

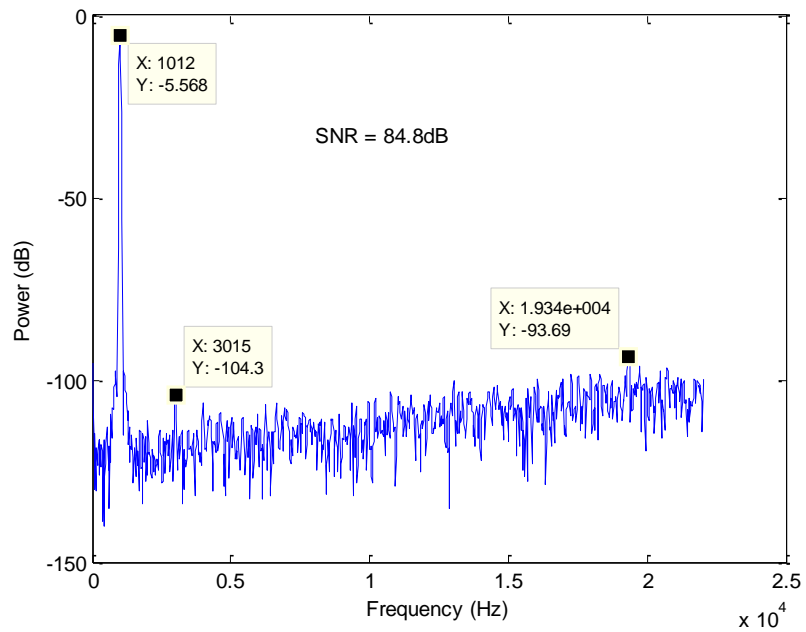


Figure 4.10: the power spectrum of the PWM output of HDL code with ISE simulation.

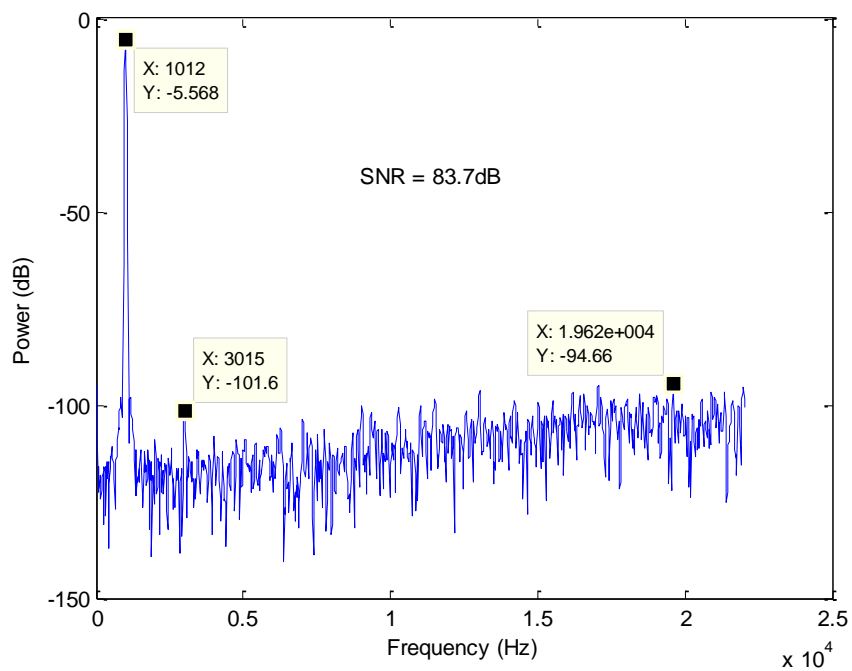


Figure 4.11: the power spectrum of the PWM output after decimator of FPGA output.

CHAPTER 5

ANALOG IMPLEMENTATION - SCHEMATIC DESIGN

In Chapter 4, the digital design of the class D modulator is presented in details. In this chapter, the analog design portion which includes a switching power stage, a continuous-time sigma-delta analog-to-digital convertor (CTSDADC), a subtractor-gain stage and a delay block is covered and described in details. The HDL code used in FPGA is also ported to into cadence for the verification of class D system. Analog-digital-mixed-signal (AMS/ADMS) top-level simulation is used to simulate and verify both HDL and analog schematics together.

The analog schematic design is using globe foundry 0.18um process with 3V analog supply, 3V output stage supply and 2.5V digital supply. Cadence 6.1 is the design tool used for design and simulation.

5.1 Class D Amplifier System Top-level Schematic and Test Bench

The class D amplifier system top-level schematic includes the designed class D amplifier and test bench. Figure 5-1 is the schematic of the top-level test bench.

The test bench includes the following items:

1. Analog supplies: 3V V_{dda} for subtractor, continuous-time ADC, delay and other analog blocks; 3V V_{ddio} for the switching power stage.
2. Digital supply: 2.5V V_{ddd} for digital class D modulator.
3. Clock: 90MHz clocks for digital class D modulator. The clock is divided down inside the digital class D modulator and the divided clock is used for continuous-time ADC.

4. Control signals: Clamp, reset, pdn, clk_enable.
5. Input signal: 20Hz~20KHz sinusoid signal.
6. 16-bits PCM analog-to-digital converter: VerilogA model to convert the analog input signal to a 16-bits PCM digital output.
7. Speaker: 32ohm load resistance speaker model.
8. Low pass filter: an ideal 5th order RC filter for the PWM output.
9. Data write: VerilogA model to sample and write the data to txt file.

The design class D amplifier system includes:

1. Digital class D modulator: HDL code generated by MATLAB.
2. Switching power stage.
3. Delay: to match the delay of the power stage.
4. Subtractor-gain stage.
5. Continuous-time delta-sigma ADC.

5.2 Continuous-time Delta-sigma ADC (CTADC)

5.2.1 ADC Top

The CTADC is an import design part in the analog loop. To avoid the sampling and anti-aliasing filter design, continuous-time ADC is chosen because of its inherent filtering effect. The system level design details have been covered in Chapter 3. The schematic will be built based on model shown in figure 3-15. The CTADC top-level schematic includes the following blocks: four integrators, quantizer, four current steering

DACs and bias blocks. Figure 5-2 shows the continuous-time delta-sigma ADC top-level schematic.

The feedback coefficients have obtained in Chapter 3: $g_1 = 1/256$; $g_2 = 1/32$; $a_4 = 1.25$; $a_3 = 1$; $a_2 = 0.5$; $a_1 = 0.1$. Active RC based circuit is used in the schematic design to implement integrators. Compared to gm-C based design, RC based architecture gives better linearity, but consumes more power. Since performance is the main concern of the design, active RC circuit is used for all integrators.

Besides the general power down and reset signals, clamp signal is a very important control signal for stability. The clamp signal clamps the input and output of the integrators which makes the amplifier in a unity gain configuration. It also resets the charges on the cap in case the integrator is saturated. After the ADC is powered up, clamp switches should be remain ON until amplifiers gets stable, then the clamps will be opened and the system is in normal operation.

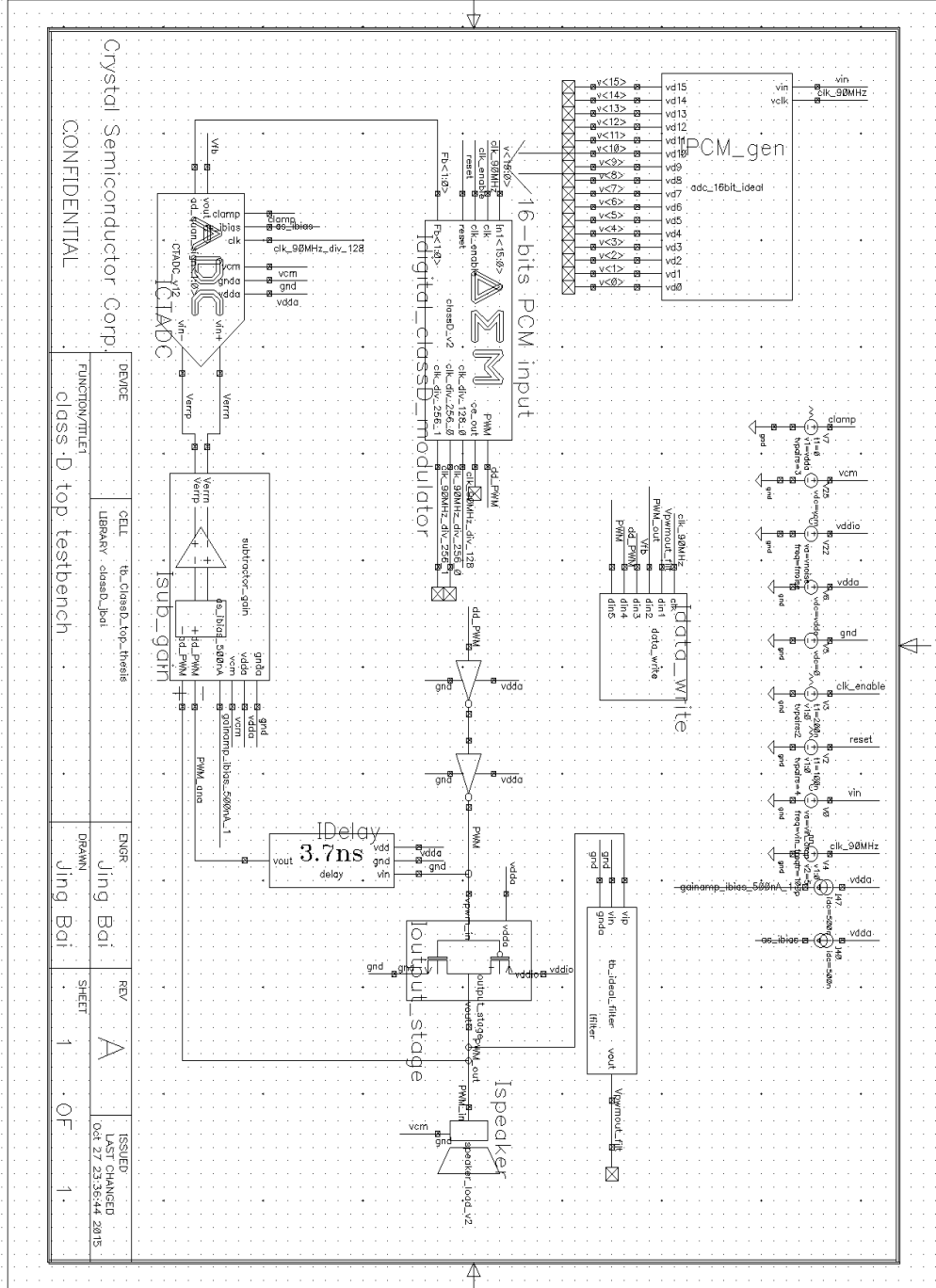


Figure 5.1: Class D amplifier system top-level schematic with test bench.

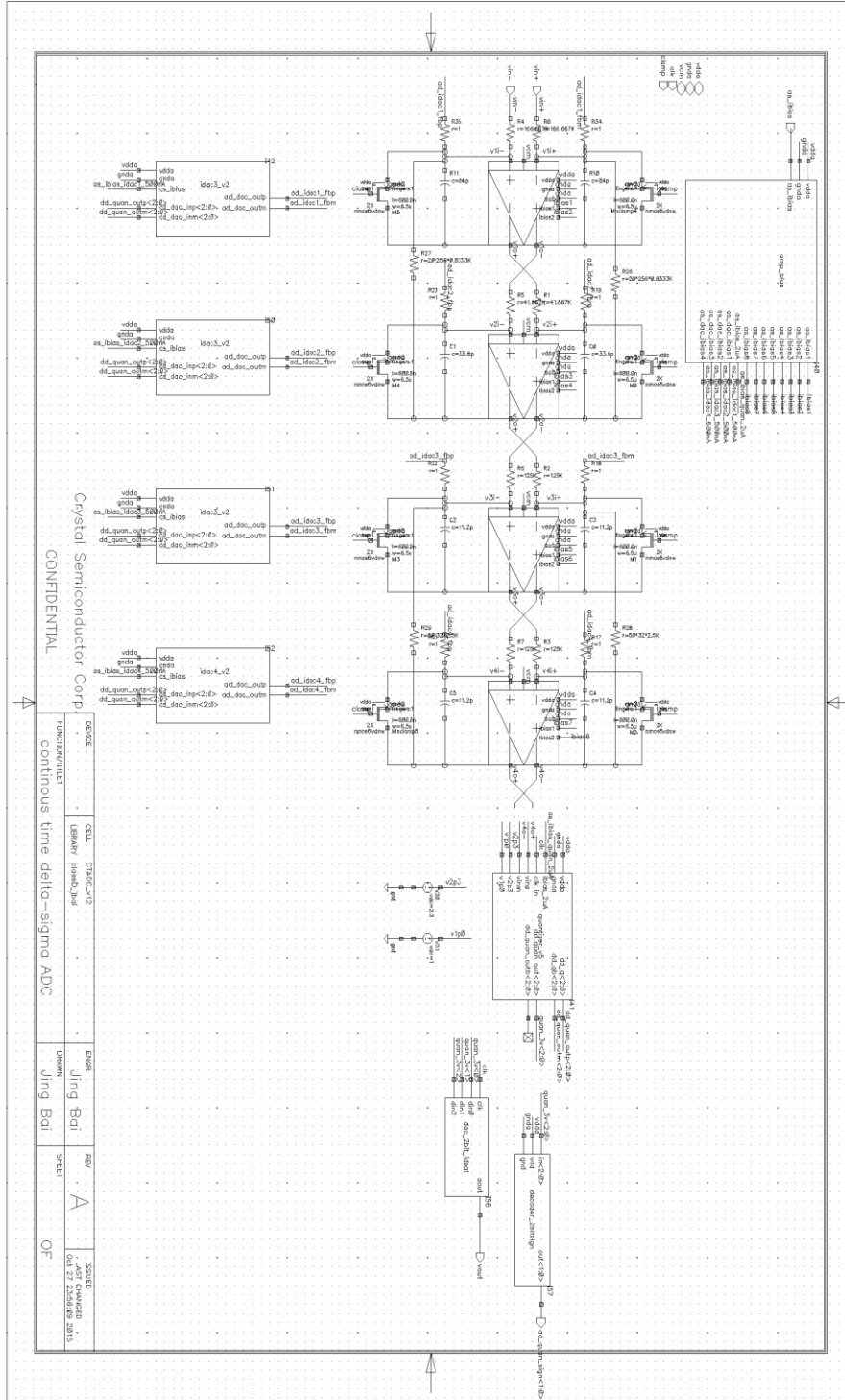


Figure 5.2: Continuous-time delta-sigma ADC top schematic.

5.2.2 Integrator Amplifier

The integrator is using the folded cascode amplifier with class AB stage amplifier as its core amplifier. Compared with other single stage amplifiers, a fully-differential folded cascode amplifier with common-mode feedback (CMFB) architecture gives relative high gain, large bandwidth and larger input and output swing. A class AB stage with zero cancellation can provide larger current to drive the active RC circuit. Figure 5-3 shows the top-level schematic of the amplifier which includes the folded-cascode amplifier with class AB stage, a CMFB amplifier, a common-mode resistor divider and a current bias block. Figure 5-4 shows the schematic of the folded-cascode amplifier. Figure 5-5 shows the schematic of the CMFB amplifier and figure 5-6 shows the current reference schematic. CMFB amplifier cascode devices must be well matched with the core amplifier devices in the layout.

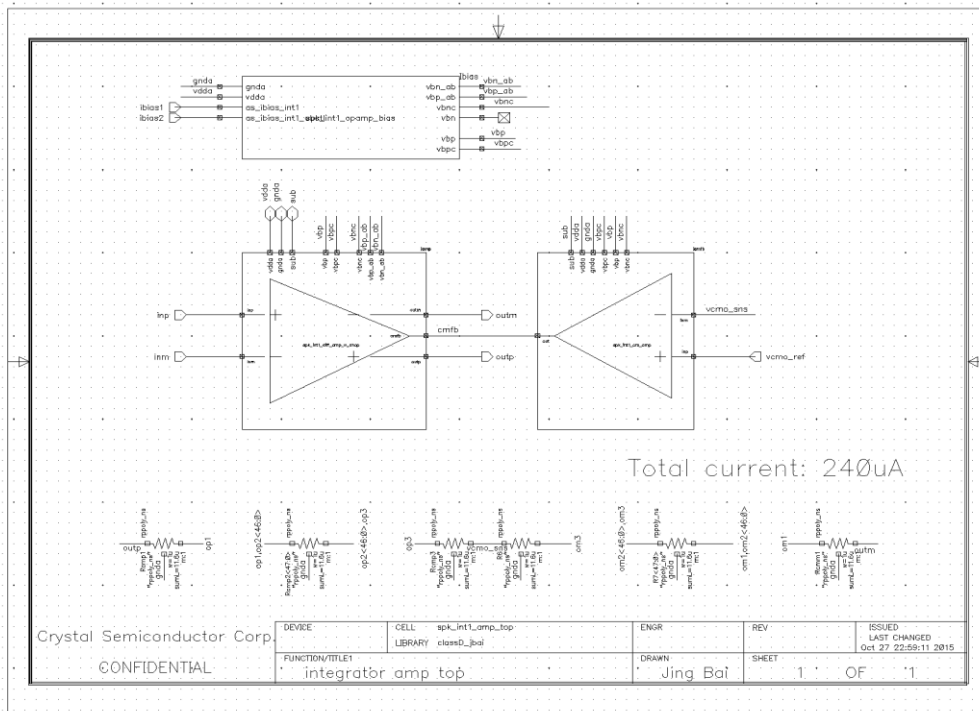


Figure 5.3: Integrator amplifier top schematic.

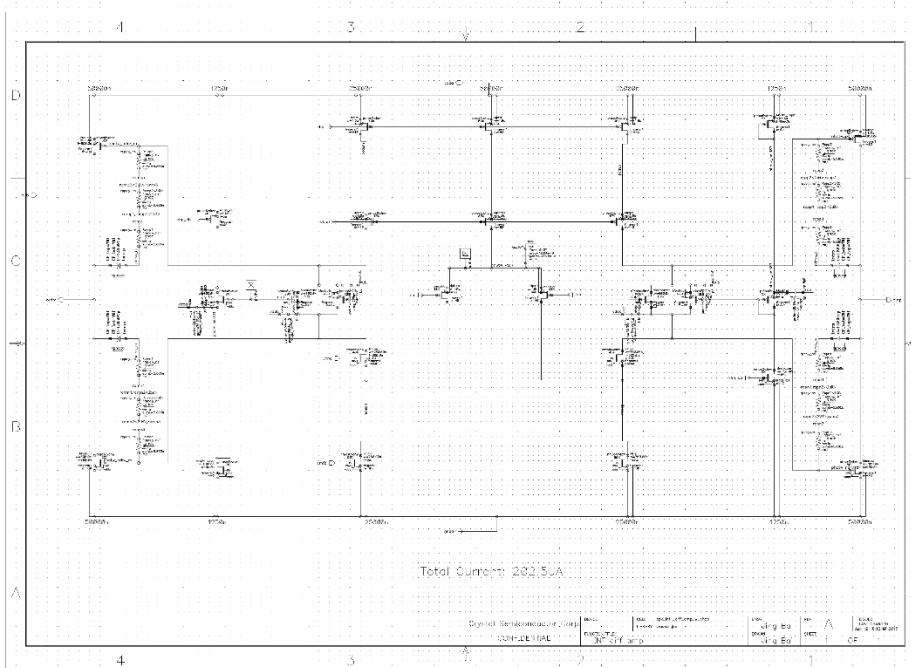


Figure 5.4: Folded-cascode class AB amplifier schematic.

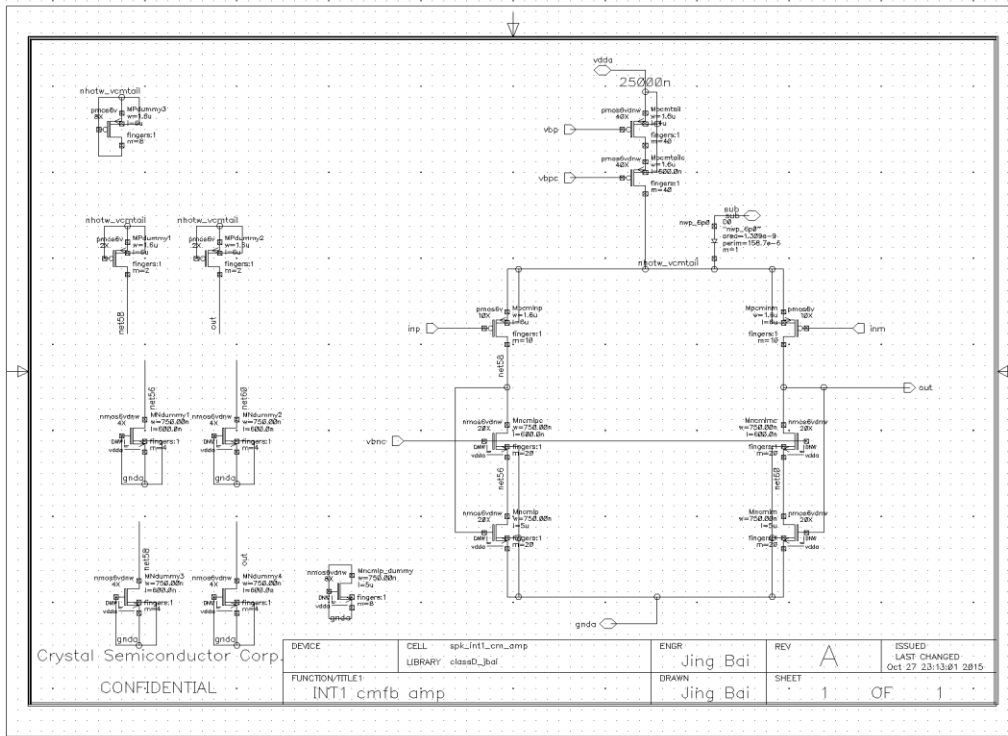


Figure 5.4: CMFB amplifier schematic.

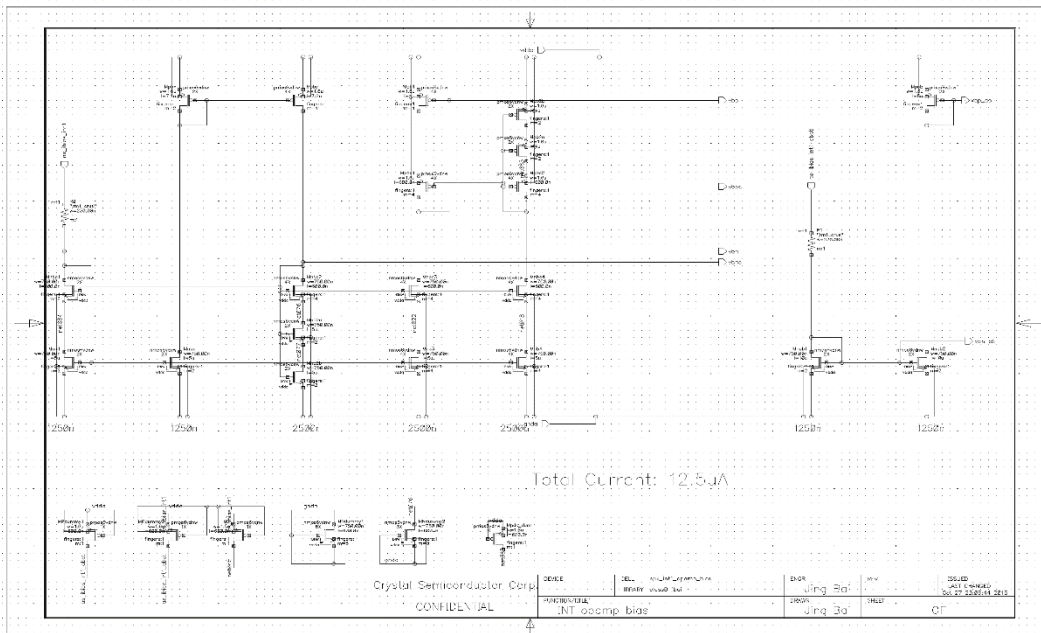


Figure 5.5: amplifier bias schematic.

Figure 5-7 shows the open loop response of the integrator amplifier. DC gain of the amplifier above 80dB is enough for the transfer function to behave like an integrator within the audio band from 20Hz to 20KHz. UGB of the amplifier is about 5MHz. Phase margin over the corners are all close to 90 degree. Figure 5-8 shows the closed-loop response of the integrator. The integrator gain over corners is above 30dB and relatively flat within the audio band. The gain is important for the loop response and flatness is important for constant behavioral within the audio band.

As mentioned in Chapter 3, to meet the system noise budget, the total input referred noise of the CTADC should be smaller than 35uVrms. Table 5-1 shows the first integrator noise simulation results. As it shown in the table, the total noise of the first integrator including input resistors, feedback resistors and amplifier is 14uVrms. Because the same integrator is used for all four stages, the total noise of the CTADC can be calculated using the following equation:

$$P_{n_ADC} = P_{n_int1} + P_{n_int2}/A_{int1} + P_{n_int3}/(A_{int1} + A_{int2}) + P_{n_int4}/(A_{int1} + A_{int2} + A_{int3}) \quad \text{Equation 5-1}$$

Where $P_{n_int1} = V_{n_int1}^2 = 196 \times 10^{-12} V_{rms}^2$ and $A_{int1} = 35dB \approx 56$, so the total noise of the CTADC is 14.22uVrms. Quantizer noise is ignored because of large loop gain.

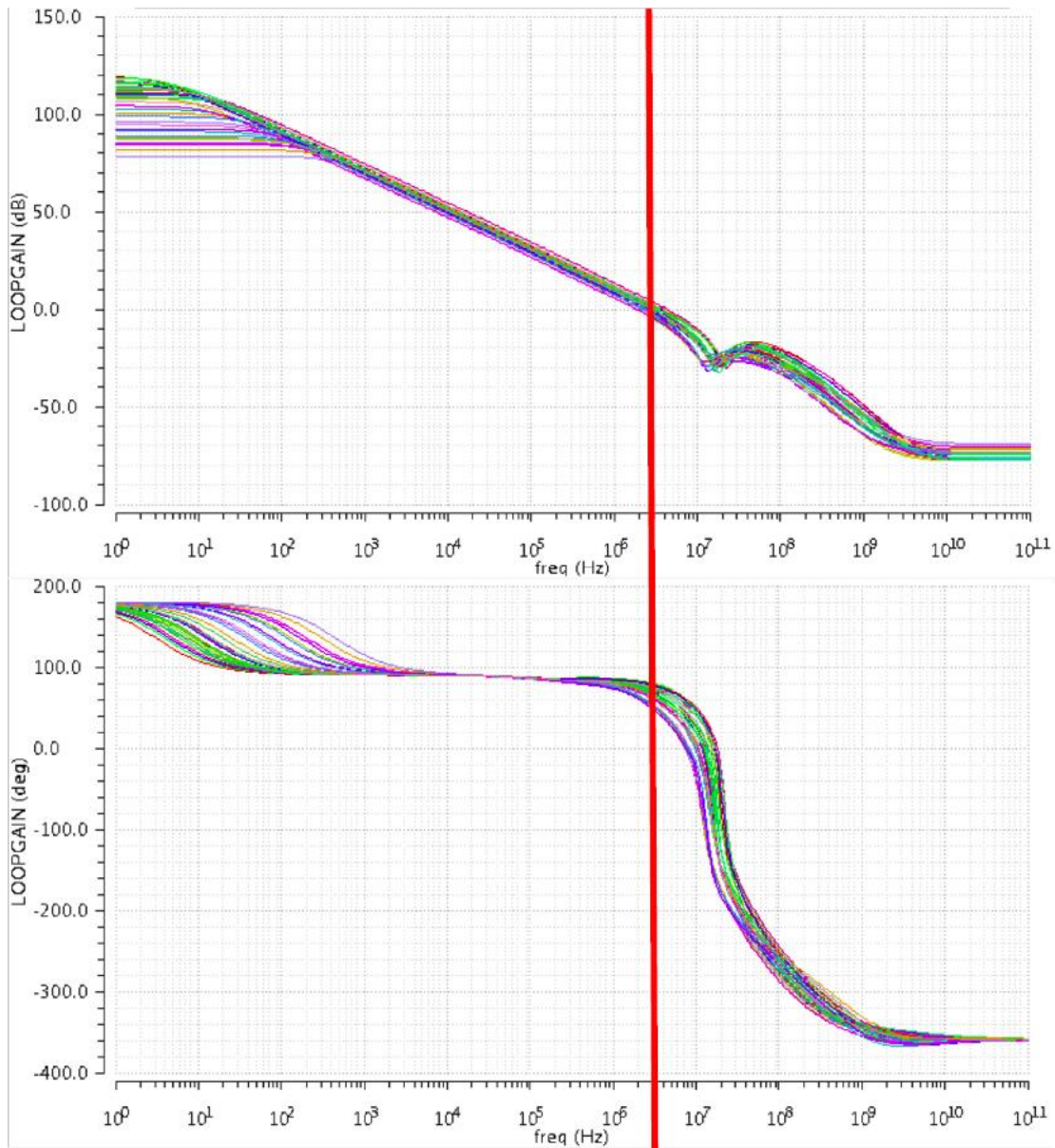


Figure 5.7: the open loop response of the integrator amplifier.

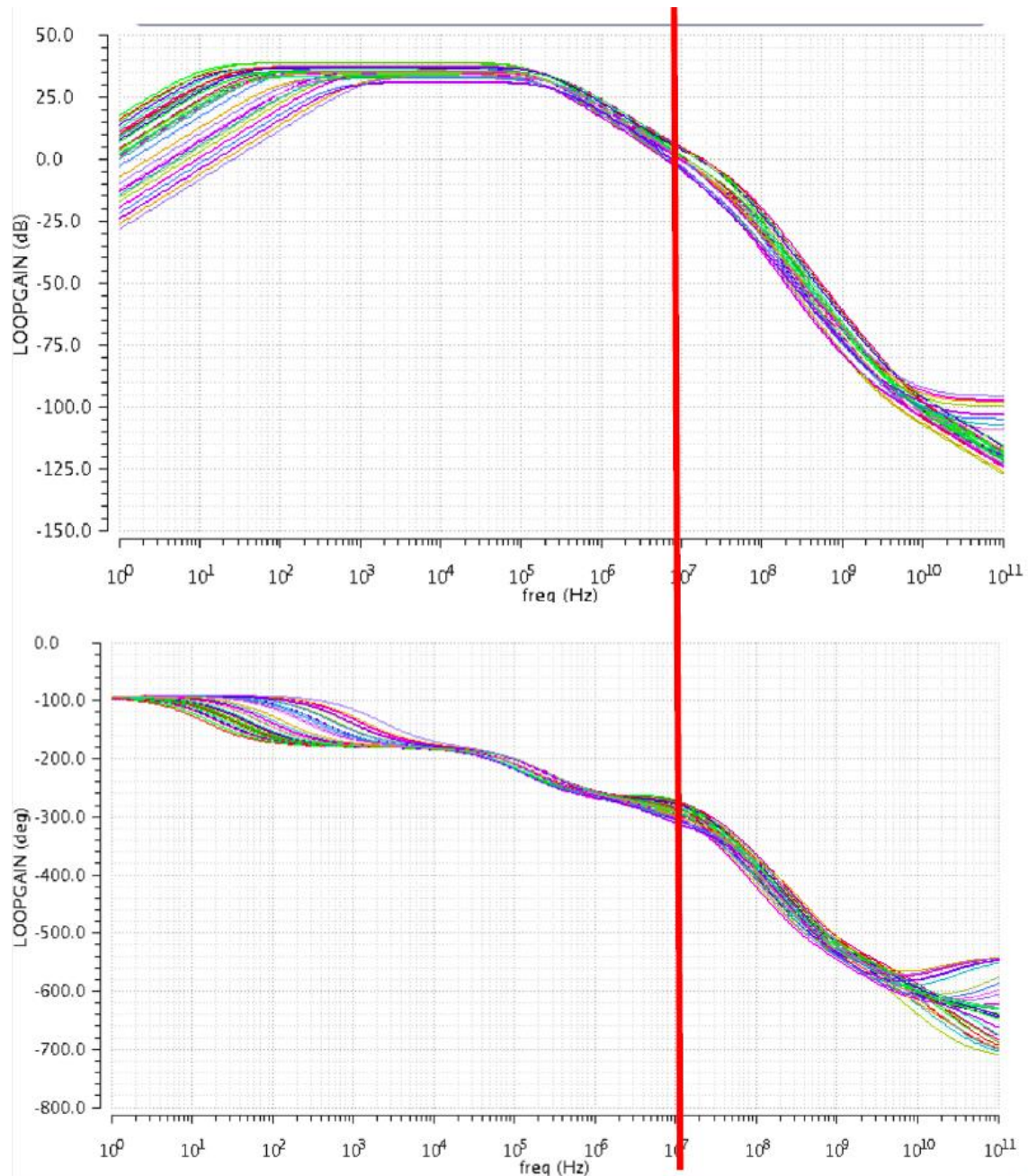


Figure 5.8: the closed-loop response of the integrator.

As mentioned in Chapter 3, to meet the system noise budget, the total input referred noise of the CTADC should be smaller than 35uVrms. Table 5-1 shows the first integrator noise simulation results. As it shown in the table, the total noise of the first

integrator including input resistors, feedback resistors and amplifier is 14uVrms. Because the same integrator is used for all four stages, the total noise of the CTADC can be calculated using the following equation:

$$P_{n_ADC} = P_{n_int1} + P_{n_int2}/A_{int1} + P_{n_int3}/(A_{int1} + A_{int2}) + P_{n_int4}/(A_{int1} + A_{int2} + A_{int3}) \quad \text{Equation 5-2}$$

Where $P_{n_int1} = V_{n_int1}^2 = 196 \times 10^{-12} V_{rms}^2$ and $A_{int1} = 35dB \approx 56$, so the total noise of the CTADC is 14.22uVrms. Quantizer noise is ignored because of large loop gain.

Table 5.1: Continuous-time delta-sigma ADC first integrator noise simulation results.

Device name	Noise type	Input referred noise
Rin	Thermal	10.5uVrms
Rfb	Thermal	2.04uVrms
Input pairs	Flicker	3.0uVrms
	Thermal	2.29uVrms
NMOS folded cascode tail	Flicker	7uVrms
	Thermal	2.33uVrms
PMOS folded cascode tail	Flicker	2.9uVrms
	Thermal	1.63uVrms
other	Flicker&Thermal	1.66uVrms
Total input referred noise	Flicker&Thermal	14uVrms

5.2.3 Current Steering DAC

A current steering DAC is used in the feedback. The current steering DAC has limited output swing, but it can drive the resistive load directly. The minimum sized switches are used to reduce the charge injection during the switching. A smaller switching voltage from 0.8V to 2.3V is used to drive the gate of the switches which can reduce the clock feedthrough. The matching of the tail devices is critical in layout to insure good current matching. Figure 5-9 shows the schematic of the current steering DAC.

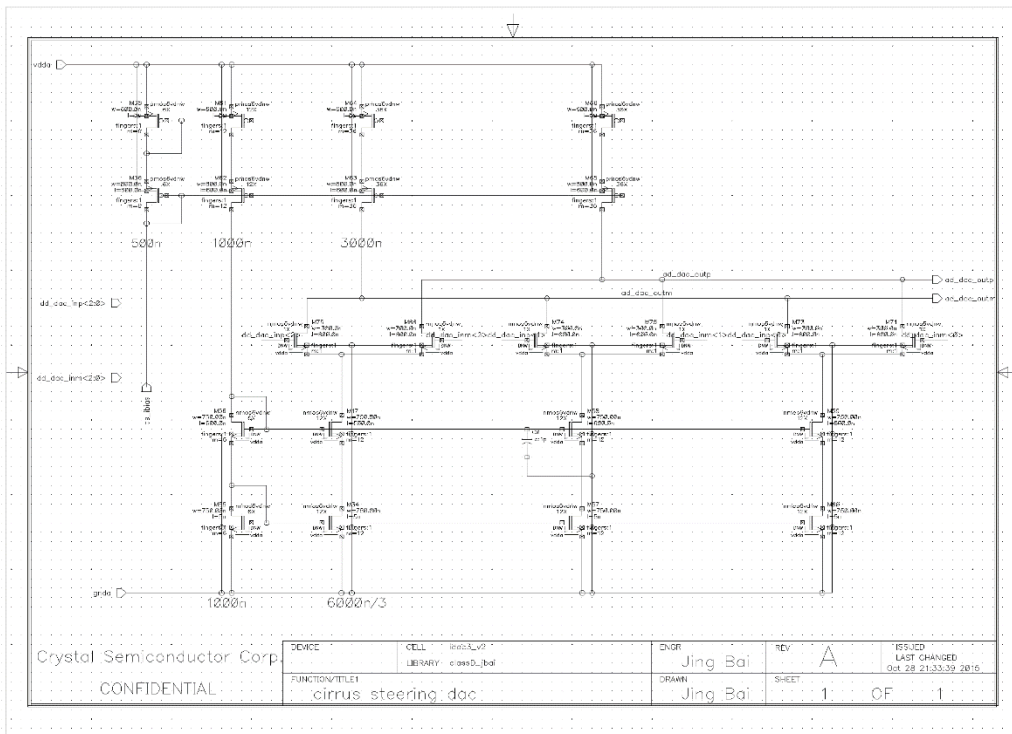


Figure 5.9: Current steering DAC schematic.

5.2.4 2-bit Quantizer

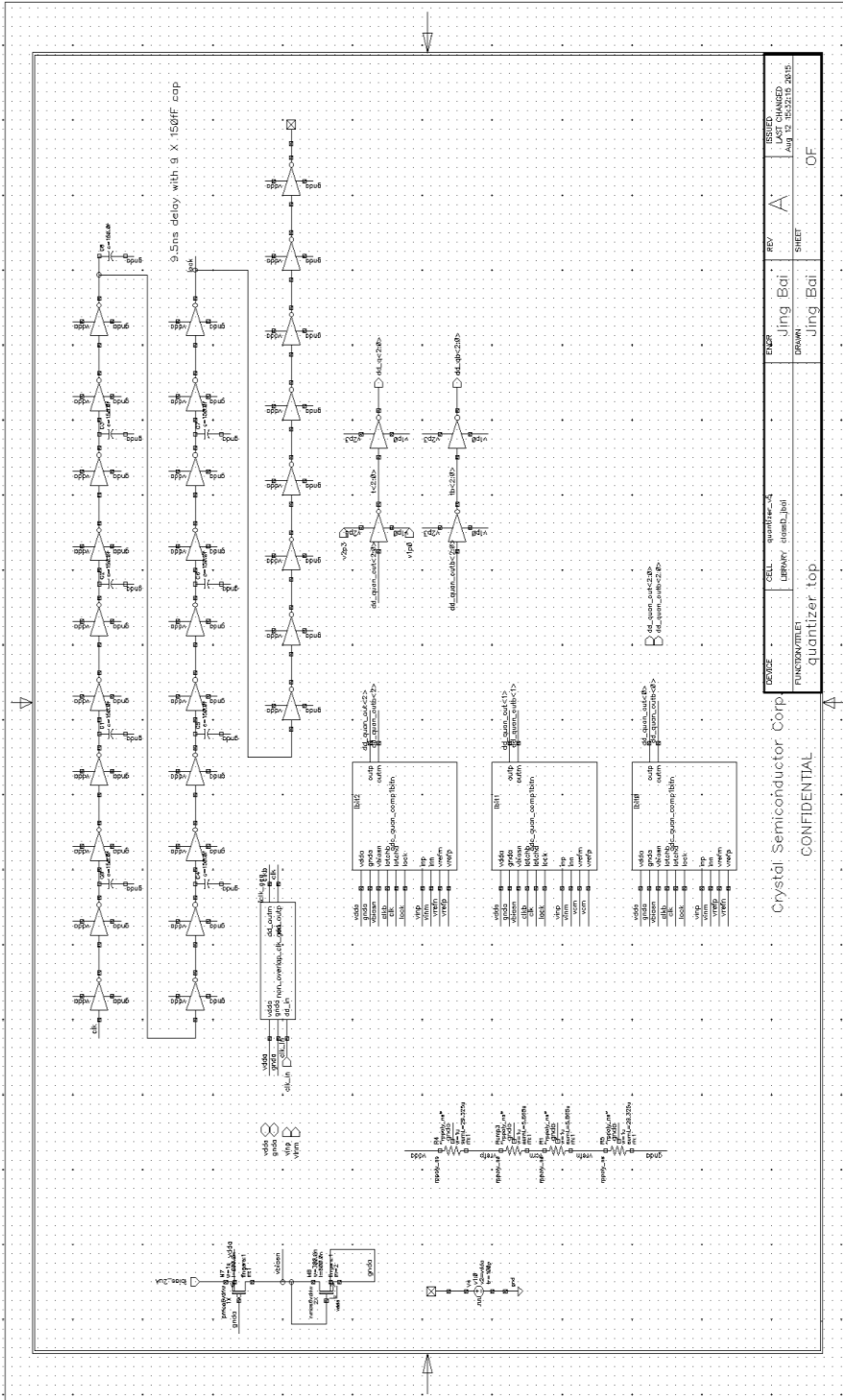
A 2-bit quantizer is used to convert the analog signal to digital signal which includes three latch comparators, non-overlap clock generator and delay blocks. Figure 5-10 shows the top-level schematic of the 2-bit quantizer. The resistor divider generates the reference voltage for quantizers. A series inverter provides about 9.5ns delay for the lock signal which locks the output of the latch.

Figure 5-11 shows the schematic of the fully differential input latch comparator with biasing current. The non-overlap clock generated by figure 5-12 controls the latch and latchb of the comparator.

The output of the three comparators is thermometer code. A decoder shown in figure 5-13 is used to convert the thermometer code to 2-bit sign code.

5.2.5 Current Bias Generation Block

Figure 5-14 shows the schematic of the current bias generation block which generates all the bias current for quantizer blocks.



REV	A	ISSUED	CHANGED
DESIGN	Jing Bai	DATE	AUG 12 16:22:19 2015
DESIGN	Jing Bai	SHEET	0F

Crystal Semiconductor Corp.
quantizer_top

CONFIDENTIAL

Figure 5.10: 2bit quantizer top-level schematic.

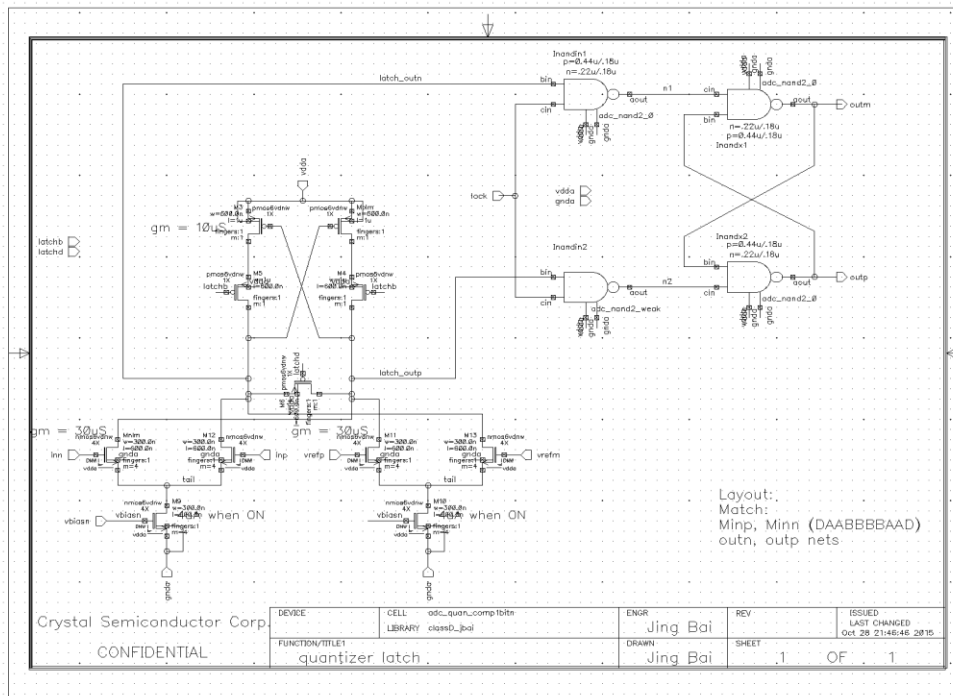


Figure 5.11: latch comparator schematic.

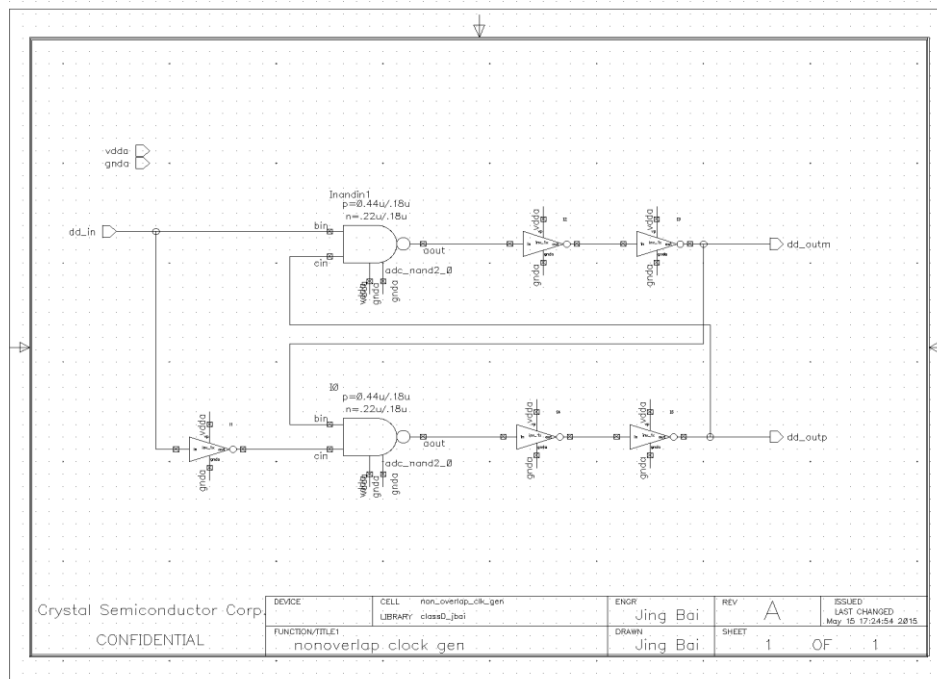


Figure 5.12: clock gen schematic.

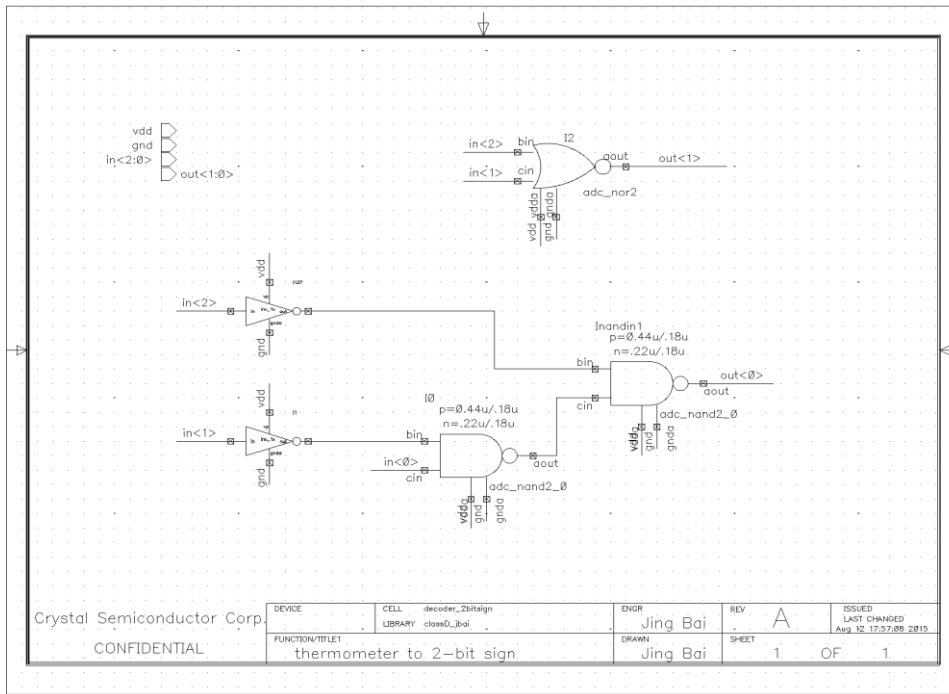


Figure 5.13: decoder schematic.

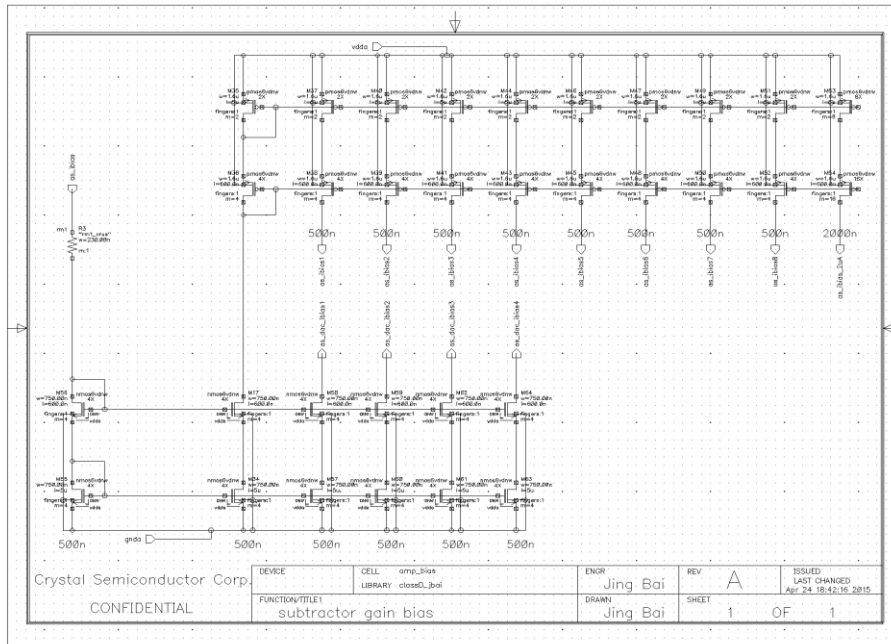


Figure 5.14: bias gen schematic.

5.3 Switching Power Stage

The switching power stage consists of dead-time generation block, PMOS/NMOS gate driver and output driver. Figure 5-15 shows the top-level schematic of the switching power stage. The sizing of the PMOS/NMOS output driver need to consider the driving strength, speed, distortion, efficiency and area. Large output driver leads to small R_{DSon} and large output current which gives better conduction loss, better linearity and larger driving strength. However, large device causes larger area and bigger parasitic capacitance which leads to high switching loss.

The power loss of the switching FET includes switching losses and conduction losses.[21]

$$P_{FET} = P_{SW} + P_{COND} \quad \text{Equation 5-3}$$

$$P_{COND} = I_{OUT}^2 \cdot R_{DSon} \cdot \frac{V_{out}}{V_{in}} \quad \text{Equation 5-4}$$

$$P_{SW} = \frac{V_{in} + I_{out}}{2} \cdot (t_r + t_f) \cdot F_{SW} \quad \text{Equation 5-5}$$

The dead-time gen block generates the non-overlap PWM gate control signal for PMOS/NMOS driver. Figure 5-16 shows the schematic of the dead-time generation block. Figure 5-17 and 5-18 shows the PMOS/NMOS gate pre-driver schematics.

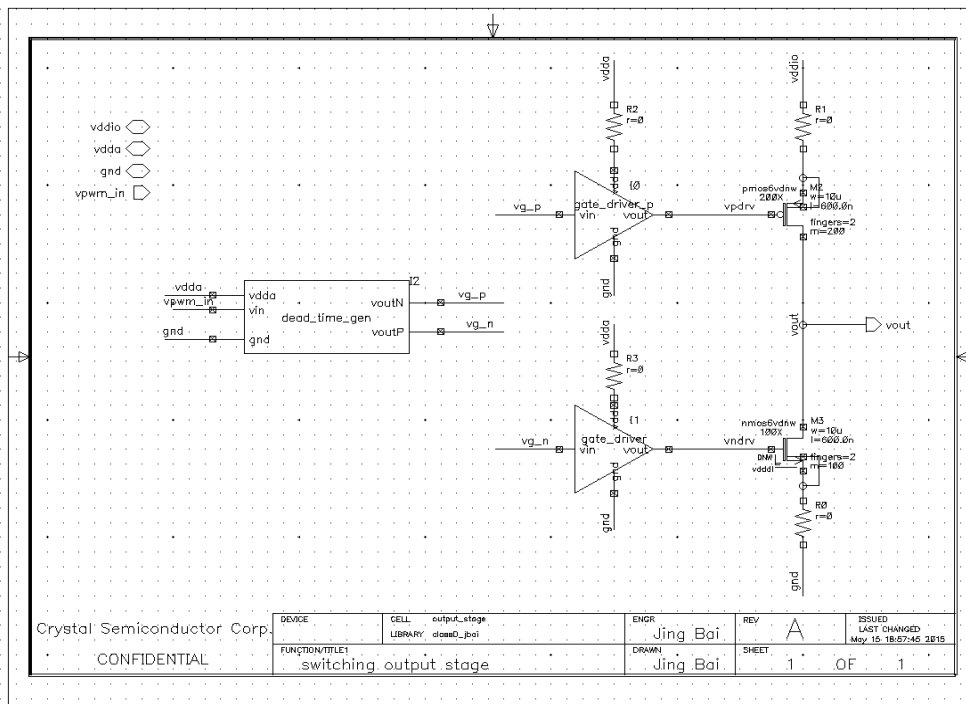


Figure 5.15: The switching power stage top schematic.

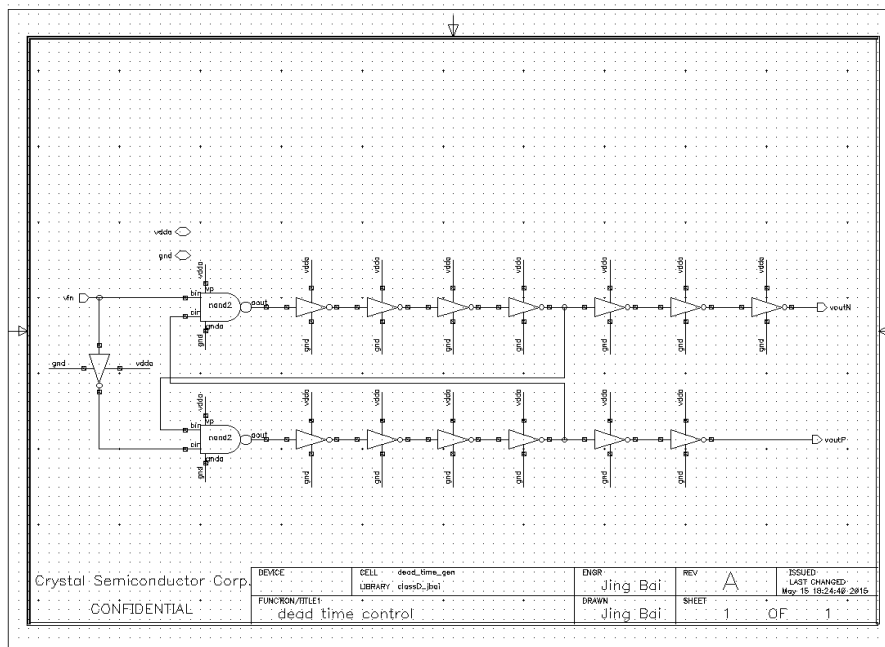


Figure 5.16: The PWM dead-time gen schematic.

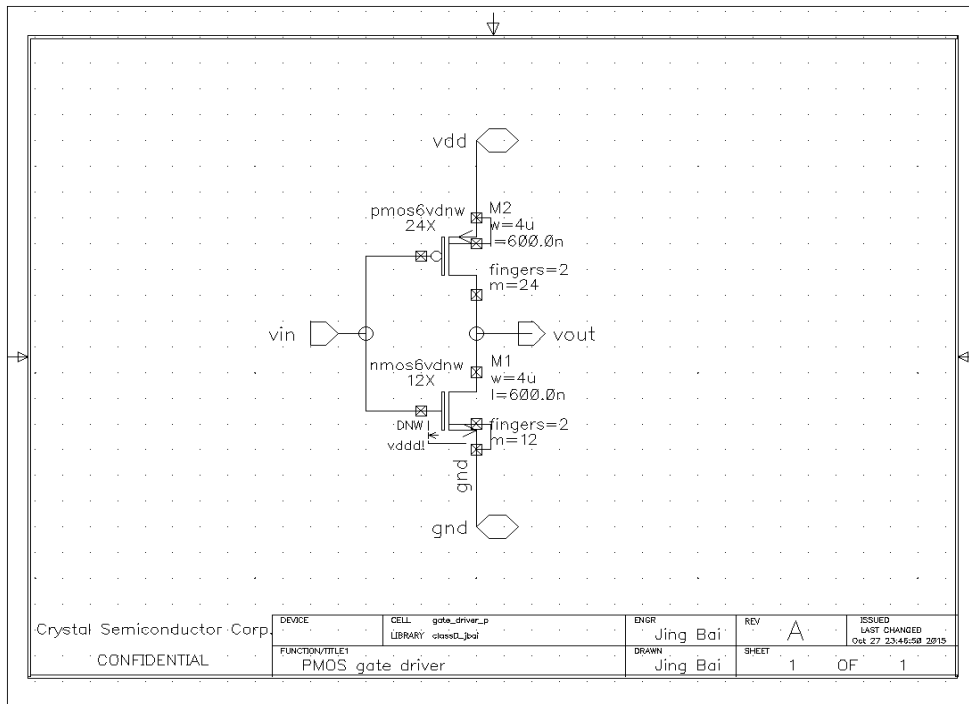


Figure 5.17: PMOS gate pre-driver schematic.

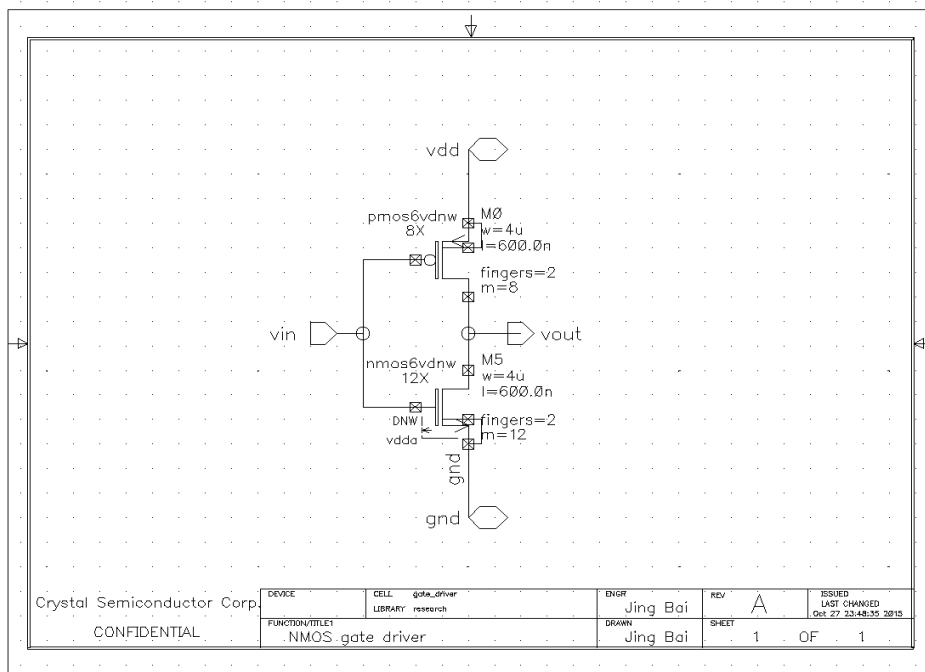


Figure 5.18: NMOS gate pre-driver schematic.

5.4 Subtractor-gain

The subtractor and gain block consists of an operational amplifier based subtractor and gain stages. The amplifier is the same amplifier used in CTADC. As mentioned in Chapter 3, the subtractor-gain block is the first stage in the feedback loop, so the input referred noise is the most critical spec for this block. Considering the dominant noise will be the subtractor-gain block, the total input referred noise of this block should be at least $<-103\text{dB}$ ($<7\mu\text{Vrms}$).

Based on the equation 3-10, 3-11 and 3-12, the resistance used in the block can be roughly calculated which is about $6\text{K}\sim 10\text{K}\Omega$. Based on equation 3-9, the total input referred noise for the analog feedback is about $5.8\mu\text{Vrms}$ (-104.7dB) loop by simulation.

5.5 Speaker Load Model

Figure 5-20 shows the speaker load model for headphone which is used in all simulations. The load resistance is about 32Ω .

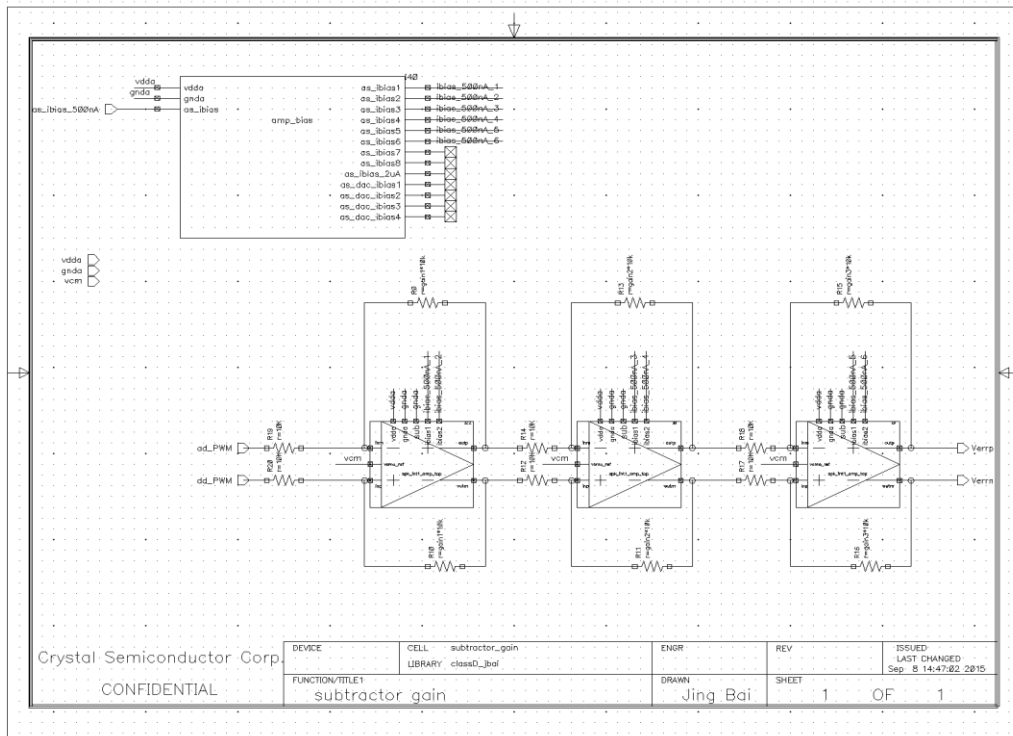


Figure 5.19: subtractor and gain schematic.

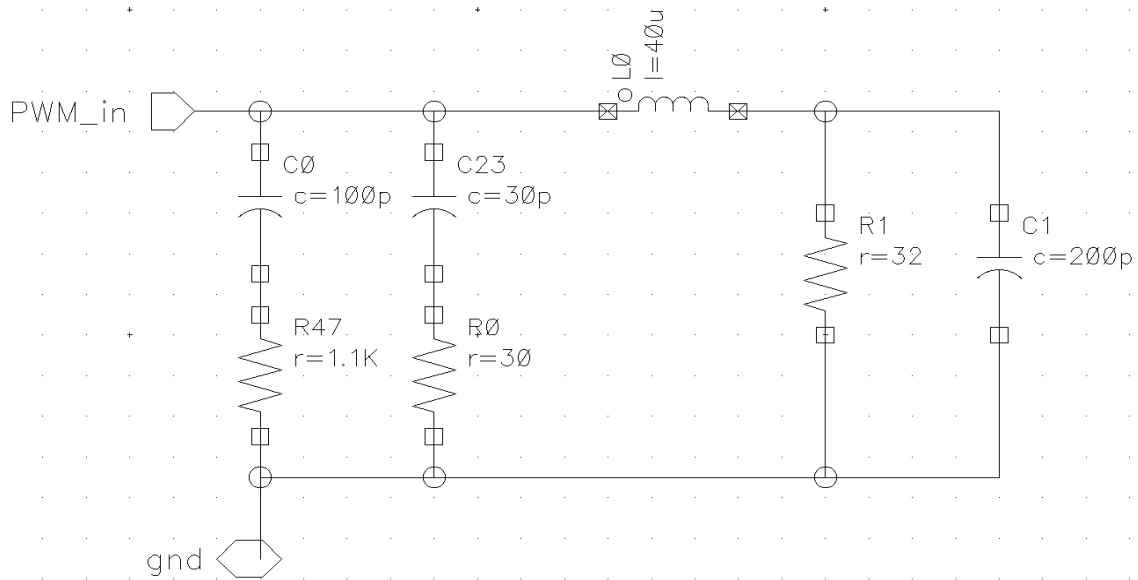


Figure 5.20: load model schematic.

5.6 Miscellanea

5.6.1 Delay

The delay block matches the delay time of the output power stage which aligns the digital PWM signal with the PWM output at the driver for better subtraction.

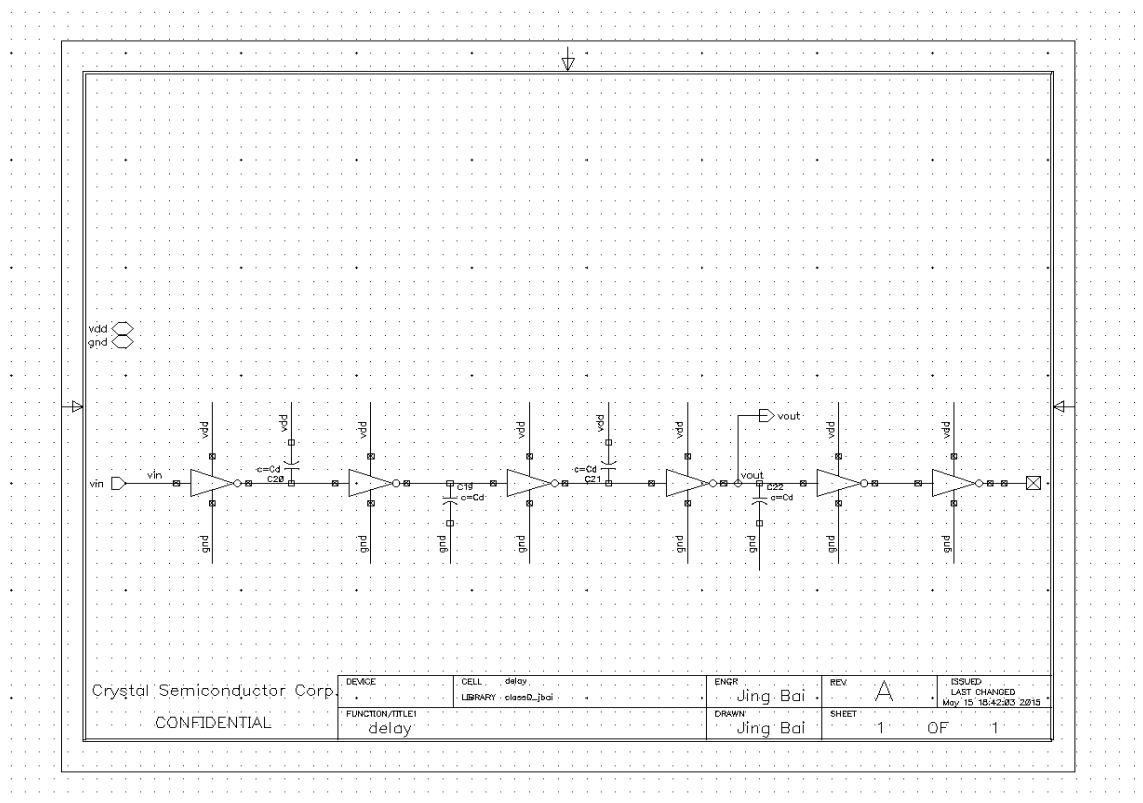


Figure 5.21: delay schematic.

5.6.2 Output Filter

The output filter is used to filter the output PWM signal for FFT data analysis.

5.6.3 PCM Generation Block

PCM generation block convert the analog signal to 16-bit PCM code. The VerilogA code is shown in the Appendix.

5.6.4 Data Write

Data write block is used to resample the data and export the data to MATLAB for FFT data analysis. The VerilogA code is shown in the Appendix.

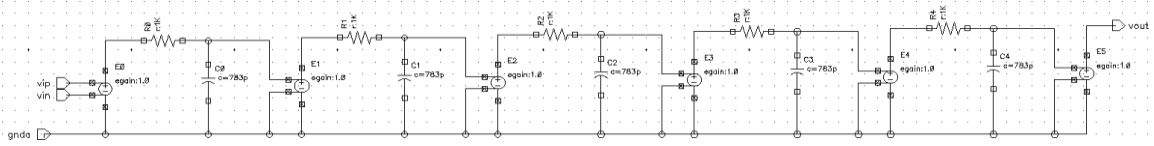


Figure 5.22: The PWM output filter schematic.

]

CHAPTER 6

FINAL RESULTS

1. System PSR vs supply noise frequency

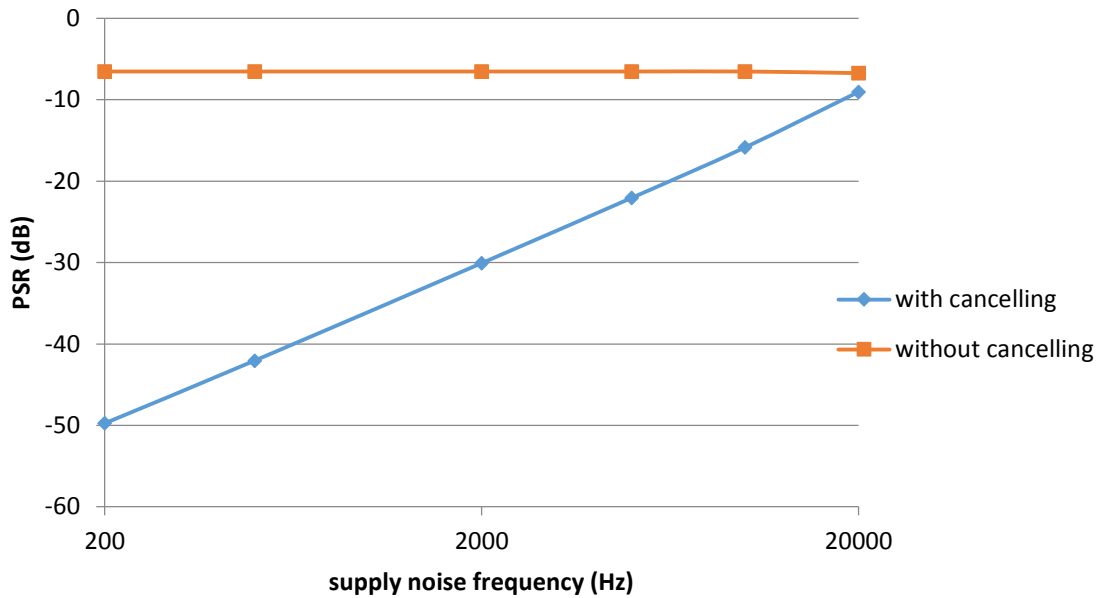


Figure 6.1: power supply rejection vs supply noise frequency

With our closed-loop cancellation scheme, the system PSR is improved across the whole audio band. At 200 Hz, the system PSR is -50dB with cancelling compared to -6.5dB without noise cancelling. Because of the feedback delay, PSR performance decreases while frequency increases.

2. Supply noise rejection over feedback delay time

Table 6.1: supply noise rejection over feedback delay time (delay from point A to B)

Feedback delay time (s)	10n	100n	800n	1.3u	1.5u	10us
Supply noise rejection (dB)	-49.763	-49.063	-48.163	-47.863	-47.163	-40.363

Supply noise rejection degrades with feedback delay time. Our feedback delay is < 800ns.

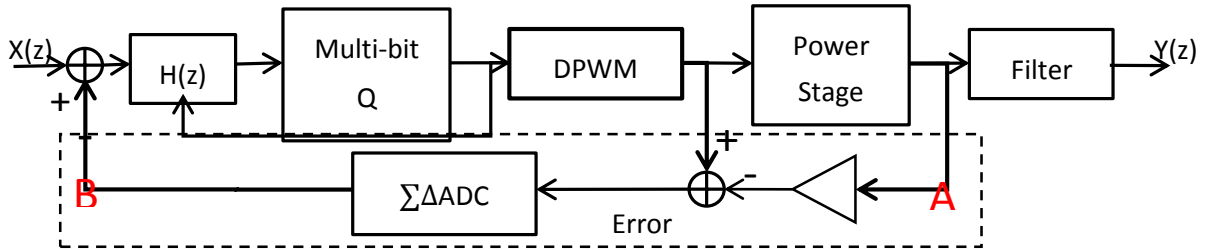


Figure 6.2: class D amplifier system showing feedback delay point A to B.

3. System THD+N

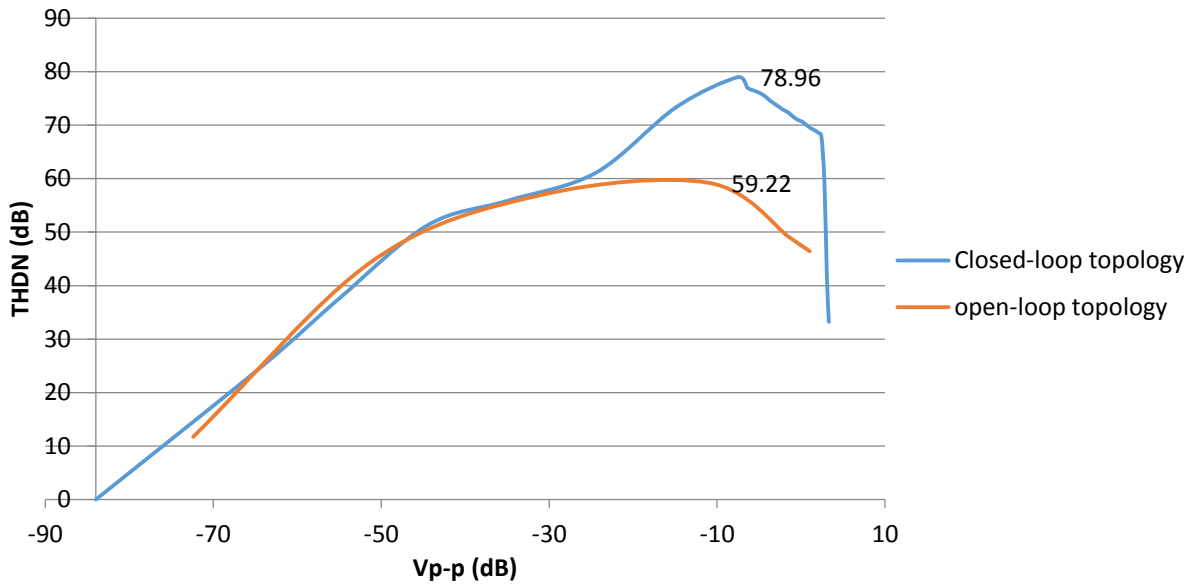


Figure 6.3: system THD+H vs input amplitude (Vpeak)

With our closed-loop cancellation scheme, the Class D system THD+N performance is improved about 20dB best case. Our best system THD+N is 78.96dB.

4. System power and efficiency

Table 6.2: System power without signal

Supply name	Block name	Average current
vdda	CTADC	566uA
	subtractor	349uA
	driver	15uA
	Delay_cell	30uA
	other	1uA
	TOTAL	961uA
vddio	Driver (quiescent current)	1.537mA
Total current		2.498mA

Table 6.3: System efficiency with 32ohm load

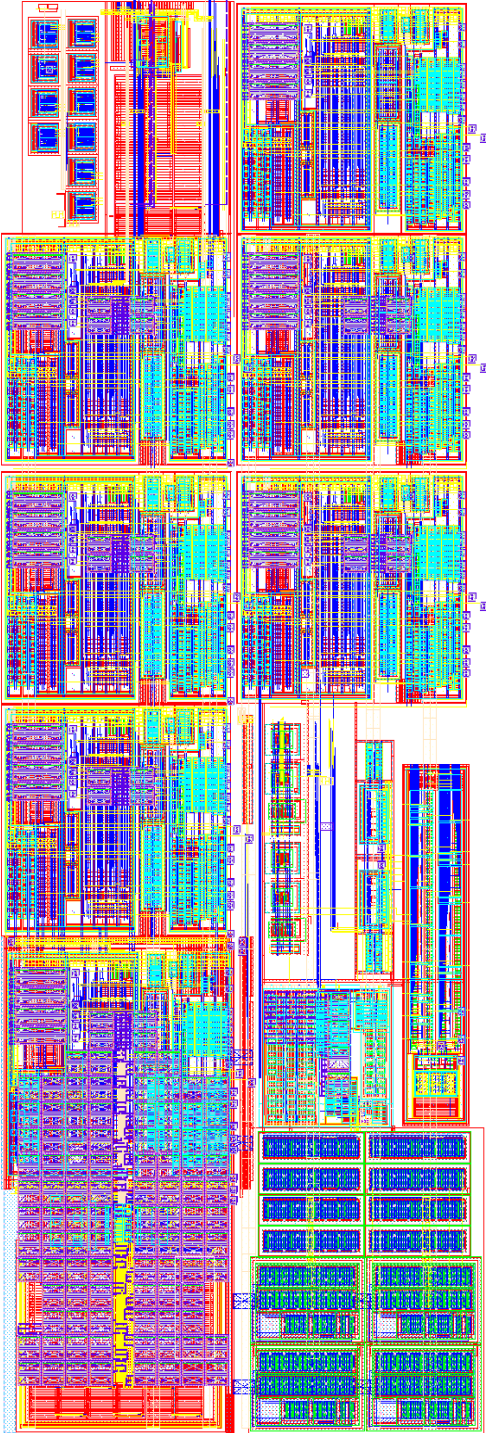
Signal amplitude Vpeak (dB)	Total power from supply (mW)	Power to load (mW)	Efficiency (%)
2.19dB	31.68	25.8	81.4
0dB	22.1	15.6	70.6
-2dB	16.5	9.86	59.8
-5dB	11.97	4.94	41.3
-12dB	8.47	0.99	11.7

5. Class D System Spec Summary

Table 6.4: Class D amplifier system spec summary.

Spec Name	Spec Detail	Min	Typ	Max	Unit
Supply	Analog supply		3		V
	Digital supply		2.5		V
temperature	Ambient temp	-40	25	125	degC
Load	Load resistance		32		ohm
	Load capacitance			200	pF
Dynamic range			81		dB
THDN	THD+N 0dB		-70	-68	dB
PSR	Power supply rejection at 200Hz		-49		dB
Noise	Input referred noise of analog feedback path		7.92		uVrms
	Output referred noise of Class D output		85.5		uVrms
Output Voltage/Power	Full scale output voltage		2.6		Vpp
	Full scale output power		27		mW
Efficiency	System efficiency at full scale (digital power excluded)		82		%

6. Layout



REFERENCES

1. Jun Honda, Jonathan Adams, "Class D Audio Amplifier Basics," Application Note AN-1071, International rectifier, Feb. 2005.
2. Pieter Stephanus Kemp, "The design of an analogue class-D audio amplifier using Z-domain methods," Thesis, University of Stellenbosch, Mar. 2012
3. Eric Gaalaas, "Class D Audio Amplifiers: What, Why, and How," Analog Dialogue 40-06, Jun. 2006.
4. Bah-Hwee Gwee, "A Novel Sampling Process and Pulse Generator for a Low Distortion Digital Pulse-Width Modulator for Digital Class D Amplifiers," IEEE, 2003.
5. Grahame Holmes, "Pulse width modulation for power converters: principles and practice", 2003. A Novel Sampling Process and Pulse Generator for a Low Distortion Digital Pulse-Width Modulator for Digital Class D Amplifiers Amplifier
6. Federico Guanziroli, "A 1 W 104 dB SNR Filter-Less Fully-Digital Open-Loop Class D Audio Amplifier with EMI Reduction," IEEE Journal of solid-state circuits, Vol. 47, No. 3, Mar. 2012.
7. Eric Gaalaas, "Integrated Stereo $\Delta\Sigma$ Class D Amplifier," IEEE Journal of solid-state circuits, Vol. 40, No. 12, Dec. 2005.
8. Marco Berkhout, "Audio at Low and High Power," IEEE, 2008.
9. Karsten Nielsen, "PEDEC-A novel pulse referenced control method for high quality digital PWM switching power amplification," Power Electronics Specialists Conference, 1998.
10. Kathleen Philips, "Power DAC: a single-chip audio DAC with a 70% efficient power stage in 0.5 μ m CMOS," IEEE International Solid-state Circuits Conference, 1999.
11. Minsheng Wang, "A 120dB Dynamic Range 400mW Class-D Speaker Driver with 4th-order PWM Modulator," IEEE Journal of Solid-State Circuits 09/2010; 45(8-45):1427 - 1435.
12. Marco Berkhout, "An Integrated 200-W Class-D Audio Amplifier," IEEE Journal of solid-state circuits, Vol. 38, No. 7, July 2003.
13. John Melanson, "Power Supply Compensation for Noise Shaped Digital Amplifier," Patent US6344811B1, Feb 5, 2002.

14. Pallab Midya, "Digital Correction of PWM Switching Amplifiers," IEEE Power Electronics Letters 07/2004; 2(2):68 - 72.
15. Chintan Trehan, "A High Performance Class-D Power Amplifier using Error Feedback Architecture", 2005
16. John Melanson, "Real time correction of a digital PWM amplifier," Patent US6373334, Apr 16, 2002.
17. Richard Schreier, "Understanding delta-sigma data converters," Published by John Wiley & Sons, Inc, Hoboken, New Jersey.
18. Chintan Trehan, "A High Performance Class-D Power Amplifier using Error Feedback Architecture," IEEE 2005.
19. <http://web.mit.edu/~bdaya/www/RPES/RPES.htm> Rapid Prototyping of Embedded System Using FPGA
20. Atlys™ Board Reference Manual, www.digilentinc.com.
21. Fairchild semiconductor, AN-6005, "synchronous buck MOSFET loss calculations."

APPENDIX A

MATLAB CODE OF THE 3RD ORDER DELTA-SIGMA MODULATOR

```

clear all;

close all;

osr = 32;

order = 3;

BW = 44.1e3;

ntf = synthesizeNTF(order,osr,0,2)

plotPZ(ntf)

f = linspace(0,0.5,1000);

z = exp(2i*pi*f);

figure;

plot(f,dbv(evalTF(ntf,z)));

ylabel('Amp (dB)');

xlabel('Normalized Freq');

sigma_H = dbv(rmsGain(ntf,0,0.5/osr));

amp=[-100:5:-10,-10:0];

[snr,amp] = simulateSNR(ntf,osr,amp,0,64);

figure;

plot(amp,snr);

[a,g,b,c]=realizeNTF(ntf,'CIFB')

Hz = 1/ntf-1

Htf = tf(Hz)

g = rmsGain(ntf,0,1/16,BW);

```

APPENDIX B

MATLAB CODE OF THE 4TH ORDER CONTINUOUS-TIME DELTA-SIGMA ADC

```

clear all;

close all;

osr = 32;

order = 4;

BW = 20e3;

ntf = synthesiseNTF(order,osr,1,2.5)

% figure;

% subplot(2,2,1);

plotPZ(ntf);

f = linspace(0,0.5,1000);

% z = exp(2i*pi*f);

% subplot(2,2,2);

% plot(f,dbv(evalTF(ntf,z)));

% ylabel('Amp (dB)');

% xlabel('Normalized Freq');

% sigma_H = dbv(rmsGain(ntf,0,0.5/osr));

% amp=[-100:5:-10,-10:0.5:0];

% [snr,amp] = simulateSNR(ntf,osr,amp,0,4);

% subplot(2,2,3);

% plot(amp,snr);

%

% [a,g,b,c]=realizeNTF(ntf,'CRFB');

```

```

% ABCD = stuffABCD(a,g,b,c);

Hz = 1/ntf-1

% Htf = tf(Hz);

% Htf_half = d2d(Htf,0.5);

% z_half = tf([1 0],1,0.5);

% Htf1 = Htf_half*z_half;

Hs = d2c(Hz)

Htf = 1/(1+Hs)

figure;

bode(Htf)

den = conv([1 0.9 0.251],[1 0.3317 0.3751]);

g1 = 0.001114

g2 = 0.007147

a4 = den(2)

a3 = den(3)-g1-g2

a2 = den(4)-a4*g1

a1 = den(5)-g1*g2-g1*a3

```

APPENDIX C

DIGITAL CLASS D MODULATOR VERILOG CODE GENERATED BY MATLAB

```
// -----  
// File Name:hdlsrc\classD.v  
// Created: 2015-08-27 18:11:39  
// Generated by MATLAB 7.5 and Simulink HDL Coder 1.2  
// Module: classD  
// Source Path: PWM_v4_hdl/classD  
// Hierarchy Level: 0  
// -----
```

```
`timescale 1 ns / 1 ns
```

```
module classD
```

```
(  
    clk,  
    reset,  
    clk_enable,  
    In1,  
    Fb,  
    ce_out,  
    PWM  
);
```



```

input  clk;

input  reset;

input  clk_enable;

input  signed [15:0] In1; // sfix16_En15

input  signed [1:0] Fb; // sfix2_En4

output ce_out;

output PWM;

parameter signed [1:0] Gain1_gainparam = 2'b01; // sfix2

parameter signed [7:0] Gain7_gainparam = 8'b01000000; // int8

wire enb;

wire enb_128_128_0;

wire enb_1_128_0;

wire enb_1_1_1;

wire signed [6:0] triangle_out1; // sfix7

wire signed [2:0] Gain1_out1; // sfix3_En5

wire signed [1:0] Constant1_out1; // sfix2_En5

wire signed [15:0] Sum1_out1; // sfix16_En15

wire signed [15:0] Sum_out1; // sfix16_En15

wire signed [6:0] SDM_core_out1; // sfix7_En6

wire signed [6:0] Gain7_out1; // sfix7

reg signed [6:0] Rate_Transition_out1; // sfix7

```

```

wire Comparator_out1;

real TmpGroundAtScope1Inport4_out1; // double
real TmpGroundAtScope1Inport3_out1; // double
real TmpGroundAtScope1Inport2_out1; // double
real TmpGroundAtScope1Inport1_out1; // double

wire signed [3:0] Sum1_out1_tmp; // sfix4_En5

wire signed [3:0] add_cast; // sfix4_En5

wire signed [3:0] add_cast_1; // sfix4_En5

wire signed [4:0] add_temp; // sfix5_En5

wire signed [16:0] Sum_out1_tmp; // sfix17_En15

wire signed [16:0] sub_cast; // sfix17_En15

wire signed [16:0] sub_cast_1; // sfix17_En15

wire signed [17:0] sub_temp; // sfix18_En15

Timing_Controller u_Timing_Controller (.clk(clk),
                                        .reset(reset),
                                        .clk_enable(clk_enable),
                                        .enb(enb),
                                        .enb_1_1_1(enb_1_1_1),
                                        .enb_1_128_0(enb_1_128_0),
                                        .enb_128_128_0(enb_128_128_0)
                                        );

```

```

assign ce_out = enb_1_1_1;

triangle u_triangle (.clk(clk),
                    .reset(reset),
                    .enb(enb),
                    .Out1(triangle_out1) // sfix7
                    );

assign Gain1_out1 = $signed({Fb[1:0], 1'b0});

assign Constant1_out1 = 2'b01;

assign add_cast = $signed({1{Gain1_out1[2]}, Gain1_out1});
assign add_cast_1 = $signed({2{Constant1_out1[1]}, Constant1_out1});
assign add_temp = add_cast + add_cast_1;
assign Sum1_out1_tmp = (add_temp[4] == 1'b0 & add_temp[3] != 1'b0) ? 4'b0111 :
    (add_temp[4] == 1'b1 && add_temp[3] != 1'b1) ? 4'b1000 : add_temp[3:0];

assign Sum1_out1 = $signed({Sum1_out1_tmp, 10'b0000000000});

assign sub_cast = $signed({1{In1[15]}, In1});
assign sub_cast_1 = $signed({1{Sum1_out1[15]}, Sum1_out1});
assign sub_temp = sub_cast - sub_cast_1;

```

```

assign Sum_out1_tmp = (sub_temp[17] == 1'b0 & sub_temp[16] != 1'b0) ?
17'b011111111111111111 :
(sub_temp[17] == 1'b1 && sub_temp[16] != 1'b1) ? 17'b1000000000000000 :
sub_temp[16:0];

```

```

assign Sum_out1 = (Sum_out1_tmp[16] == 1'b0 & Sum_out1_tmp[15] != 1'b0) ?
16'b01111111111111111 :
(Sum_out1_tmp[16] == 1'b1 && Sum_out1_tmp[15] != 1'b1) ?
16'b1000000000000000 : Sum_out1_tmp[15:0];

```

```

SDM_core u_SDM_core (.clk(clk),
.reset(reset),
.enb(enb),
.enb_1_128_0(enb_1_128_0),
.In1(Sum_out1), // sfix16_En15
.Out1(SDM_core_out1) // sfix7_En6
);

```

```

assign Gain7_out1 = SDM_core_out1;

```

```

always @ (posedge clk or posedge reset)

```

```

begin: Rate_Transition_process

```

```

if (reset == 1'b1) begin

```

```

    Rate_Transition_out1 <= 0;
end
else begin
    if (enb_1_128_0 == 1'b1) begin
        Rate_Transition_out1 <= Gain7_out1;
    end
end
end // Rate_Transition_process

Comparator u_Comparator (.clk(clk),
    .reset(reset),
    .enb(enb),
    .Tri_rsvd(triangle_out1), // sfix7
    .SDin(Rate_Transition_out1), // sfix7
    .v(Comparator_out1)
);

assign PWM = Comparator_out1;

initial
begin
    TmpGroundAtScope1Inport4_out1 = 0.0000000000000000E+000;
end

```

```
initial
begin
  TmpGroundAtScope1Inport3_out1 = 0.0000000000000000E+000;
end
```

```
initial
begin
  TmpGroundAtScope1Inport2_out1 = 0.0000000000000000E+000;
end
```

```
initial
begin
  TmpGroundAtScope1Inport1_out1 = 0.0000000000000000E+000;
end
```

```
endmodule // classD
```

APPENDIX D
PCM GEN VERILOGA CODE

```
//systemVerilog HDL for "classD_jbai", "data_write" "systemVerilog"
```

```
module data_write (din1,din2,din3,din4,din5,clk);
```

```
input clk;
```

```
input real din1,din2,din3,din4,din5;
```

```
real out_data1,out_data2,out_data3,out_data4,out_data5;
```

```
integer f1,f2,f3,f4,f5;
```

```
initial begin
```

```
    f1 = $fopen("/home/jbai/myfolder/Vfb_filt.txt","w");
```

```
    f2 = $fopen("/home/jbai/myfolder/Vfb.txt","w");
```

```
    f3 = $fopen("/home/jbai/myfolder/VPWM_out.txt","w");
```

```
    f4 = $fopen("/home/jbai/myfolder/VPWM_out_filt.txt","w");
```

```
    f5 = $fopen("/home/jbai/myfolder/Verr_filt.txt","w");
```

```
end
```

```
assign out_data1 = din1;
```

```
assign out_data2 = din2;
```

```
assign out_data3 = din3;
```

```
assign out_data4 = din4;
```

```
assign out_data5 = din5;
```



```
always @(posedge clk) begin

    $fwrite(f1,"%1.7f\n",out_data1);

    $fwrite(f2,"%1.7f\n",out_data2);

    $fwrite(f3,"%1.7f\n",out_data3);

    $fwrite(f4,"%1.7f\n",out_data4);

    $fwrite(f5,"%1.7f\n",out_data5);

end

endmodule
```

APPENDIX E
DATA WRITE VERILOGA CODE

```

`include "discipline.h"

`include "constants.h"

// $Date: 2015/07/22

// $Revision: 1.1 $

//

//

// Based on the OVI Verilog-A Language Reference Manual, version 1.0 1996

//

//

//-----

// adc_16bit_ideal

//

// - Ideal 16 bit analog to digital converter

//

// vin:      [V,A]

// vclk: [V,A]

// vd0..vd7:  data output terminals [V,A]

//

// INSTANCE parameters

//  tdel, trise, tfall = {usual} [s]

//  vlogic_high = [V]

//  vlogic_low  = [V]

```

```

// vtrans_clk = clk high to low transition voltage [V]
// vref      = voltage that voltage is done with respect to [V]
//
// MODEL parameters
// {none}
//
// This model is ideal in the sense that there is no mismatch modeled.
//

module adc_16bit_ideal(vd15 ,vd14, vd13, vd12, vd11, vd10, vd9, vd8, vd7, vd6, vd5,
vd4, vd3, vd2, vd1, vd0, vin, vclk);
electrical vd15, vd14, vd13, vd12, vd11, vd10, vd9, vd8, vd7, vd6, vd5, vd4, vd3, vd2,
vd1, vd0, vin, vclk;
parameter real trise = 0 from [0:inf);
parameter real tfall = 0 from [0:inf);
parameter real tdel = 0 from [0:inf);
parameter real vlogic_high = 2.5;
parameter real vlogic_low  = 0;
parameter real vtrans_clk   = 0.5;
parameter real vref        = 1.0;

`define NUM_ADC_BITS    16

    real unconverted;

```

```

real halfref;

real vd[0:`NUM_ADC_BITS-1];

integer i;

analog begin

    @ ( initial_step ) begin

        halfref = vref / 2;

    end

    @ (cross(V(vclk) - vtrans_clk, 1)) begin

        unconverted = V(vin);

        for (i = (`NUM_ADC_BITS-1); i >= 0 ; i = i - 1) begin

            vd[i] = 0;

            if (i == 15) begin

                if (unconverted > halfref) begin

                    vd[i] = vlogic_low;

                    unconverted = unconverted - halfref;

                end else begin

                    vd[i] = vlogic_high;

                end

            end

            end else begin

```

```

        if (unconverted > halfref) begin
            vd[i] = vlogic_high;
            unconverted = unconverted - halfref;
        end else begin
            vd[i] = vlogic_low;
        end
    end

    unconverted = unconverted * 2;

end

end

//

// assign the outputs

//

V(vd15) <+ transition( vd[15], tdel, trise, tfall );
V(vd14) <+ transition( vd[14], tdel, trise, tfall );
V(vd13) <+ transition( vd[13], tdel, trise, tfall );
V(vd12) <+ transition( vd[12], tdel, trise, tfall );
V(vd11) <+ transition( vd[11], tdel, trise, tfall );
V(vd10) <+ transition( vd[10], tdel, trise, tfall );
V(vd9) <+ transition( vd[9], tdel, trise, tfall );
V(vd8) <+ transition( vd[8], tdel, trise, tfall );
V(vd7) <+ transition( vd[7], tdel, trise, tfall );

```

```
V(vd6) <+ transition( vd[6], tdel, trise, tfall );  
V(vd5) <+ transition( vd[5], tdel, trise, tfall );  
V(vd4) <+ transition( vd[4], tdel, trise, tfall );  
V(vd3) <+ transition( vd[3], tdel, trise, tfall );  
V(vd2) <+ transition( vd[2], tdel, trise, tfall );  
V(vd1) <+ transition( vd[1], tdel, trise, tfall );  
V(vd0) <+ transition( vd[0], tdel, trise, tfall );
```

```
`undef NUM_ADC_BITS
```

```
end
```

```
endmodule
```