TU Delft

Integrated Readout Circuit for Cross-Correlation Based Ultrasonic Ranging

Master of Science Thesis

For the degree of Master of Science in Electrical Engineering at Delft University of Technology

N. Radeljic-Jakic

December 9, 2015

Faculty of Electrical Engineering Mathematics and Computer Science (EEMCS) · Delft University of Technology

The work in this thesis was supported by Texas Instruments. Their cooperation is hereby gratefully acknowledged.

Abstract

Distance measurement using ultrasonic waves is employed in a wide range of industrial applications. In this thesis our main goal is to investigate the possibility of processing ultrasonic signals with a $\Sigma\Delta$-modulator and to process its bitstream output signal with the cross-correlation technique, while being independent of the transducer that is used. A compact chip has been designed in a 0.5 μm CMOS process for the readout of the transducer. We present simulation results that show that the system is able to accurately estimate the distance to a target when low-Q transducers are used. Furthermore, we present measurement results in which we show that the system is able to determine the zero-crossings of the received echo with a resolution of 0.169 mm.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my supervisor Dr.ir. Michiel A.P. Pertijs, for giving me an opportunity to be part of his research group. I appreciate the guidance and input that he has given me throughout the entirety of my M.Sc. thesis project.

I would especially like to thank Dr.ir. Wilko Kindt, my daily supervisor at Texas Instruments, for giving me an opportunity to do my M.Sc. thesis project within the company. I am forever grateful for letting me be a part of his group. He always made time to answer any questions I had and helped me understand various aspects of analog circuit design.

Both Dr.ir. Michiel A.P. Pertijs and Dr.ir. Wilko Kindt have been role models for me.

I would also like to thank Dr.ir. Frerik Witte and the rest of the people that I've had the pleasure to work with at Texas Instruments, for their valuable inputs and the great atmosphere within the company.

Delft, University of Technology — N. Radeljic-Jakic
December 9, 2015

Chapter 1

Introduction

Ultrasonic range measurement is employed in a wide range of industrial applications and consumer products. These applications include robots, park assist systems, sonar, fluid level measurement, mapping and many other forms of object detection. Between many range measurement techniques existing today, ultrasonic range measurement distinguishes itself by providing high resolution at a low cost. Furthermore, ultrasonic transducers are insensitive to dirt and dust as opposed to various different types of sensors.

In this thesis, we will address the implementation of a system that allows us to use ultrasonic waves originating from a piezoelectric transducer to measure distance. Many techniques have already been proposed with which ultrasonic distance measurement is possible. In [2],[3] an accuracy of one-tenth wavelength (λ) is achieved with the cross-correlation method. In [4] a one-hundredth λ accuracy is achieved by combining cross-correlation with sine-fitting and a 12-bit ADC. In [5] cross correlation is performed on the bitstream output of a 7^{th} order $\Sigma\Delta$-modulator and the transmitted chirp signal to achieve a 0.2 mm standard deviation for one particular data point at a distance of 1 m. In all aforementioned methods the reference signal used for the cross-correlation operation has either been generated during transmission or was carefully calibrated using thousands of samples. This reference signal is heavily dependent on the piezoelectric transducer characteristics. In this work, our main focus is to investigate the possibility of processing the received signal of a piezoelectric transducer operating in bulk resonance mode with a $\Sigma\Delta$ modulator and to process its bitstream output signal with the cross-correlation technique while being independent of the transducer that is used.The results of this investigation are discussed in detail in this thesis.

In this chapter we will give a brief introduction to ultrasonic distance measurement, followed by the target specifications and the organization of the thesis.

1-1 Ultrasonic Distance Measurement

The general principle of ultrasonic distance measurement is the pulse-echo technique and is shown in Figure 1-1. A short sequence of pulses is applied to an ultrasonic transducer which

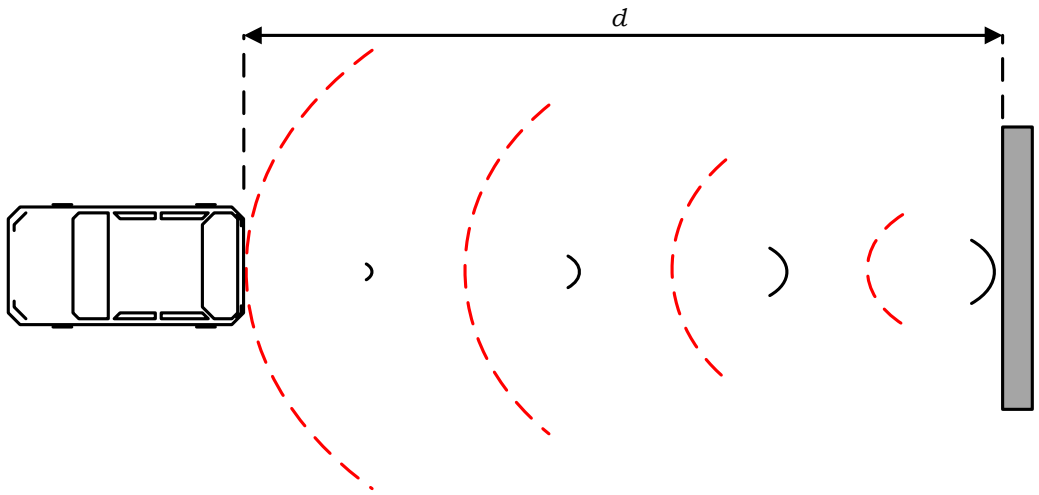

Figure 1-1: General principle of ultrasonic distance measurement: pulse-echo technique.

causes oscillation of the piezoelectric material resulting in a short sequence of ultrasonic waves in the air. These waves are reflected when they encounter a change in acoustic impedance, for instance an air-concrete interface. The reflected waves travel back in the direction of the transducer. The time passed since the transmission and reception of the ultrasonic waves is called the Time of Flight (ToF). The ToF can be used to calculate the distance between transducer and object, d:

$$d = \frac{c \cdot ToF}{2} \tag{1-1}$$

in which c represents the speed of sound in the medium and the factor 2 accounts for the distance that is traveled by the ultrasonic waves from transducer to object and back to the transducer.

The speed of sound c is given by [6]:

$$c = \sqrt{\gamma \cdot \frac{p}{\rho}} = \sqrt{\frac{\gamma RT}{M}} \tag{1-2}$$

in which p is the air pressure,ρ is the air density, R is the molar gas constant, M is the molar mass of the gas and T is the absolute temperature in Kelvin. It is independent of the frequency f and air pressure p in ideal gasses. This is because the ratio between air pressure p and the density ρ of ideal gasses is always the same. It is however dependent on the temperature and humidity. Higher humidity slightly increases the speed of sound in air because the molar mass M of H_2O (water) is lower than that of N_2 (nitrogen). Furthermore, since air is not an ideal gas, there is a slight dependence on the ratio between air pressure and density. γ is the adiabatic index that is approximately 1.4 for air at room temperature and varies less than 1% from 0 °C - 100 °C. At room temperature (20 °C) and a humidity of 20% the speed of sound in air is 343 m/s.

From Equation 1-1 and Equation 1-2 we can conclude that the speed of sound in air depends on several factors that can influence the correctness of the readout. Therefore, before doing distance measurements, the temperature and humidity have to be measured and the heat adiabatic index has to be determined.

1-2 Target Specifications

As mentioned before, the main goal of this thesis is to investigate the possibility of processing a bitstream output with the cross-correlation technique while being independent of the piezoelectric transducer that is used. A simplified system representation is shown in Figure 1-2.

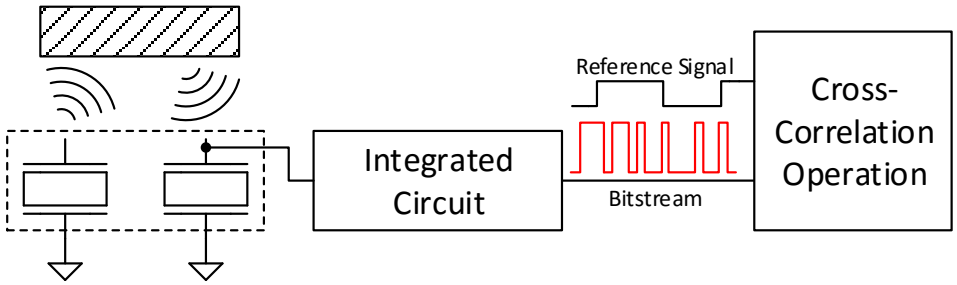

Figure 1-2: Simplified System Architecture.

The piezoelectric transducer can act as both transmitter and receiver. The integrated circuit will transform the received signal into a bitstream which will then be cross-correlated against a reference signal, the result of which should contain the ToF information.

The prototype circuit will be designed in a Texas Instruments 0.5 μm CMOS process with 3 metal layers and 5.0 V devices. Table 1-1 lists the target specifications of the system.

Target Parameter	Objective
Range	10 cm - 10 m
Resolution [1]	$<$ 1 mm
Supply Voltage	5.0 V
Transducer Frequency	40 kHz
Transmission Medium	Air

Table 1-1: System target specifications.

The readout circuit should be designed in such a way that it is insensitive to piezoelectric transducer variations, given that the frequency remains unchanged. Power consumption is not of primary concern but should be kept as low as possible without negatively affecting the performance.

[1] For a single pulse-echo measurement.

1-3 Organization of the Thesis

Chapter 2 gives background information on piezoelectric transducer characteristics. The piezoelectric effect is discussed, followed by the Butterworth van Dyke Model for piezoelectric transducers. The noise characteristics and input-output characteristics are derived. The chapters concludes by discussing the effects of non-ideal quality factors and the effect of varying beam angles.

Chapter 3 discusses different ultrasonic distance measurement techniques, such as threshold detection, phase detection, self-interference and cross-correlation. The chapter also gives a comparison between the aforementioned techniques.

Chapter 4 deals with the system level architecture and analysis of the system. The chapter starts by discussing the global system architecture. Several system level considerations are discussed, followed by an analysis of the system.

Chapter 5 deals with the circuit implementation. The design of various blocks is discussed in detail. Furthermore, the chapter discusses the circuit level simulation results. Finally, the implementation and analysis of the filter are discussed.

Chapter 6 deals with the measurement results. First the measurement approach will be presented followed by a discussion on the obtained results.

Chapter 7 will summarize the contributions of this thesis.

Chapter 2

Piezoelectric Transducer Characteristics

In this chapter some background information is given on the piezoelectric transducer. First the model that is used will be discussed. Next, the noise characteristics will be derived. Then the input-output characteristics will be derived. Finally, the effects of both the quality factor and beam profile will be treated.

2-1 Butterworth-van Dyke Model

Ultrasonic transducers are used both as transmitter and as receiver. When used as a transmitter, the transducer converts an applied electrical signal into ultrasonic waves. When used as a receiver, the transducer converts an incoming ultrasonic wave into an electrical signal. A typical ultrasonic transducer is composed of several layers that are shielded from external signals by a case. The structure of a typical transducer is shown in figure 2.

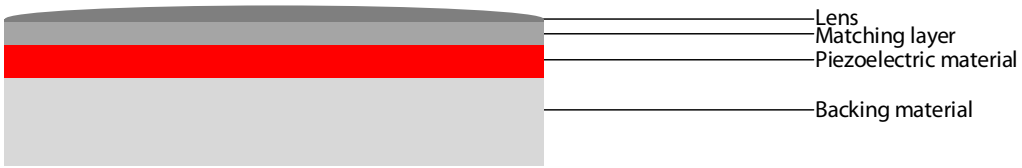

Figure 2-1: Simplified layer composition of an ultrasound transducer[1].

The function of the backing material is to prevent excessive vibrations, hence increasing the spatial resolution of the transducer. It also reduces the quality factor of the transducer resulting in an increased bandwidth, which is not always wanted. When a voltage is applied to the piezoelectric material, it experiences the piezoelectric effect and causes the transducer to

expand or compress based on the polarity of the applied voltage. The resulting wavelength of the emitted ultrasonic signal is dependent on F_{drive}. When driven at the resonance frequency the resulting wavelength is roughly equal to twice the thickness of the piezoelectric layer. In order for the ultrasonic waves to propagate from the piezoelectric material to air (and vice versa) without being significantly attenuated, a matching layer is needed to match the impedance of the transducer to the acoustic impedance of air. The lens is used to focus the emitted waves to a transducer specific beam angle.

Piezoelectric transducers can be modelled by the Butterworth-van Dyke (BVD) lumped circuit model shown in Figure 2-2 [7].

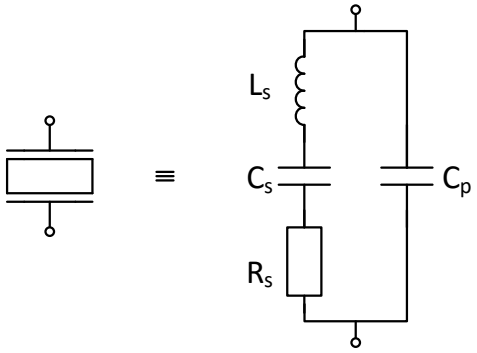

Figure 2-2: Butterworth-van Dyke equivalent circuit.

Table 2-1: Lumped parameter circuit values for a typical 40 kHz ultrasonic transducer.

Parameter	Typical Value
R_s	1 kΩ
L_s	159 mH
C_s	99 pF
C_p	3 nF

The BVD model consists of a series branch composed of a resistor R_s, capacitor C_s and inductor L_s, in parallel with a capacitor C_p. The capacitance C_p represents the dielectric capacitance of the piezoelectric material and the capacitance of any device that is connected to it. This capacitance represents the static state of the transducer, i.e. the part of the transducer that is not in motion. The series branch consisting of R_s, C_s and L_s represents the mechanical behaviour of the system. The resonance frequency is determined by the total impedance of the series branch, whereas the mechanical losses are characterized by R_s. The total impedance of the series branch is a sum of the reactance of the inductance and capacitance, X_{L_s} and X_{C_s} respectively, summed with the resistance of R_s:

$$Z_s = X_{L_s} - X_{C_s} + R_s = \omega L_s + \frac{1}{\omega C_s} + R_s \tag{2-1}$$

At 40 kHz, X_{L_s} and X_{C_s} are equal but opposite in sign cancelling each other out and reducing the total reactance in the branch to 0. The total impedance of the branch is then reduced to its minimum value of R_s. Typical values for ultrasonic transducers with a resonance frequency of 40 kHz are listed in Table 2-1.

The BVD model is used in this thesis because it gives straightforward insight into the behaviour of transducers. The BVD model is accurate for a narrow band of frequencies around the resonance frequency, but becomes less accurate for frequencies that are far away from the resonance frequency [7][8]. Since our main interest is the resonance frequency, this model is sufficient.

2-2 Noise Characteristics

It is useful to determine the total noise from the transducer to deduce the minimum requirements for the rest of the circuit. The thermal noise originating from the transducer is modeled by the thermal noise associated with resistor R_s. Using the lumped circuit parameters from Table 2-1, the noise spectral density can be calculated:

$$\sqrt{S_{n,R_s}} = \sqrt{4kTR_s} = \sqrt{1.65 \cdot 10^{-20} \cdot 1 \cdot 10^3} = 4 \text{ nV}/\sqrt{\text{Hz}} \tag{2-2}$$

The total noise voltage depends on the equivalent noise bandwidth of the transducer which is determined by the transducer's Q.

2-3 Input-Output Characteristics

So far a lumped circuit model of a piezoelectric transducer in the form of the BVD model was introduced. Furthermore, we have given some background information on the piezoelectric effect and have derived the noise characteristics of the transducer. Let us talk about the output of the transducer now.

The transfer function of the BVD model when the transducer is used in voltage mode is given by:

$$H(s) = \frac{V_o}{V_i} = \frac{C_s}{s^2 L_s C_s C_p + s R_s C_s C_p + (C_s + C_p)} \tag{2-3}$$

Where V_i is located in series with resistor R_s and V_o is the voltage on the top node in Figure 2-2.

At low frequencies the transfer of the system is dominated by the capacitance of both branches and can be simplified to:

$$H(s) = \frac{1}{1 + C_p/C_s} \tag{2-4}$$

Since $C_s \ll C_p$, the transducer has almost no response when driven by low frequency signals. At the oscillation frequency, the total reactance in the series branch becomes 0 and the transfer function can be simplified to:

$$H(s) = \frac{1}{1 + sR_sC_p} \tag{2-5}$$

At frequencies greater than the oscillation frequency the system's transfer function is:

$$H(s) = \frac{1}{1 + s^2 L_s C_p} \tag{2-6}$$

producing a 40 dB/dec drop in magnitude. A bode plot that consists of all aforementioned frequency ranges is displayed in Figure 2-3.

2-4 Effect of Quality Factor Q

The quality factor Q of a transducer modeled by the BVD lumped circuit model is given by Equation 2-7 and gives a dimensionless measure for the dissipation.

$$Q = \frac{\sqrt{L_s/C_s}}{R_s} \tag{2-7}$$

In the frequency domain a high-Q transducer is recognized by a narrow peak in the transfer function around the resonance frequency, whereas a low-Q transducer has a lower peak and a wider -3 dB bandwidth. The normalized transfer function has been plotted in Figure 2-3 for transducers with different Q's to allow straightforward comparison. Because Equation 2-4 reveals that different transducer characteristics result in different gains, the normalized results are displayed to allow straightforward comparison.

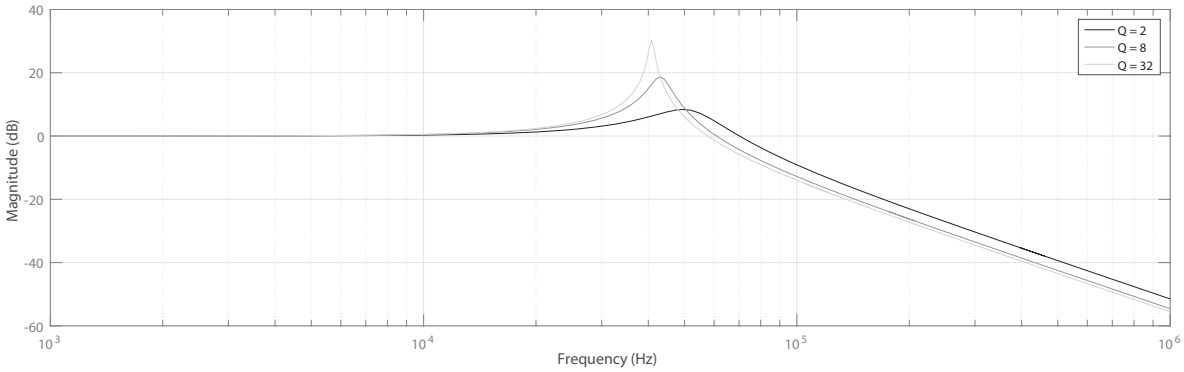

Figure 2-3: Normalized transfer function of the piezoelectric transducer when modelled by the BVD lumped circuit model.

The time-domain response of transducers with different Q's is displayed in Figure 2-4 when driven by a sine-wave at the resonance frequency. (The shift in resonance peak has been compensated for by adjusting the frequency of the input signal.)

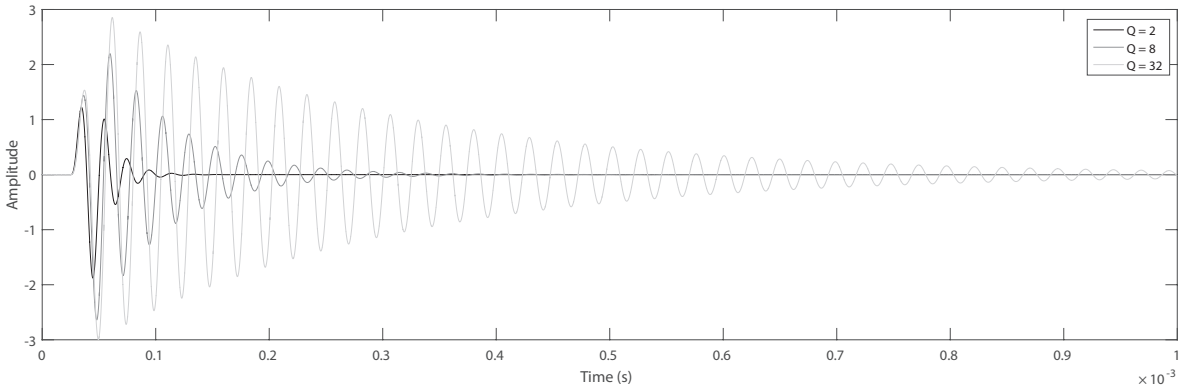

Figure 2-4: Normalized transfer function of the piezoelectric transducer when modelled by the BVD lumped circuit model.

The time-domain results give important insight in the behaviour of transducers. When a low-Q transducer is used, the ringing of the piezoelectric transducer is minimized, but the emitted ultrasonic wave will have a lower amplitude and be more susceptible to noise. A high-Q transducer will have a larger amplitude when driven by the same signal, increasing its SNR and also increasing its maximum range. A high-Q transducer will however experience more ringing, resulting in a reduced spatial resolution. Furthermore, aging effects will reduce the Q of transducers over time.

Even though the response of transducers is heavily dependent on their Q, it is the goal of this thesis to design a circuit that will be independent of the transducer used, which has proven to be quite a challenge.

2-5 Propagation of Ultrasonic Waves in Air

It is important to study the propagation of ultrasonic waves in air to understand the dynamic range of the arriving signals. We know from section 2-1 that waves emitted by an ultrasonic transducer originate from the whole surface of that transducer. It is difficult to model the behaviour of such a system at short distances however. At far distances the transducer can be modeled by a point source. The limit between near-field and far-field is givem by the fraunhofer distance [9] and is expressed as:

$$d_f = \frac{2D^2}{\lambda} \tag{2-8}$$

In which D represents the diameter of the piezoelectric transducer and λ represents the wavelength of the transmitted wave. The transducers we use have a diameter of 16 mm and a wavelength of 8.57 mm, leading to a d_f of 4.4 cm. Since we are interested in ranges above 10 cm, we can use far-field approximations and model the piezoelectric transducer by a point source. The sound intensity of the transmitted signal decreases quadratically as the distance increases because the available signal power is spread over the area of a sphere:

$$A_{sphere} = 4\pi r^2 \tag{2-9}$$

From section 2-1 we also know that ultrasonic transducers often have a certain beam angle θ that depends on the curvature of the lens used. In that case the available signal power is not spread of the entire sphere, but is limited to the area of the spherical cap:

$$A_{cap} = 2\pi r h \qquad ; h = r(1 - \cos\theta) \tag{2-10}$$

where r is the distance from the source and h is displayed in Figure 2-5. Equation 2-10 can be re-written to:

$$A_{cap} = 2\pi\alpha r^2 \qquad ; \alpha = 1 - \cos\theta \tag{2-11}$$

From Equation 2-11 it is obvious that the behaviour of a transducer with a limited beam angle is the same as an omnidirectional transducer. The only difference between the two lies

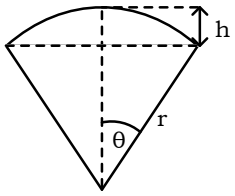

Figure 2-5: Spherical cap.

in the factor α which leads to an increase in range for narrower beam angles. The factor of range increase that can be achieved by a reduced beam angle θ is given by

$$f = \frac{A_{sphere}}{A_{cap}} = \frac{2}{\alpha} = \frac{2}{1 - \cos\theta} \tag{2-12}$$

Qualitatively this can be explained as follows: For a transducer with a smaller beam-angle, the area of the spherical cap increases with the same $1/r^2$ fashion as the area of a full sphere does as the distance increases. The difference lies in the available power per area. Where for an omnidirectional transducer the power is spread across the whole surface-area of the sphere, the same power is available in a smaller part of the sphere for a beam-angled transducer. Along with the decrease in sound intensity, the atmospheric sound absorption is also taken into account. This absorption is however only in the order of 1.3 dB/m for 40 kHz ultrasonic waves at room temperature [10].

Chapter 3

Comparison of Ultrasonic Distance Measurement Techniques

In this chapter we will discuss various techniques that can be used for ultrasonic distance measurement. The chapter starts with a review on threshold detection, followed by a discussion on phase detection. Then self-interference is discussed. Subsequently cross-correlation for ultrasonic distance measurement is discussed. The chapter ends with a comparison of the discussed methods.

3-1 Threshold Detection

The threshold detection method uses the signal's amplitude information to determine the distance. The ToF is defined as the point in time where the echo passes a certain predefined threshold level. This point in time is defined in Figure 3-1 as the level cross time. The threshold may be crossed at a later time than the true ToF, this can however be systematically compensated.

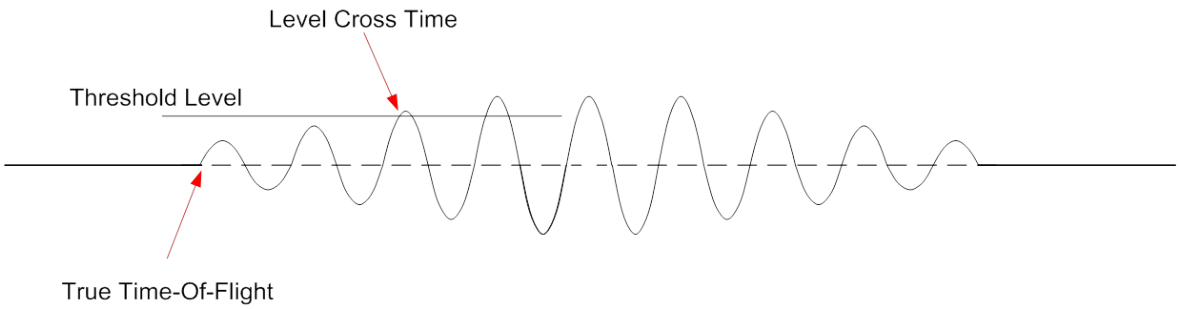

Figure 3-1: Threshold detect.

For low-SNR signals, this method is not very suitable because (slight) amplitude variations can cause a pulse to be skipped . Figure 3-2 shows the original signal in black and the noise

affected signal in red. (Note that in this figure the noise causes the signal to decrease at all points in time, just for simplicity reasons.) In the figure one pulse has been skipped because noise has caused the signal amplitude at the sampling moment of the initial level cross time to decrease to a value lower than the threshold. Pulse skipping will cause an error of one or more wavelengths, ultimately limiting the resolution of such systems to 1 wavelength [11]. The wavelength λ is given by:

$$\lambda = \frac{c}{f_{carrier}} = \frac{343 \text{ m/s}}{40 \text{ kHz}} = 8.57 \text{ mm} \tag{3-1}$$

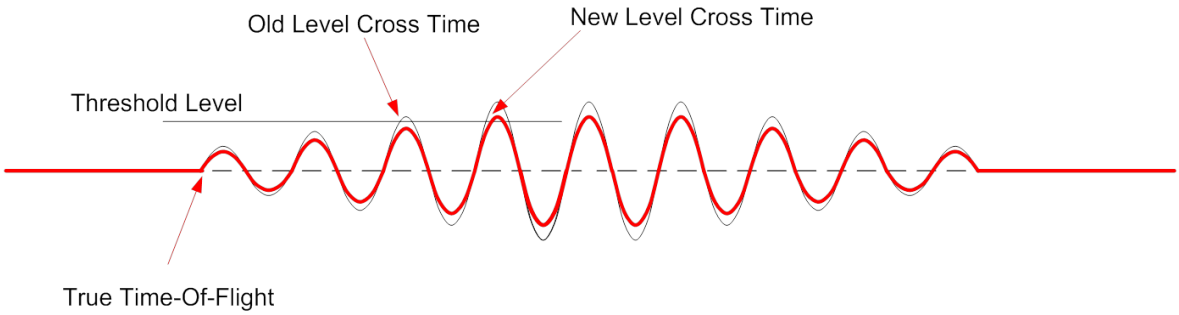

Figure 3-2: Pulse skipping.

Furthermore, the threshold detection method requires a time-dependent gain or threshold as depicted in Figure 3-3. As the signals continue to grow weaker for longer distances, the threshold needs to continually be adjusted as well. By doing so, the input range of the comparator is adjusted to the echo amplitude. Additionally, if no variable gain or threshold is applied, the comparator would produce more pulses for short distance echos than for long distance echos.

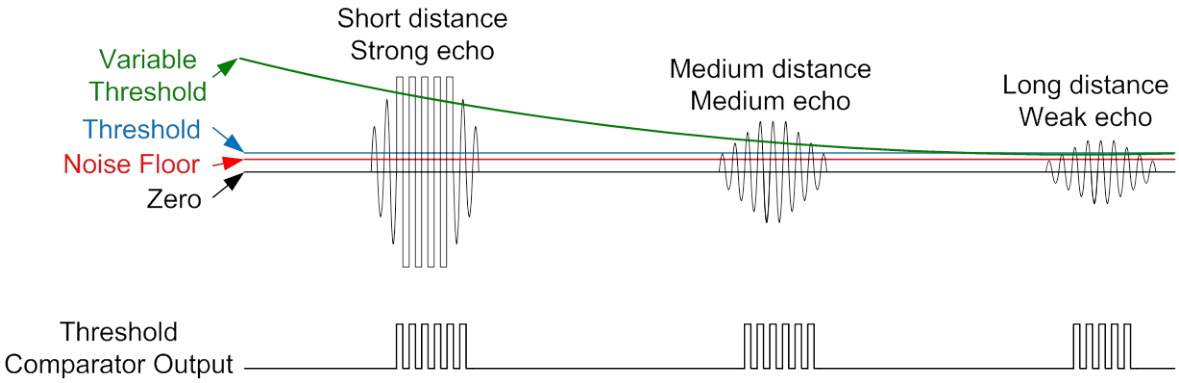

Figure 3-3: Variable threshold and the corresponding comparator output for short, medium and long distance echos.

3-2 Phase Detection

By sending a continuous time signal and measuring the phase difference between the sent and received signals it is possible to accurately measure the distance. The problem however, is that the distance can only be determined unambiguously within one signal wavelength λ [12]. Therefore, if the distance is greater than λ (8.57 mm) the system will be unable to determine the distance unambiguously, effectively reducing the measurable range to one wavelength.

However, phase detection is often paired with other techniques to increase the measurement range of such ultrasonic ranging systems. In [12] rough Time-of-Flight (RTOF) measurements using the envelope of the received signals are combined with phase detection measurements to determine the distance with an accuracy of 0.295 mm for a range of 0.5842 m (0.05%). In [13] phase detection is paired with cross-correlation for typically better than 1 mm resolution for distances up to 1 m.

3-3 Self-Interference

Self-interference was first proposed in [14] in 1993. The principle of operation is described here. Two pulse trains with a finite number of pulses are transmitted by a transducer. The phase delay between the pulse trains is set to π. Due to interference, phase inversion will occur, causing the signal envelope to become zero. This is illustrated in Figure 3-4.

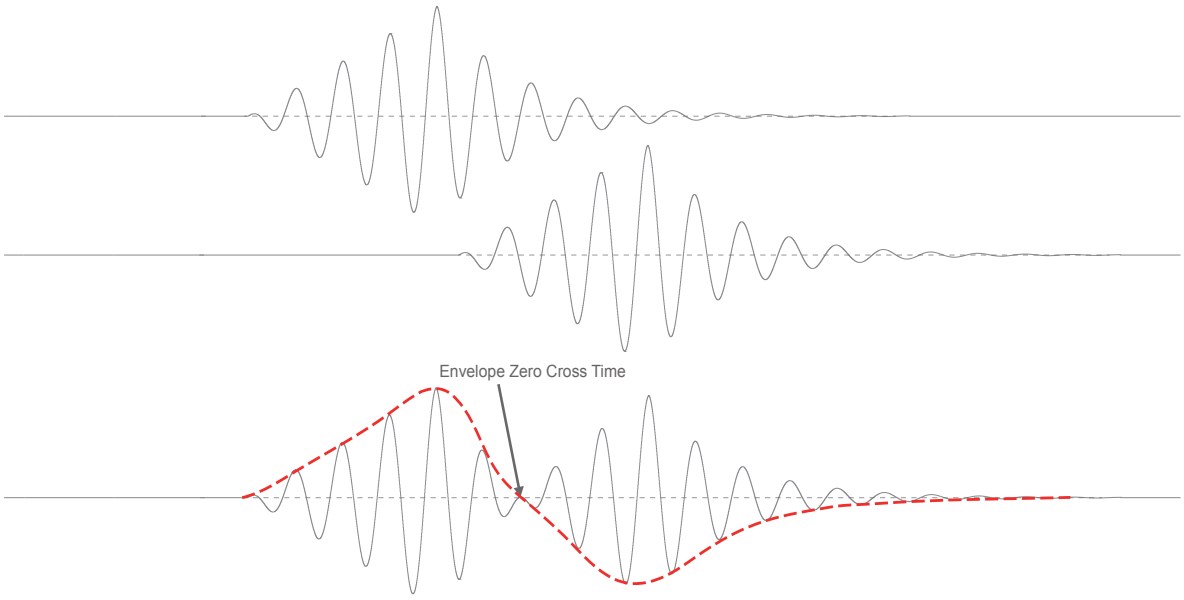

Figure 3-4: Self-interference.

The point at which phase inversion occurs is the point at which the power contained in a pulse of the second train equals the power contained in a pulse of the first pulse train. At this point, the sign of the signal will remain the same. To accurately determine the ToF this point should be well known.

An advantage of this method is its reduced noise sensitivity because zero crossings are detected instead of amplitudes.

The problem with this method however, is also amplitude related. In low-SNR situations it is hard to distinguish the signal from the noise. While interference will still introduce a phase inversion, the exact moment this happens on will be unknown. Multiple zero crossings can exist between the consecutive pulse trains due to low SNR. Furthermore, a change in Q will affect the location of the zero crossing. A lower Q will result in slower rise times and reduced ringing, ultimately causing the zero crossing to be shifted forward.

In [15] two algorithms are proposed that can calculate the ToF in high and low SNR with an average relative error of 0.34% in a range from 40mm - 180mm. However, the complexity of the proposed algorithms requires increased hardware complexity. In [16] self interference is combined with sub-sampling and interpolation to reduce hardware complexity and achieves around 1/10 λ accuracy for 2m range. In both [15] and [16] the system will have to be calibrated for different transducers.

3-4 Cross-Correlation

The cross-correlation operation is a measure of similarity between two waveforms [17]. For discrete-time waveforms the cross-correlation operation is given by:

$$Corr(x[n], h[n]) = \sum_{k=0}^{\infty} (h[k]x[n+k]) \tag{3-2}$$

Similar to convolution, the cross-correlation operation performs a summation on the multiplication of received samples with stored samples. For closely matching signals, the cross-correlation operation will output higher values than for poorly matching signals. The difference between convolution and cross-correlation lies in the time-reversal of one of the signals in the convolution operation. By letting the received signal pass through the stored samples, from back to front, we are comparing the two signals in a causal order.

Unlike threshold detection where a single point in time is compared to some predefined threshold level, cross-correlation takes into account all available information in the two waveforms. It acts as a matched filter, making the cross-correlation less sensitive to noise. The more samples are compared, the less sensitive the system becomes to noise, but the more hardware complexity is added. In general, the outstanding capability of cross-correlation to recover signals from noise, allows correlation based systems to do longer range measurements. Moreover, instead of longer ranges, higher frequencies can be employed as to increase the resolution.

Cross-correlation can be used in many different configurations regarding ultrasonic distance estimation. In [2] a $1/10\lambda$ accuracy is achieved by cross-correlating the transmitted signal with a reference signal that is acquired by averaging over 10^3 scans. In [3] a comparable accuracy is achieved by correlating against a well-known signal. The reference signal is obtained by measuring the output of the transducer at a well-known distance.

In [4] cross-correlation is combined with sine-fitting to achieve $1/100\lambda$ resolution for distance measurement. The cross-correlation operation is used to detect the integer number of wavelengths, whereas the sine-fitting technique is used to accurately estimate the phase shift.

In [5] cross-correlation is employed on a bitstream output, but the reference signal is acquired much as in [3] by digitizing the transmitted pulse. The signal used was a 200-period chirp

signal resulting in a 10^4 point reference signal. A recursive cross-correlation method was employed to reduce the length of the reference signal to approximately 2x the number of periods. The very long length of the pulse signal increases the minimum range of the sensor to around 1.7m between two transducers or 0.85 meters for a transducer and a reflective surface.

3-5 Comparison Results

In this work, our main focus is to investigate the possibility of processing a bitstream output signal with the cross-correlation technique while being independent of the transducer that is used. Table 3-1 summarizes the discussion on the compared ultrasonic distance measurement techniques.

Table 3-1: Comparison of ultrasonic distance measurement techniques.

Measurement Technique	Pulse Skipping	Amplitude Sensitivity	Ranging Accuracy	Hardware Complexity
Threshold Detection	-	- -	- -	+ +
Phase Detection	- -	+	+ -	+
Self-Interference	+	-	-	-
Cross-correlation	+	+	+ +	-

The main goal of this thesis is to propose an ultrasonic distance measurement system that achieves sub wavelength resolution for distances between 10 cm and 10 m while being transducer independent.

Threshold detection is not suitable for systems that require sub-wavelength resolution because of poor performance in low SNR situations. Because the ToF is retrieved from the amplitude information in the noise corrupted signal, pulse-skipping can occur. Due to pulse-skipping the resolution of such systems is limited to a single wavelength.

Phase detection provides good accuracy for the fractional part of a single wavelength. However, it cannot accurately determine the integer number of wavelengths unless it is paired with other techniques such as cross-correlation or threshold detection.

Self-interference and cross-correlation do not suffer from the pulse skipping problem and are both suitable for long and short range measurements. Self-interference is however very dependent on the quality factor of the transducer in order to determine the zero-cross time of its envelope signal. A transducer with a lower Q will ring less than one with a higher Q causing the zero-cross time of the envelope to shift forward. In order for ranging to be accurate, the system would need to be calibrated for every transducer. The effect of transducer Q on cross-correlation is less severe than for self-interference because only one signal is sent and received. Furthermore, the hardware complexity of a cross-correlation system can be reduced by applying a recursive filtering method [5].

Because of the aforementioned reasons, cross-correlation was chosen as the ultrasonic distance measurement technique for this thesis. More elaborate design choices regarding cross-correlation will be treated in chapter 4.

Chapter 4

System-Level Architecture and Simulation

In this chapter we will discuss various components of the system-level architecture and the corresponding simulation results. This chapter starts by introducing the chosen system architecture. This is followed by discussing the system level considerations, including the readout of the transducer, ADC choice, DAC-type and filter architectures. Then the designed MATLAB GUI and its underlying models are introduced. Finally, the performed MATLAB simulations are treated, the main focus of which is on the cross-correlation method.

4-1 System Architecture

A simplified circuit diagram and the corresponding timing diagram are shown in Figure 4-1a and Figure 4-1b respectively.

On the left side a 20 V driver is connected to a 40 kHz piezoelectric transducer. The transducer is connected to the high-voltage driver and disconnected from the rest of the circuit in phase ϕ_1. In phase ϕ_1 the transducer is excited with an electrical signal. In order to protect the readout circuit from the high voltage, high voltage switches are implemented. A non-overlap phase exists between phase ϕ_1 and ϕ_2 to ensure that no high-voltage signal reaches the readout circuit. In phase ϕ_2 the high-voltage driver is disconnected from the transducer and the readout circuit is connected in order to capture the echoes of the transmitted pulse. The readout circuit is a first order continuous-time $\Sigma\Delta$-modulator. The Feedback DAC is implemented using a voltage DAC in combination with resistors, instead a current steering DAC in order reduce power consumption.

The system is arranged in a voltage readout configuration where current feedback is applied directly to the transducer. A continuous-time $\Sigma\Delta$-ADC with an over-sampling ratio (OSR) of 200 is implemented to provide amplitude information in a 1-bit format. The OSR is derived from the system simulations in section 4-3. The transducer is also used for the integration

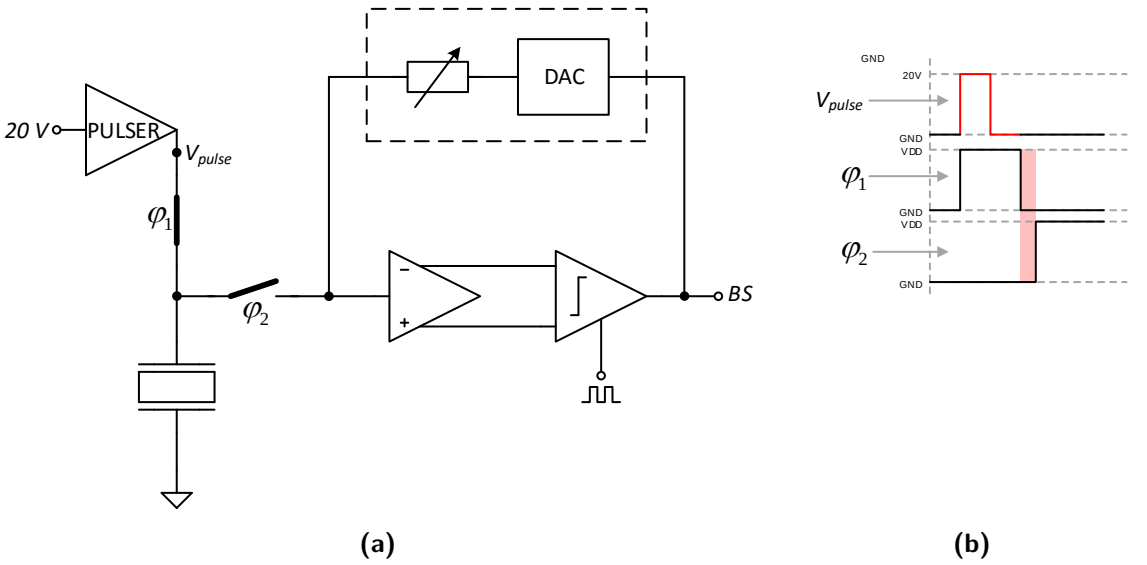

Figure 4-1: (a) Simplified circuit diagram and (b) the corresponding timing diagram.

operation of the ΣΔ-modulator, eliminating the need for an on-chip integration capacitor. The cross-correlation operation is implemented off-chip, allowing for flexibility in the filter architecture used, the best of which can be implemented into hardware in future designs.

4-2 System-Level Considerations

This section starts by discussing the readout of the transducer, followed by a comparison of suitable ADC architectures. Next is a discussion on the feedback DAC type. Finally, several filter options are introduced.

4-2-1 Readout of the Transducer

The most commonly used low-noise amplifiers (LNA) for piezoelectric transducer readout are the inverting voltage amplifier with capacitive feedback (IVACF) to read out voltage and the charge amplifier (CA) to read out current/charge displayed in Figure 4-2. The IVACF senses currents, whereas the CA senses charge.

For the chosen architecture of Figure 4-1a it is very difficult to perform current readout. ΣΔ modulators often use high over-sampling ratios (OSR) to increase the system resolution. These high frequencies however, also reduce the impedance of the parasitic capacitance of the transducer (Z_p). Normally, when the transducer is not used for the integration operation, efficient current readout is only possible when the input impedance of the readout amplifier is much lower than the parasitic impedance of the transducer.

At an OSR of 200, the sampling frequency becomes:

$$F_s = F_{BW} \cdot 2 \cdot OSR = 40 \cdot 10^3 \cdot 2 \cdot 200 = 16\text{MHz} \tag{4-1}$$

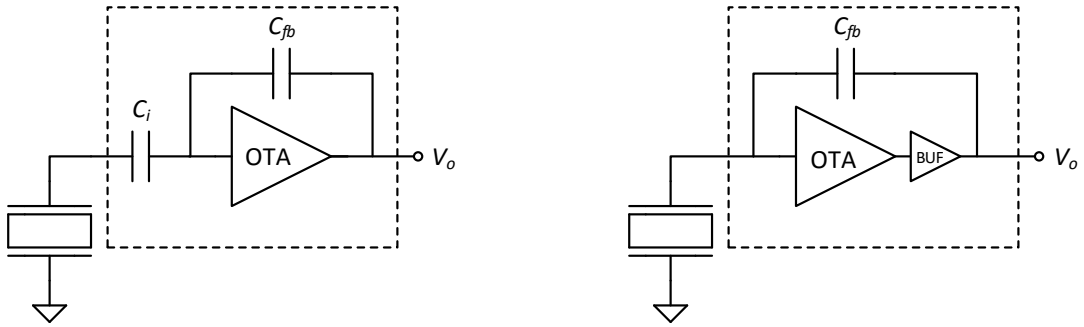

Figure 4-2: Inverting voltage amplifier with capacitive feedback (left) and a charge amplifier (right).

At this sampling frequency the parasitic impedance of C_P becomes:

$$Z_{C_P} = \frac{1}{\omega C_P} = \frac{1}{2\pi \cdot 16 \cdot 10^6 * 3 \cdot 10^{-9}} \approx 3.3\ \Omega \tag{4-2}$$

Because of this low impedance, it would be very power demanding to perform active integration for the chosen architecture of Figure 4-1a.

For our chosen system architecture however, voltage readout proves to be very desirable since both the input current and feedback-DAC current are integrated into a voltage across the capacitor. At the sampling frequency of interest, the transducer behaves as a voltage source that stays relatively constant around the input bias voltage of the pre-amplifier due to the negative feedback provided by the DAC.

Additional information can be found in [18] in which the noise-efficiency of both current and voltage readout of ultrasonic transducers is discussed.

If an active integrator is implemented by inserting a feedback capacitor C_{FB} between the input of the pre-amplifier and the input of the quantizer, the majority of the current provided by the feedback DAC will still go through C_P at the frequency of interest because of its low impedance. This situation can be modeled as shown in Figure 4-3

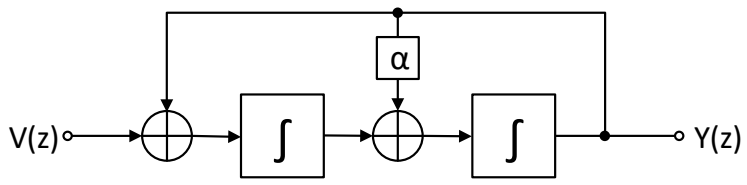

Figure 4-3: Model of active integration.

The first integrator represents the passive integration over C_P, the second integrator represents active integration over C_{FB} and α represents a leaky path from the feedback circuit directly to the second integrator as a result of voltage division between the resistance of the DAC,the switch on-resistance and Z_{C_P}. If the leaky path is ignored, the operation is as follows: The DAC provides a feedback current that is passively integrated across the transducer.

A current then results from the transducer which is actively integrated across C_{fb}. This means that a 180 degree phase shift is introduced in the modulator at a certain frequency, resulting in positive feedback.

Our choice is therefore to combine passive integration across the transducer with voltage readout. Using this combination results in a compact form, because the transducer can be used as the integration capacitor. Furthermore, the current can be effectively integrated across the transducer because of it's low impedance at high OSRs. The downside of passive integration is that the integrator can become leaky if small resistances exist in the feedback path, which gives rise to dead zones in the ΣΔ output [19]. This can actually be solved if an active integrator is implemented as depicted in Figure 4-3. If the loop can be stabilized, the additional the gain will result in a reduced signal swing at the transducer, resulting in reduced leakage [20].

4-2-2 Continuous-Time ΣΔ Modulator or Discrete-Time ΣΔ Modulator

ΣΔ-ADCs are oversampling ADCs that have sampling frequencies much higher than the Nyquist frequency. The ratio between the sampling frequency and oversampling frequency is called the oversampling ratio (OSR). A ΣΔ-ADC generally consists of a ΣΔ-modulator followed by a decimation filter as shown in Figure 4-4.

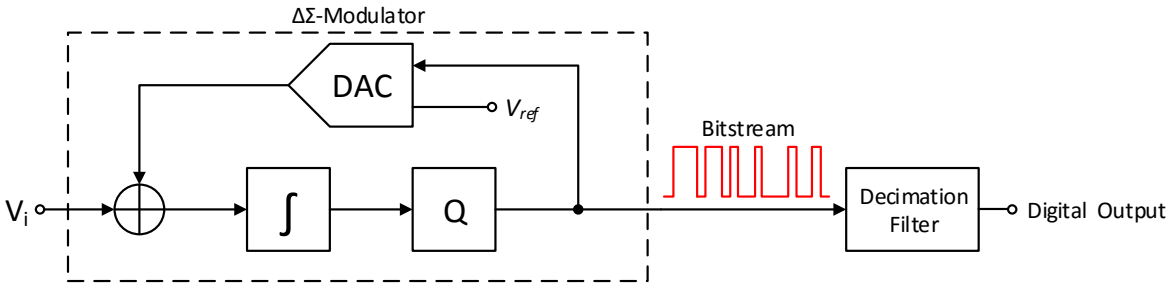

Figure 4-4: A typical ΣΔ architecture.

The ΣΔ-modulator consists of a loop filter $\int$, followed by a quantizer Q. The quantizer can either be single-bit or multi-bit. The function of the loop filter is to shape the quantization noise to higher frequencies while acting as a low-pass filter (LPF) for the in-band signals. The loop filter and feedback DAC can either be implemented in discrete-time or continuous-time depending on the chosen modulator architecture.

The ΣΔ-modulator can be modeled by its linear z-domain model in Figure 4-5.

Where X(z) is the input signal, Q(z) is the quantization noise that is injected at the quantizer and Y(z) is the bitstream output signal. The corresponding Signal-Transfer-Function (STF)

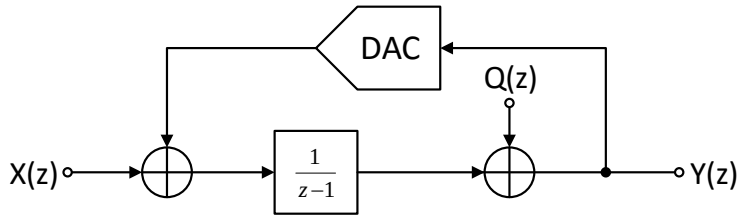

Figure 4-5: Z-domain model of a ΣΔ modulator.

and Noise-Transfer-Function (NTF) are given by [21]:

$$STF = \frac{Y(z)}{X(z)} = z^{-1} \tag{4-3a}$$

$$NTF = \frac{Y(z)}{Q(z)} = 1 - z^{-1} \tag{4-3b}$$

From equations 4-3a and 4-3b it is clear that the loop filter only adds a delay to the input signal whereas the quantization noise is differentiated. Equation 4-3b also reveals that the ΣΔ-modulator takes into account past values, eliminating the one-to-one relation with the input signal. This should not be seen as a disadvantage because it allows the system to store amplitude information in a 1-bit signal which is the main reason we prefer them over Nyquist-Rate ADCs. Assuming a 1-bit quantizer, the modulator will primarily output ones for high signal levels, whereas it will primarily output zeros for low-level input signals. Amplitude information in a 1-bit format can lead to decreased filter complexity.

Once the signal is in a 1-bit format, a digital low-pass filter is applied to attenuate the high frequency quantization noise and to reconstruct the original signal from the bitstream. A decimation filter is then applied to the output of the digital filter to reduce the sampling rate at the output of the ΣΔ-ADC to the Nyquist rate.

The SNR can be improved by increasing the OSR, increasing the order of the ΣΔ-modulator or by increasing the number of quantization bits.

Discrete-time ΣΔ-Modulator

In discrete-time ΣΔ-modulators the input signal is sampled at the input of the loop filter. The loop filter is implemented with switched capacitor circuits and operates in discrete-time. The feedback DAC is implemented using switched-capacitor techniques as well.

Continuous-time ΣΔ-Modulator

Continuous-time ΣΔ-modulators operate in much the same way as discrete-time ΣΔ-modulators. The main difference lies in the sampling location. For continuous-time ΣΔ-modulators the sampling takes place at the quantizer input. The loop filter can therefore be (more easily)

implemented using continuous-time circuits such as Gm/C or active RC integrators. Typically, continuous-time ΣΔ-modulators also contain a continuous-time feedback DAC. It is however possible to implement a ΣΔ-modulator, where the loop-filter is implemented using continuous-time circuits, while the feedback is implemented in discrete-time using switched capacitor circuits.

Comparison

For continuous-time ΣΔ-modulators the input signal is low-pass filtered by the loop filter before sampling, reducing the need for a dedicated Anti-Aliasing Filter (AAF) at the input of the modulator. Furthermore, the aliased signal have the same NTF as the quantization noise (Equation 4-3b), because they are injected at the quantizer input [22]. This is not the case for for discrete-time ΣΔ-modulators, because the sampling is done prior to the loop filter, requiring an anti-aliasing filter at the input of the modulator.

The continuous-time feedback DAC is more sensitive to clock jitter than its discrete-time counterpart. The difference lies in the way the charge is transferred during feedback. In a discrete time DAC, most of the charge is transferred at the start of the integration phase, whereas only a small amount of charge might be transferred at the end of the integration phase depending on how well the signal has settled.

Figure 4-6: Effect of jitter on SC DACs (left) and CT-DACs (right).

For continuous-time DACs the amount of transferred charge is directly proportional to the time of the integration phase. The effect of jitter is therefore more severe in continuous-time DACs as is displayed in Figure 4-6. On the other hand, due to settling requirements, discrete-time DACs have higher bandwidth and power requirements.

In [23] the jitter contributed charge is expressed as:

$$Q_j = \sigma_j I_{DAC} \tag{4-4}$$

where jitter is considered to be white noise, with standard deviation σ_j. The jitter current can then be expressed as:

$$I_j = \frac{\sigma_j}{T_s} \cdot I_{DAC} = \sigma_j F_s \cdot I_{DAC} \tag{4-5}$$

The jitter noise power in the signal band of interest is given by:

$$N_j^2 = \int_{-F_{BW}}^{+F_{BW}} \frac{I_j^2}{F_s} df = I_j^2 \cdot \frac{2F_{BW}}{F_s} \tag{4-6}$$

The total current noise due to jitter can then be expressed as:

$$i_{n,j} = \sqrt{I_j^2 \frac{2F_{BW}}{F_s}} \tag{4-7}$$

The total jitter noise current should be lower than the noise current generated by the transducer. The current noise generated by the transducer is derived from the voltage noise generated by the transducer, which can be calculated to be 1.8 μV for the transducer parameters listed in Table 2-1. By dividing by R_s, the transducers total current noise is obtained which has a value of 1.8 nA. From this the maximum $\sigma_{j,max}$ is derived for the strongest DAC feedback configuration that supplies ± 340 μA (derived later in sec:circdac):

$$\sigma_{j,max} = \frac{\sqrt{i_{n,R_s}^2 \cdot \frac{F_s}{2F_{BW}}}}{F_s \cdot I_{DAC}} = 2,1ps \tag{4-8}$$

In order to keep the system compact and low-power a continuous-time ΣΔ ADC has been chosen. To further reduce power consumption, a continuous-time feedback DAC is implemented because it has lower bandwidth and power requirements than a discrete-time feedback DAC. The jitter noise can be reduced by choosing a clock with a $\sigma_{j,max}$ that is lower than 2.1 ps.

4-2-3 Voltage- or Current-Domain DAC

The DAC is directly connected to the transducer. The function of the DAC is to bring the input-level of the system as close as possible to its bias point. This is done by either sourcing a current to, or sinking a current from the transducer. By doing so, charge packets are transferred to or from the transducer. Furthermore, the DAC will need to have a programmable feedback current in order to reduce quantization noise for longer range measurements where the received echoes are smaller. These currents can be generated directly through a current-mirror DAC or can be made indirectly with a voltage DAC in combination with a resistor.

Current-steering DACs are fast because their transistors are never turned off. However, their power consumption is high because unused current branches cannot be turned off. Turning unused current sources off would lead to the discharge of depletion layers and parasitic capacitance and remove the speed advantage.

Voltage DACs are generally made with resistors. Common architectures are the resistive- and R-2R ladder networks.

To reduce power consumption we have have chosen to implement a voltage DAC in combination with resistors. An R-2R-based scheme was proposed that can provide current in any desired ratio, however due to time-constraints we have chosen to implement a resistive ladder DAC.

4-2-4 Cross-Correlation Implementation

The cross-correlation operation can be implemented with a matched filter at 40 kHz. When the sampling rate is known, the length and coefficients of the filter can be chosen such that a

band-pass filter is formed around the frequency of interest. A FIR filter is used to implement the matched filter. FIR filters are digital filters that are specified with taps (coefficients). To achieve high accuracy, large arrays of coefficients are used. FIR filters can be linear phase and do therefore not cause phase distortion. Furthermore, because there is no feedback element in FIR filters, they are inherently stable. Unlike IIR filters, they also do not suffer from limit cycles.

We have chosen to only use two discrete values for our FIR filter: 1 and -1. This is done to keep the calculation and any (future) hardware implementations low. Suppose a sampling frequency of 16 MHz is used, that equals 400 discrete sampling points for one period of a 40 kHz signal. The first 200 taps of the FIR filter can be set to +1 and the last 200 taps can be set to -1. When a 40 kHz sine-wave goes through the filter (the cross-correlation operation) the filter will output a maximum. In effect, this configuration represents a bandpass (sinc) filter around 40 kHz.

To reduce the number of filter taps and the power consumption, more advanced structures can be used, two of which are regarded here.

Cascaded-Integrator-Comb Filters

Cascaded Integrator Comb (CIC) filters are a class of linear-phase low-pass FIR moving average filters. CIC filters can achieve sampling rate increase and decrease without the use of multipliers [24]. Since we are mainly interested in reducing the number of filter taps, we focus only on the decimation CIC filter type. The CIC decimating filter structure is shown below:

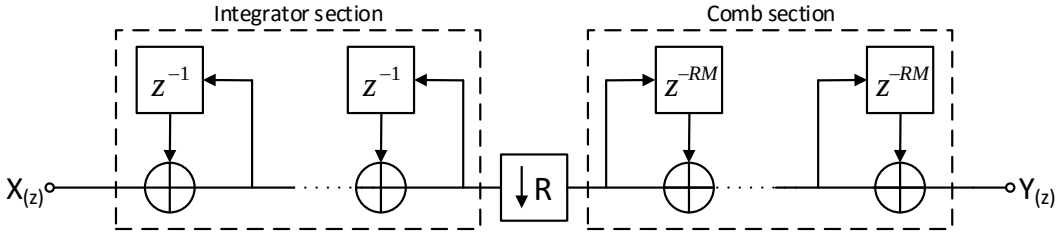

Figure 4-7: Decimating CIC filter architecture.

The decimating CIC filter structure is a cascade of single-pole integrator and comb stages, where the integrator stages are sampled at a sampling rate of F_s and the comb stages are sampled at a reduced sampling rate F_s/R. The $\downarrow$ R decimation operation discards all but the R^{th} sample, allowing for a reduced sampling rate in the comb section.By reducing the sampling rate, the memory and speed requirements for the comb section and any consecutive stages are relaxed. Both these effects result in a lower power consumption, while signal integrity remains unaffected.

Recursive difference of the cross-correlation

The cross-correlation operation of Equation 3-2 requires large amounts of multiplications and accumulations. In [5] the recursive cross-correlation operation is used to reduce the calculation cost. The calculation of the recursive cross-correlation is done by integrating the difference of the cross-correlation:

$$C[k] - C[k-1] = \frac{\pi}{4}\frac{1}{N} \cdot \sum_{k=0}^{N} \Big[x[k] \cdot \{h[N-k] - h[N-k-1]\} \Big] \tag{4-9}$$

Where $C[k-1]$ is just a delayed version of $C[k]$. The result of the recursive cross-correlation operation only changes value when a zero-crossing occurs in the received signal and is independent of sampling frequency. The number of taps can therefore be reduced to the number of zero crossings present in the transmitted signal. Consequently, the sampling frequency can be reduced as well.

In [5] the filter function was to perform a cross-correlation operation on a 200-period chirp signal. The filter length was reduced to approximately 400 taps by implemented the recursive cross-correlation operation and an accuracy of 0.2 mm was achieved. However, only one distance was measured and as analyzed in chapter 3 such a system requires two separate transducers to work for ranges lower than 1m.

For this project we have chosen to implement the FIR filter digitally in MATLAB so that we have the flexibility of trying out different filter types. Future implementations of the chip could incorporate the most suitable filter type into its hardware implementation.

4-3 System Simulation and Analysis

In this section a system level analysis is presented with a primary focus on the cross-correlation operation. The Graphical User Interface (GUI) that was designed to enhance the usability of the implemented system is introduced first. After that the underlying models of the GUI are discussed. Finally, that the simulation results are presented and the results are analysed.

4-3-1 Introduction to the GUI

The GUI that was designed in MATLAB is shown in Figure 4-8. The variables that can be adjusted in these simulation are listed in Table 4-1 with a short explanation.

Further information about the code is available in the attached MATLAB script in Appendix A. The part of the code that defines the GUI is left out since it is not of interest for the behavioural model of the system.

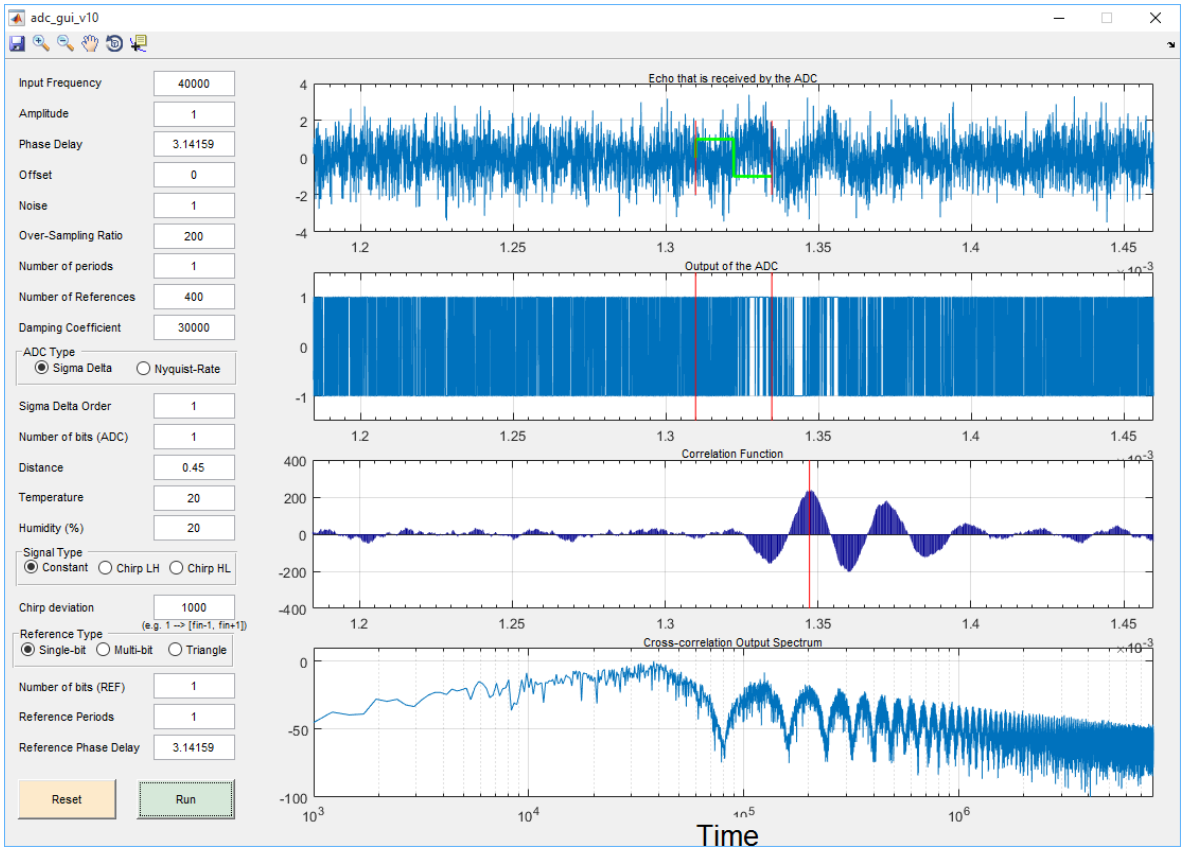

Figure 4-8: System Level Architecture.

Table 4-1: GUI parameter definitions.

Parameter	Definition
Input Frequency	The frequency of the input signal. The reference signal is also matched to the input frequency.
Amplitude	The amplitude of the transmitted signal at the end of excitation.
Phase Delay	the phase delay of the input signal.
Offset	The offset of the input signal.
Noise	Standard deviation for a normal distribution with a bandwidth equal to F_s.
Over-Sampling Ratio	$F_s = F_{in} \cdot 2 \cdot \text{OSR}$
Number of Periods (NoP)	The number of periods that drive the transducer.
Number of References (NoR)	NoR = NoP·2·OSR (Auto-generated)
Damping Coefficient	Related to the quality factor of the transducer.
ADC Type	Choose between a $\Sigma\Delta$ or Nyquist-Rate ADC
Sigma Delta Order	A 1^{st}- and 2^{nd}-order $\Sigma\Delta$ ADC have been implemented.
Number of Bits (ADC)	The resolution of the ADC.
Distance	The distance is used to calculate the signal's attenuation and resulting amplitude.
Temperature	Temperature in °C.
Humidity	Relative humidity in %.
Signal Type	Choose between a constant frequency or one that goes from low-to-high or from high-to-low.
Chirp Deviation	The deviation in frequency when Chirp LH or Chirp HL is chosen.
Reference Type	Single-bit, Multi-bit and Triangle references are implemented.
Number of Bits (REF)	The resolution of the reference signal.
Reference Periods	The number of periods in the reference signal.
Reference Phase delay	The phase delay of the reference signal w.r.t. the input signal.

4-3-2 Underlying Models

In order to understand the upcoming simulation results it is worthwhile to briefly review the most important underlying models. First the waveform model is introduced, followed by a brief introduction to the implemented cross-correlation operation and $\Sigma\Delta$ modulators.

Wave Model

The BVD model that was introduced in section 2-1 was implemented to model the transducer. After all the properties of the drive-signal and the transducer have been set in the GUI, the waveform is generated by passing the drive signal through the implemented BVD model. This is done twice to replicate the sending and receiving of the signal and is displayed below.

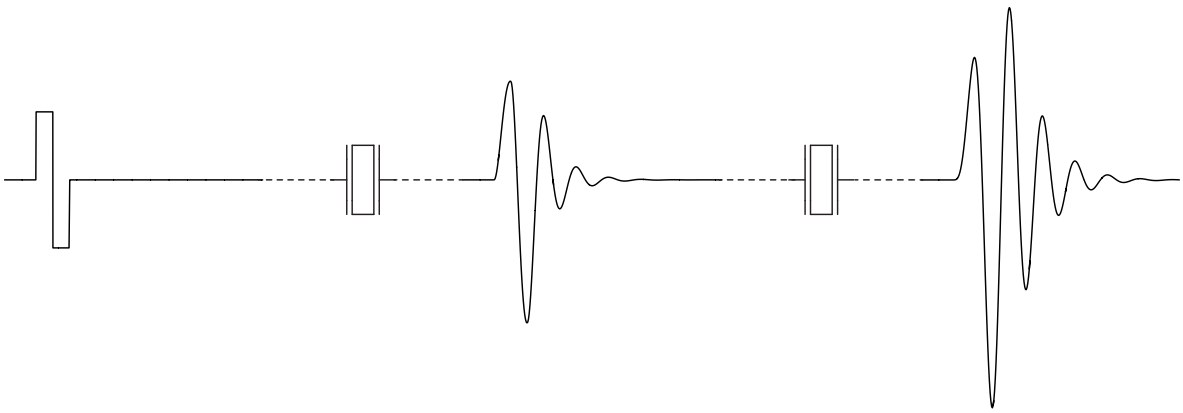

Figure 4-9: Signal transformation as it is passed through the transducer.

Depending on the chosen quality factor, the transducer amplitude will change. More importantly however, the amount of ringing is also heavily dependent on the Q of the transducer. The implemented MATLAB code can be found in Appendix A under the `get_echo` function.

Cross-Correlation Implementation

The cross-correlation operation is implemented by performing a convolution between the received echo and the time-reversed reference signal as presented in Equation 3-2 and repeated here for clarity:

$$Corr(x[n], h[n]) = \sum_{k=0}^{\infty} (h[k]x[n+k])$$

First- and Second-Order ΣΔ-Modulator

The linear model of the first-order ΣΔ-modulator is shown in Figure 4-10.

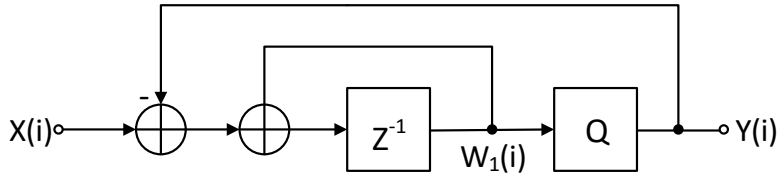

Figure 4-10: First-order ΣΔ-modulator model implemented in MATLAB.

The implemented equations that belong to this model are given by:

$$W_1(i) = W_1(i-1) + X(i-1) - Y(i-1)$$
$$Y(i) = Q(W(i))$$

The second-order modulator has a similar model:

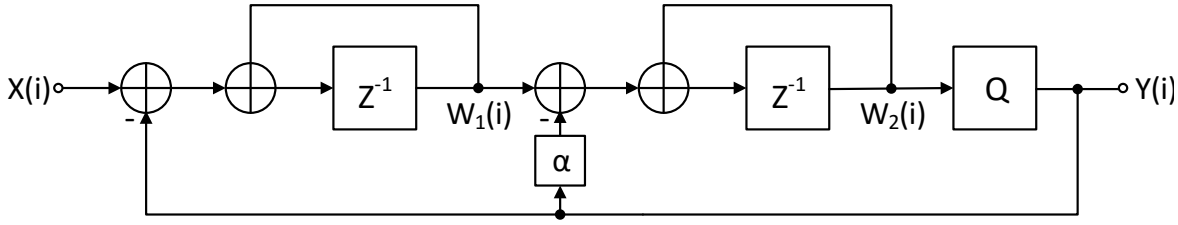

Figure 4-11: Second-order ΣΔ-modulator model implemented in MATLAB.

The implemented equations are given by:

$$W_2(i) = W_2(i-1) + W_1(i-1) - \alpha Y(i-1)$$
$$W_1(i) = W_1(i-1) + X(i-1) - Y(i-1)$$
$$Y(i) = Q(W(i))$$

The corresponding code can be found in Appendix A under the `sigmadelta1` and `sigmadelta2` functions.

4-3-3 System Simulations

In this section we will discuss the simulation results that were obtained from the MATLAB models.

Effect of Noise

There are two types of noise that we need to keep in mind when examining the noise within this simulator. The first type is the white-noise that is superimposed on top of the received

signal. This noise is added onto every (16 MHz) sample in the form of a normally distributed sequence with a standard deviation that is set in the simulator. From the simulations we learn that for this type of noise, an SNR of 0 dB is sufficient to accurately determine the ToF with the the cross-correlation operation as displayed in Figure 4-12.

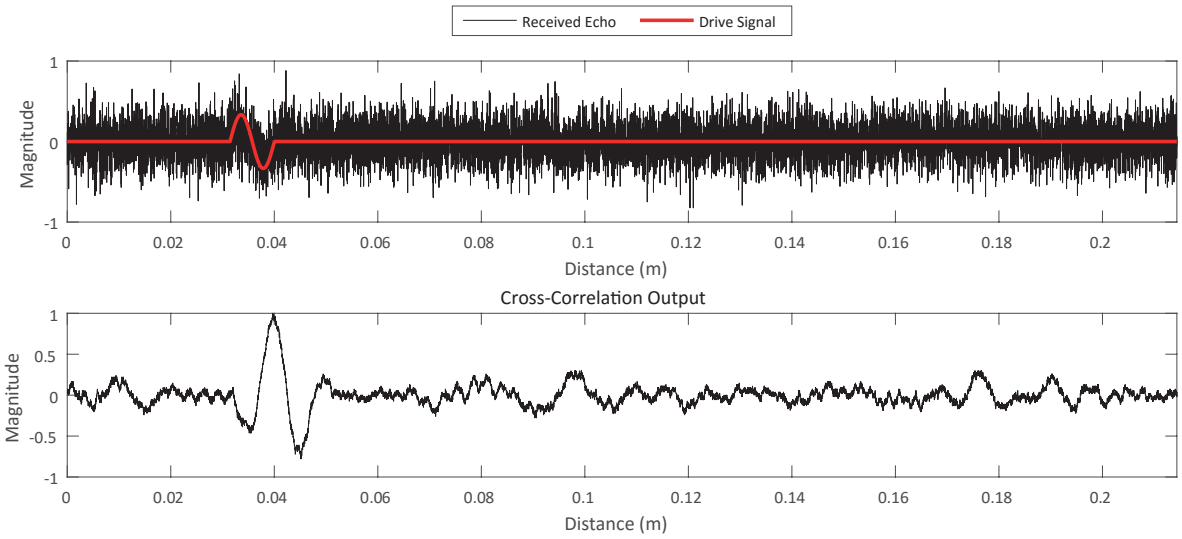

Figure 4-12: Cross-correlation output for a highly noise corrupted signal.

Note that the result displayed here is for a first-order ΣΔ-modulator with a 1-bit quantizer in combination with a 1-bit reference signal. Furthermore, a low-Q transducer was used to reduce ringing.

The other type of noise is quantization noise, which originates from the quantizer. The quantization noise (or better quantization error) is introduced when the analog input voltage of the quantizer is transformed into a digital value. The quantization noise can be reduced by increasing the OSR, which causes the quantization noise to spread out over a wider spectrum. Furthermore, the resolution of the quantizer can be increased to reduce the quantization error. Finally, the quantization noise can be reduced by increasing the order of the ΣΔ-modulator, which increases the effect of noise-shaping.

In our system, the signals grow smaller as the distance increases. This means that the quantization error increases as the distance increases. In order to reduce this effect a time-dependent gain is needed. This time-dependent gain is implemented in the feedback DAC, which will be discussed in chapter 5.

Order of the ΣΔ-Modulator

Increasing the order of the ΣΔ-modulator reduces the quantization noise. The spectrums of the bitstreams of the first- and second-order ΣΔ-modulators are shown in Figure 4-13

The noise-shaping slope depends on the order of the modulator. In the left plot, the dashed line represents a first-order slope (20 dB/decade) and in the right plot the dashed line represents a second-order slope (40 dB/decade).From this plot it is clearly visible that increasing the order of the ΣΔ-modulator increases the effect of noise shaping. The cross-correlation

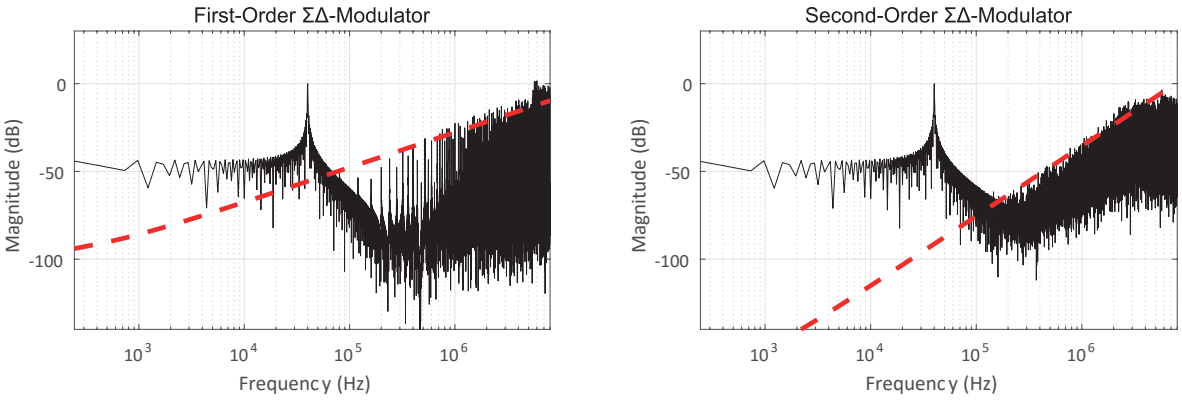

Figure 4-13: Spectrum of the implemented ΣΔ-modulators.

operation is able to recover the incoming signal for SNRs as low as 0 dB. A first-order modulator clearly suffices in that spec. It would therefore be a waste to allocate resources to implement a second-order modulator.

Over-Sampling-Ratio

Assuming the quantization noise is white, its PSD is given by [25]:

$$S_e = \frac{e_{rms}^2}{F_s/2} \tag{4-12}$$

Where e_{rms}^2 is the mean square value of the quantization error and F_s is the sampling frequency. The over-sampling ratio is defined as:

$$OSR = \frac{F_s}{2F_{bw}} \tag{4-13}$$

Where F_{BW} is the signal bandwidth. The total in-band noise power then becomes:

$$q_{rms}^2 = S_e \cdot F_{bw} = \frac{e_{rms}^2}{OSR} \tag{4-14}$$

A doubling of the OSR decreases the in-band noise power by 3 dB.

As shown in Equation 4-3b, ΣΔ-modulators act as high-pass filters for the quantization noise. When the OSR $\gg 1$, a good approximation for the in-band quantization noise power is given in [21]:

$$q_{rms}^2 = \frac{\pi^2 e_{rms}^2}{3OSR^3} \tag{4-15}$$

It is easily seen that when the OSR is increased by a factor of 2, the in-band quantization noise power is reduced by a factor of 8 (9 dB). This increased effect is due to the noise-shaping properties of the ΣΔ-modulator. For higher order modulators the shaping effect of noise shaping is further increased, leading to even lower in-band quantization noise power.

In addition to reducing the quantization noise, increasing the OSR also increases the number of samples that are correlated. The correlation function is implemented as a FIR filter. Increasing the OSR and thus also the number of filter taps leads to an improved filtering operation. This increased filtering allows for better signal reconstruction. We have chosen for a relatively high OSR of 200 ($F_s = 16$ MHz) because it allows us to recover the ToF from echoes with SNRs as low as 0 dB.

$\Sigma\Delta$-Quantizer and Reference-Resolution

As can be expected, increasing the $\Sigma\Delta$ quantizer resolution, or increasing the resolution of the cross-correlation reference signal further increases the SNR of the cross-correlation operation. In Figure 4-14, four different configurations are depicted in which we have zoomed in on the peaks of the cross-correlation output signal.

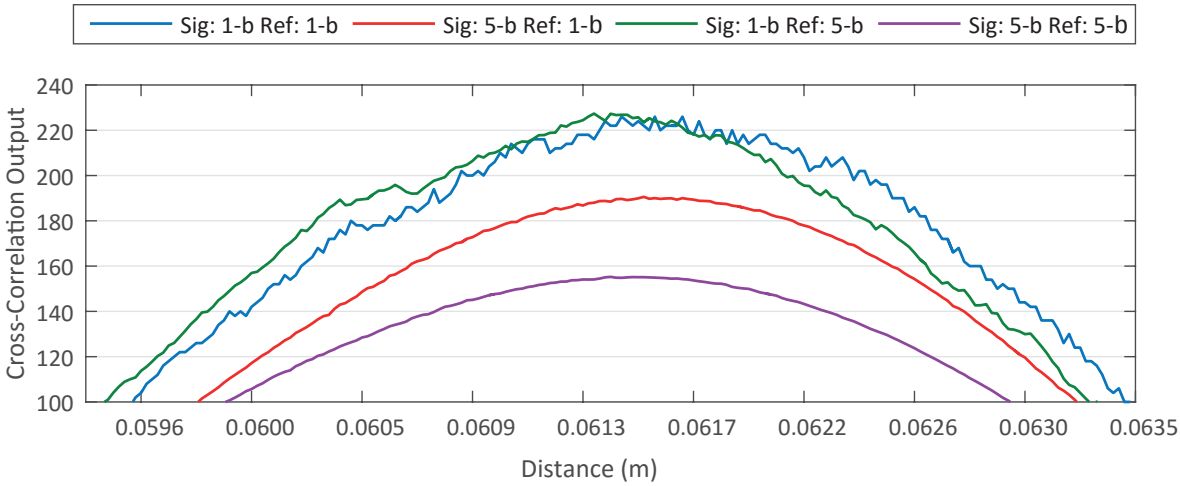

Figure 4-14: Cross-correlation output for four different system configurations (zoomed in at the peak).

A normally distributed noise signal with a standard-deviation equal to the signal amplitude and a bandwidth of 16 MHz was superimposed on the received signal.

Increasing the reference resolution leads to an increased SNR because the filter taps are adjusted such that the low signal amplitudes that are most susceptible to noise, contribute less to the cross-correlation output than the high signal amplitudes. Increasing the $\Sigma\Delta$ quantizer resolution however also reduces the quantization noise in addition to becoming less susceptible to the white-noise making this the favorable option if one of the two has to be chosen.

Instead of looking at the peaks of the cross-correlation signal we could look at the zero-crossings. The zero-crossings are displayed in Figure 4-15 and are seemingly less affected by noise.

If we compare both results, we can see that the measurement inaccuracy for a 1-bit reference in combination with a 1-bit quantizer is significantly reduced if instead of the peaks, the zero-crossings are used as the reference points. This is because the zero-crossings are less affected by noise because the derivative of the cross-correlation output is highest at those points.

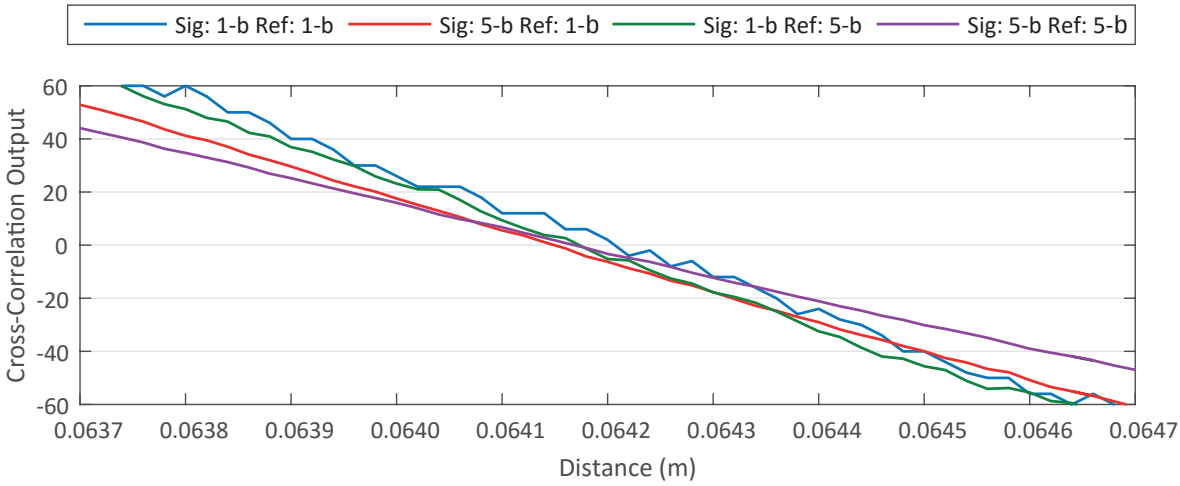

Figure 4-15: Cross-correlation output for four different system configurations (zoomed in on the zero-crossing following the maximum).

Number of Transmitted Periods and Number of Reference Points

Increasing the number of transmitted periods leads to an increase in the amplitude of the transmitted signal because more energy is added to the transducer. Furthermore, it increases the amount of transmitted pulses in making it harder to detect the ToF. It is best to send just one period, to reduce the amount of ringing in the received echo. Increasing the number of reference points for the cross-correlation operation, is directly related to the OSR. When the OSR is increased, the number of reference points should be increased accordingly because the center frequency of the filter depends on the sampling frequency. There is no point in increasing the number of reference periods beyond the number of transmitted periods however, since the surplus periods of the reference signal will be correlated against unwanted signals that do not correspond to the transmitted pulses. Because an OSR of 200 is used, resulting in a sampling rate of 16 MHz, the number of reference points should be set to 400 to correspond with a 40 kHz bandwidth. The first half of the reference points should be set to +1 and the other half to -1 in order to create a bandpass filter around 40 kHz.

Effect of Quality Factor

Depending on the transducer Q, the transmitted signal will show a certain amount of ringing as explained in section 2-4 and displayed in Figure 2-4. Depending on the Q of the transducer used, the peak in the cross-correlation signal could be at the end of excitation or any arbitrary number of periods later. In Figure 4-12 the results for a low-Q transducer were shown. In Figure 4-16 we show the results for a high-Q transducer.

In this plot we can see that it is hard to determine the maximum of the cross-correlation operation for high-Q transducers because of the large amount of ringing. We therefore expect the the system to have a better performance for low-Q transducers than it does for high-Q transducers.

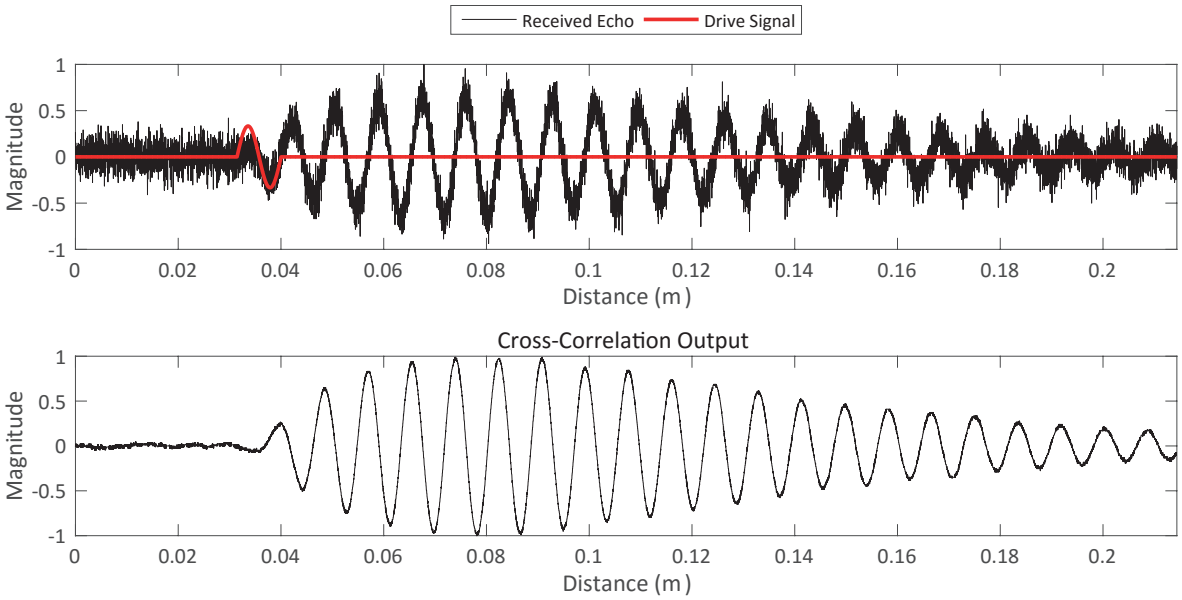

Figure 4-16: Cross-correlation output for high-Q transducer.

Chirp Signal

The cross-correlation operation becomes increasingly more effective when a signal is transmitted that has a very distinct signature. Simply transmitting a pulse with a constant frequency will, as we know, result in a signal of that same frequency with a certain amount of ringing that is associated with the quality factor of the transducer. The ToF is hard to extract from such a signal if a transducer with a high Q is used because of the ringing. If a chirp signal is transmitted, the zero cross-points of the transmitted signal are not spaced equally in time and the cross-correlation operation becomes less dependent of the amplitude and amount of ringing. A chirp signal was simulated with 10 periods and swept from 25 kHz to 55 kHz. The result is shown in Figure 4-17.

In order to transmit a signal with varying frequency however, a low Q transducer is required. Such a transducer would show no ringing in the first place, such that a chirp signal would not be required.

From this we can conclude that the cross-correlation operation is best used in combination with a low-Q transducer. In that configuration the transducer will produce the most reliable results.

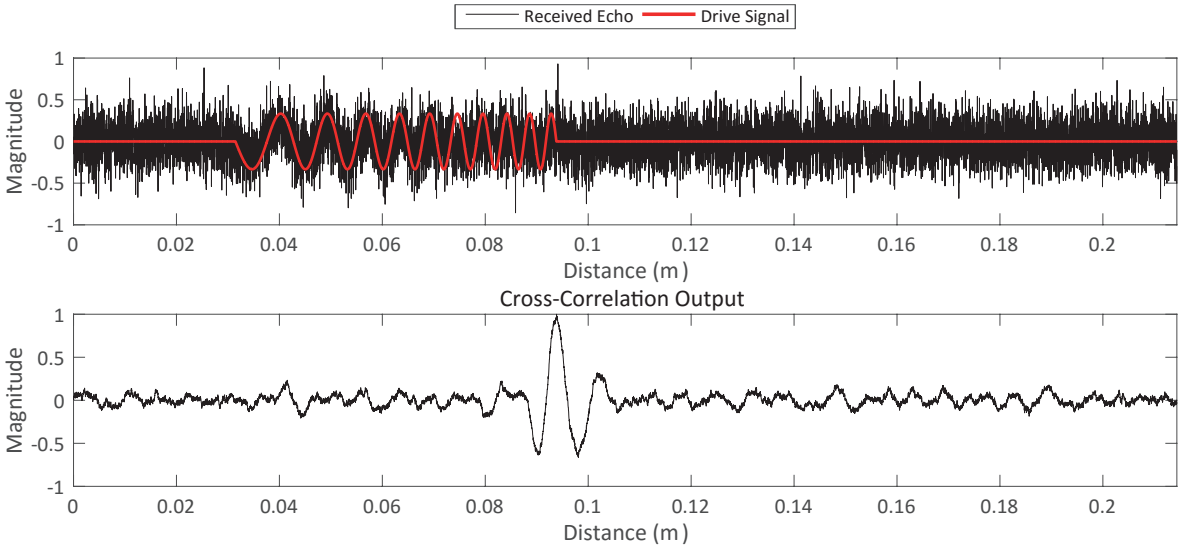

Figure 4-17: Cross-correlation output for a 10-period chirp signal swept from 25 kHz to 55 kHz.

Chapter 5

Circuit-Level Implementation and Simulation

In this chapter the circuit level implementation of the system will be discussed. First, an overview of the complete circuit implementation is given. This is followed by the transistor-level implementation of its sub-blocks. When all the transistor-level implementations have been treated, the layout of the complete chip is shown, in which the various sub-blocks will be identified. Finally, circuit-level simulations are presented and analysed.

5-1 Circuit Level Implementation

In this section the circuit level implementation of the most important blocks in the design is discussed. The section starts by giving an overview of the circuit level implementation, followed by the implementations of the individual blocks. First the pre-amplifier is introduced, followed by the comparator and the clock delay circuits. Next the feedback DAC, bias circuitry and pre-charge circuits are discussed. Note that in these sections, some current mirrors have been omitted from the schematic drawings to emphasize the functional behaviour of the systems. The layout of the design is shown last.

5-1-1 Overview

An overview of the circuit implementation is shown in Figure 5-1. The implementation contains an internal as well as external part. The internal part is integrated into the chip, whereas the external part of the circuit is implemented on a PCB. The internal part of the implementation is basically a first-order $\Sigma\Delta$-modulator, where the external piezo-electric transducer acts as a passive integrator. The external part contains the piezo-electric ultrasonic transducer, the driver that excites the transducer and high-voltage switches that protect the internal circuit while the transducer is being excited. To reduce the overall area, internal TRX switches could be implemented in future designs. However due to a tight tape-out deadline the decision was made to not include them at this time.

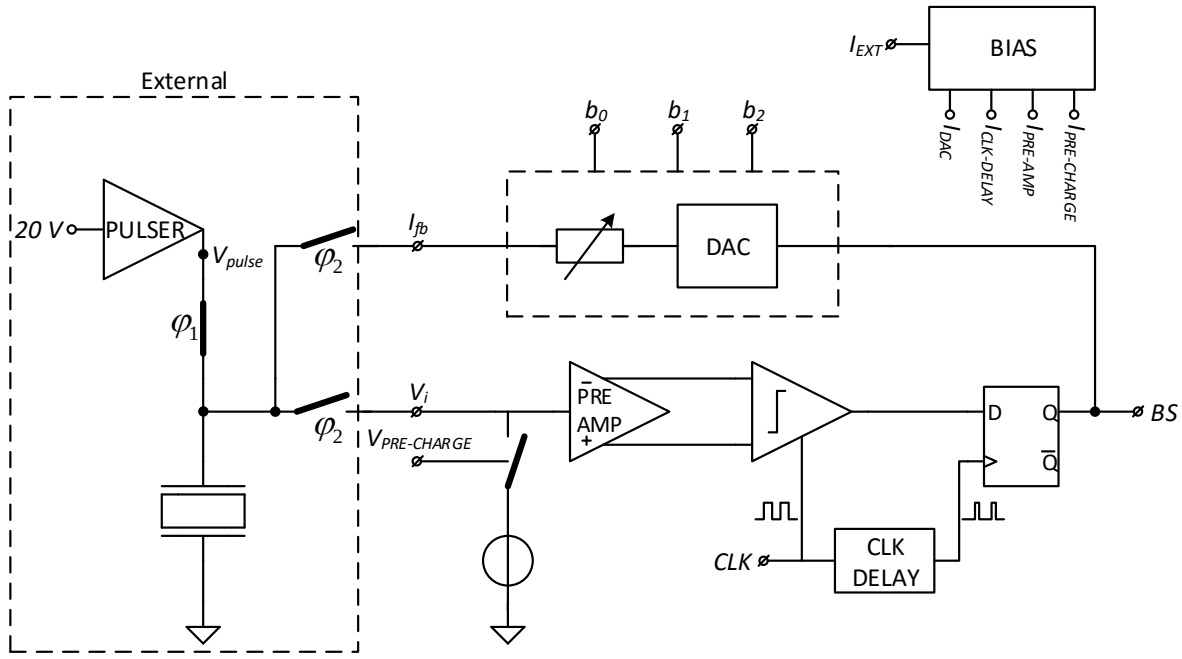

Figure 5-1: Overview of the circuit level implementation.

The circuit contains a pre-amplifier followed by a comparator. The output signal of the comparator goes into a positive-edge triggered D Flip-Flop (DFF) which samples the comparator's output when it has settled. To ensure that the signal has settled when it is sampled by the DFF, a delayed clock is generated for the DFF by the CLK DELAY block. The DFF's output is the bitstream that corresponds to the received input signal. This bitstream signal is sampled by an FPGA and eventually processed in MATLAB; it also goes into the DAC to implement negative feedback. The DAC consists of a voltage source made with a class-AB amplifier, resistor ladder and 3-to-8 decoder for the selection of the correct feedback strength. A bias block provides currents to the DAC, CLK-DELAY, PRE-AMP AND PRE-CHARGE circuits to ensure correct operating points. An overview of the chip's pins is given in Table 5-1.

Table 5-1: GJALLARHORN node information.

Node	Information
VDD	5 V
GND	Ground
I_{EXT}	3 μA
$V_{PRE-CHARGE}$	'1' = on, '0' = off
CLK	16 MHz clock generated by a PLL on an FPGA
V_i	The input of the $\Sigma\Delta$-modulator
BS	The bitstream output of the $\Sigma\Delta$ modulator
I_{fb}	The feedback current generated by the DAC
b_0	bit-0 of the feedback strength selection scheme
b_1	bit-1 of the feedback strength selection scheme
b_2	bit-2 of the feedback strength selection scheme

5-1-2 Pre-Amplifier

Early circuit simulations have shown us that the comparator requires a pre-amplifier with a gain of 10 at the sampling frequency (16 MHz), in order for it to make a correct decision within one clock period.

To keep area to a minimum we have chosen to use an NMOS input stage. A common-source stage with a resistive load can be used to achieve a gain of 10. However, the output bias voltage is not well defined for such a structure, which is problematic because it has to be compared to the reference voltage of the quantizer.

In order to circumvent this problem, a differential pair can be used in which the input nodes are connected to V_{in} and V_{ref}. The downside of the differential pair is that the input common mode voltage will shift up by at least one V_{DSsat}. If the input common mode voltage is increased, the transducer will have to be charged up to a higher voltage after it has been excited by the pulser. This would effectively increase the time required to pre-charge the transducer back to the input common-mode level, hence increasing the minimum detection range of the system.

In order to reduce the input common-mode level, we decided to bias the differential pair from the top. The resulting circuit is shown in Figure 5-2.

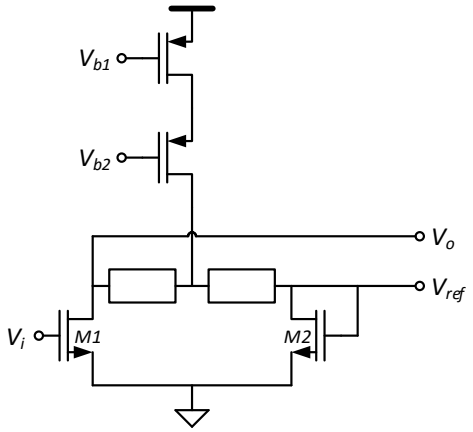

Figure 5-2: Circuit level implementation of the pre-amplifier.

On the left side, the pre-amplifier accepts the output voltage of the transducer. On the right side a reference voltage of one V_{GS} is generated by connecting the gate and drain of the transistor. The Resistors are chosen such that the differential gain equals 10. When $V_i = V_{ref}$ the differential output voltage is 0. When $V_i > V_{ref}$, V_o decreases and V_{ref} increases.

In order to be able to measure small signal amplitudes, the noise generated at the input of the pre-amplifier should be lower than the noise generated by the transducer ($V^2_{n,R_s} = 4kTR_s$). The most significant noise source for MOSFETs is generated in the channel and is modeled by a current source connected between drain and source with a spectral density $\overline{I_n^2} = 4kt\gamma g + m$ where γ is derived to be 2/3 [26]. The input-referred noise voltage of the pre-amplifier can be approximated by the noise generated in both channels of the differential pair which can be approximated by:

$$\overline{V_{n,in}^2} = 8kT \cdot \frac{2}{3} g_m \tag{5-1}$$

For an R_s of 1 kΩ the required transconductance equals $g_m = 1.3$ mS. To achieve this transconductance, the top current sources are biased at 300μA and the width of transistors M1 and M2 was set to 150 μm.

5-1-3 Comparator

For this design we have chosen to implement the dynamic comparator with zero static power consumption [27] shown in Figure 5-3.

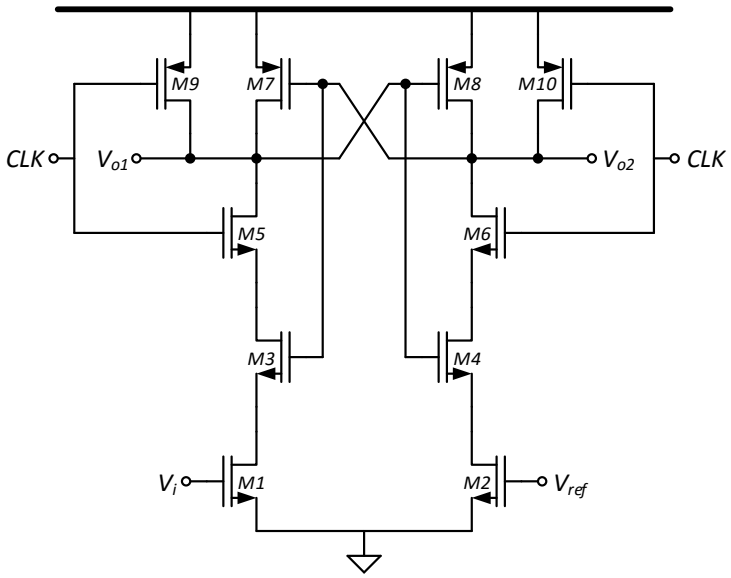

Figure 5-3: Circuit level implementation of the dynamic comparator.

The comparator is triggered by a 16 MHz clock signal. When the CLK signal turns low, the comparator enters a reset state and transistors M9 and M10 turn on. This causes both V_{o1} and V_{o2} to be pulled up to V_{DD}. At the same time transistors M5 and M6 turn off, reducing the current consumption of the comparator structure to zero once V_{o1} and V_{o2} have reached V_{DD}.

The comparator enters its active state on the positive edge of the CLK signal. If $V_i > V_{ref}$ the current in the left branch pulls down V_{o1} down faster than the current in the right branch pulls down V_{o2}. Transistors M4 and M8 act as an inverter whose input voltage V_{o1} is rapidly dropping due to the high current running through the left branch and its output voltage V_{o2} is thus rising rapidly as well. In the other branch, transistors M3 and M7 also make an inverter whose input voltage V_{o2} is dropping slowly due to the lower current running through the right branch and its output voltage V_{o1} is rising slowly as well. Eventually, the inverter composed of transistors M4 and M8 win the battle against the inverter composed of transistors M3 and M7 and V_{o1} becomes low and V_{o2} becomes high. In the same manner V_{o1} becomes high and V_{o2} becomes low if $V_i < V_{ref}$.

5-1-4 Clock Delay

Because the comparator is positive edge triggered and resets to a default high state on the negative edge it is important to sample the output at the end of active phase. A positive edge-triggered DFF is used to sample and hold the output value of the comparator during the remainder of the clock period. Because the comparator requires some time in order for its output to make a decision, the DFF requires a clock with a rising-edge that is slightly delayed. This delayed clock is implemented by the following circuit:

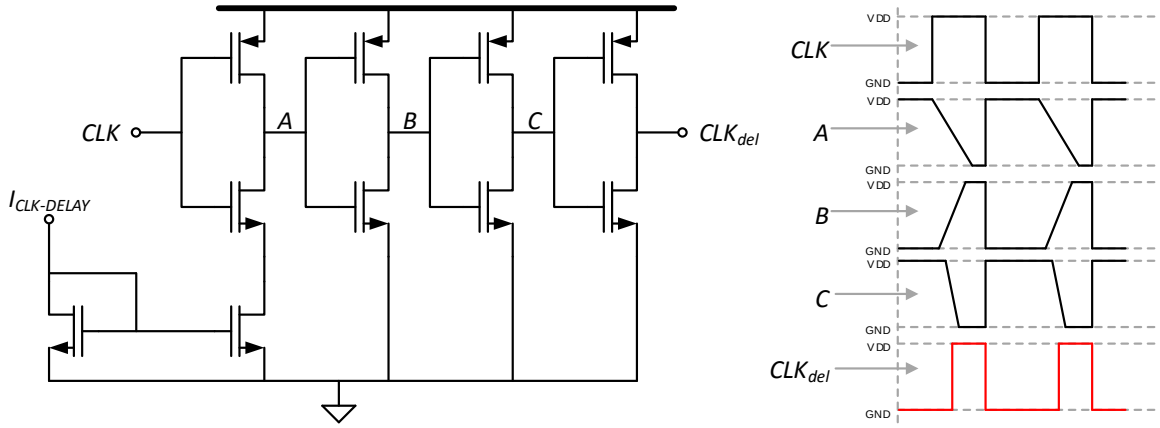

Figure 5-4: Circuit level implementation of the delayed clock for the DFF.

The external clock signal is offered as an input signal to an integrator of which the NMOS speed is limited by the current $I_{CLK-DELAY}$, supplied by the BIAS block. This results in slewing falling edge on node A. Moving from node A to the final node CLK_{del} the integrators have an increased drive strength and the speed of their NMOS transistors is no longer limited by a bias current. The integrators will amplify the slewing signal until it clips at the output node resulting in a clock signal with a delayed rising edge, but otherwise unaltered falling edge. The falling-edge is not of interest since we are using a positive-edge triggered DFF.

5-1-5 Feedback DAC

Sound intensity decreases quadratically as explained in section 2-5; this means that for the targeted distances between 10 cm - 10 m (Table 1-1) the sound intensity will vary by roughly 60 dB. The amplitude of the signal output by the transducer is directly proportional to the sound intensity. At longer distances the received signal would be small in magnitude, which would cause the quantization noise to increase. In order to keep the quantization noise from dominating the transfer of the system it is necessary to implement a scalable feedback strength that depends on the traveled distance. We do this by implementing a DAC with a scalable feedback current, the implementation of which is shown in Figure 5-5.

It is important to provide feedback currents that are identical in magnitude but opposite in sign for high and low output voltages of the ΣΔ-ADC. Since the input bias level of the pre-amplifier is set to V_{GS} and not $V_{DD}/2$, we cannot simply use GND and VDD as the two reference voltages that we source or sink currents from/to. A $2V_{GS}$ voltage source is made by stacking two diode-connected transistors and connecting them to a class-AB buffer.

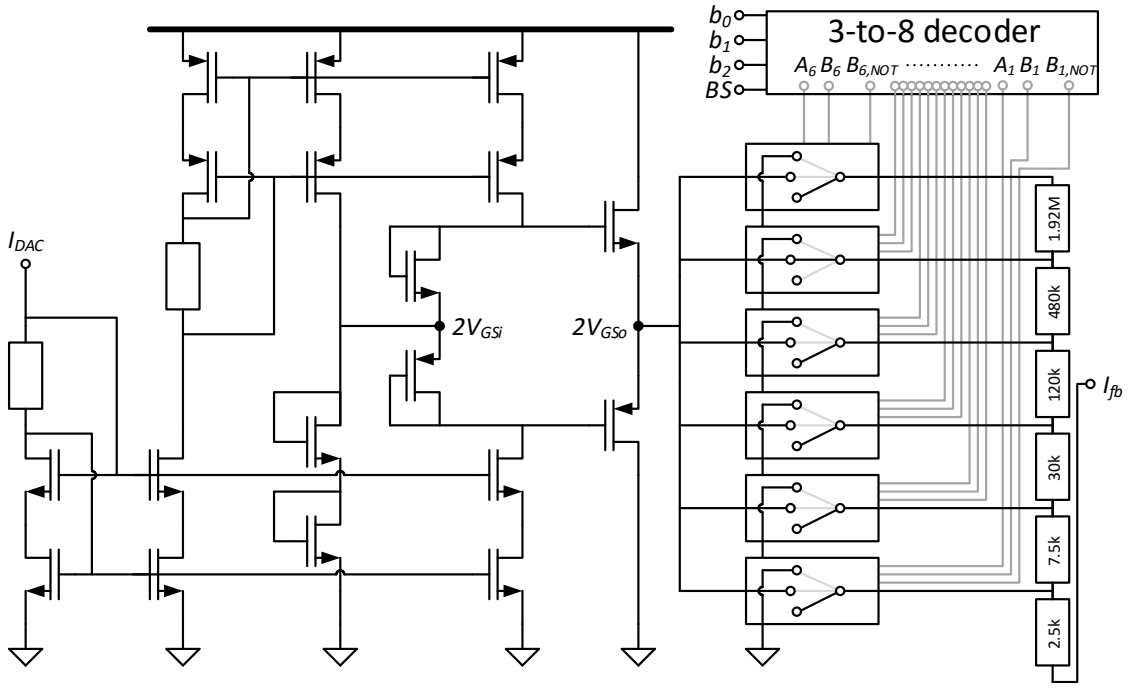

Figure 5-5: Circuit level implementation of the DAC.

Biasing with the same current density ensures that the V_{GS} of these two transistors is equal to the V_{GS} of the pre-amplifier. The class-AB buffer acts as a voltage source, ensuring that its output voltage remains constant when current is being sourced to the transducer.

The implementation of the tri-state switches that are connected to the output of the class-AB buffer is shown in Figure 5-6.

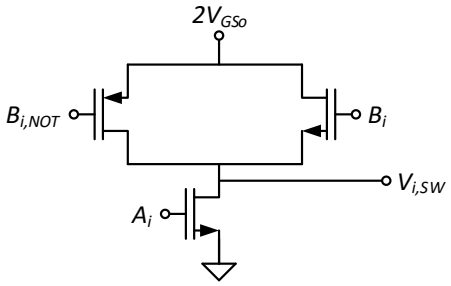

Figure 5-6: Circuit level implementation the tri-state switches used in the DAC.

Table 5-2: Truth table that belongs to the tri-state switch.

A_i	B_i	$V_{i,SW}$
0	0	$Floating$
0	1	$2V_{GSo}$
1	X	GND

The switches have the following three states: GND, $2V_{GS}$ and $Floating$ and are controlled by the signals A_i, B_i and $B_{i,NOT}$ that are output by a 3-to-8 decoder. The corresponding truth table is shown in Table 5-2. The decoder is in turn controlled by the three external signals b_0, b_1 and b_2 and by the sign of the bitstream output signal of the system BS. The three external signals control the DAC feedback strength and are set by an FPGA. They are programmed in such a way that when the time since transmission increases by a factor of 2, the resistance in the feedback path is increased by a factor of 2^2. The feedback current

therefore reduces with a $1/d^2$ fashion with respect to time. This was done because the sound intensity of the transmitted signal reduces in the same $1/d^2$ fashion. Depending on the chosen resistance and the sign of BS, one switch is connected to either the output of the class-AB buffer that provides $2V_{GSo}$ or to ground. All other switches are floating.

5-1-6 Bias Block

A biasing circuit was implemented that is driven by an external current I_{EXT} with a value of 3 μA. The implementation is shown in Figure 5-7.

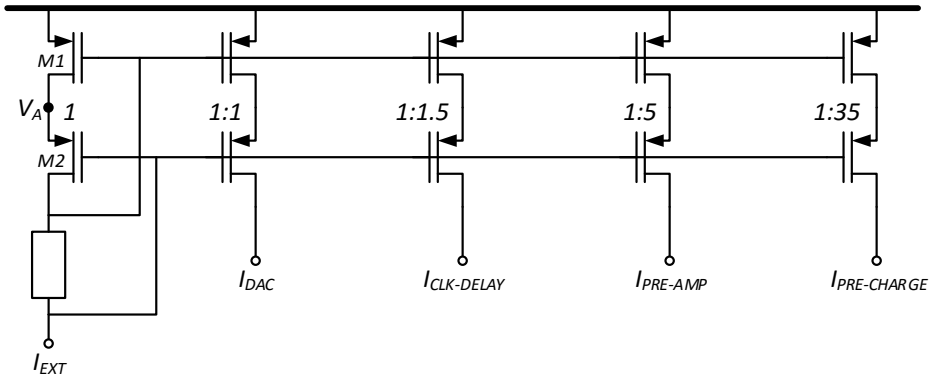

Figure 5-7: Circuit level implementation of the bias block.

To reduce the effect of channel length modulation, cascode current mirrors have been implemented. For minimal voltage headroom consumption, while still keeping the top transistors in saturation V_A should be set to maximally $V_{DD} + (V_{GS1} - V_{TH1})$. V_{G2} should therefore set to $V_A + V_{GS2} = V_{DD} + (V_{GS1} - V_{TH1}) + V_{GS2}$. Since $V_{G1} = V_{DD} + V_{GS1}$, the voltage drop over the resistor should be set to $|V_{GS2} - V_{TH1}|$ which is approximately one overdrive voltage. The value of the implemented resistance is 150 kΩ and the value of $I_{EXT} = 3\mu$A resulting in a voltage drop of 450 mV, slightly higher than one overdrive voltage, to operate deeper in the saturation region.

5-1-7 Pre-Charge

The function of the pre-charge circuit is to charge the transducer to V_{GS} after it has been excited with a signal with a high voltage coming from the pulser. Although unlikely to have a serious effect on the performance of the circuit, it has been implemented as a precaution measure. The implemented circuit is shown below.

The setup consists of 2 transistors, the top of which acts as a switch that is controlled by an external signal. The bottom transistor is biased with a constant current directly from the bias block and has the same V_{GS} as the pre-amplifier. After the transducer has been excited, V_{PC-EXT} becomes high and the switch starts conducting, pulling the voltage of the transducer up to V_{GS}. The feedback DAC will pull the input-node to V_{GS} even if no pre-charge circuit is used. The pre-charge circuit increases the speed of this process, allowing shorter ranges to be detected.

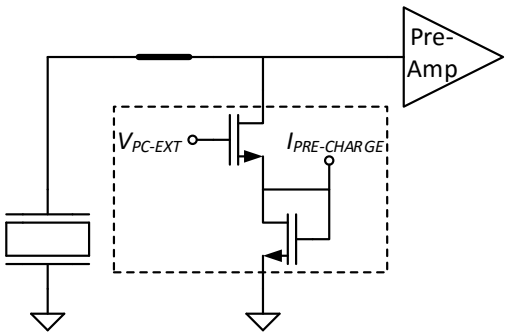

Figure 5-8: Circuit level implementation of the bias block.

5-1-8 Layout of the complete Chip

The layout of the complete chip is shown in Figure 5-9 and has a total area of 1 mm^2. The different blocks are identified with different border colors. The chip is built in a Texas Instruments 0.5 μm CMOS process with 3 metal layers and 5.0 V devices.

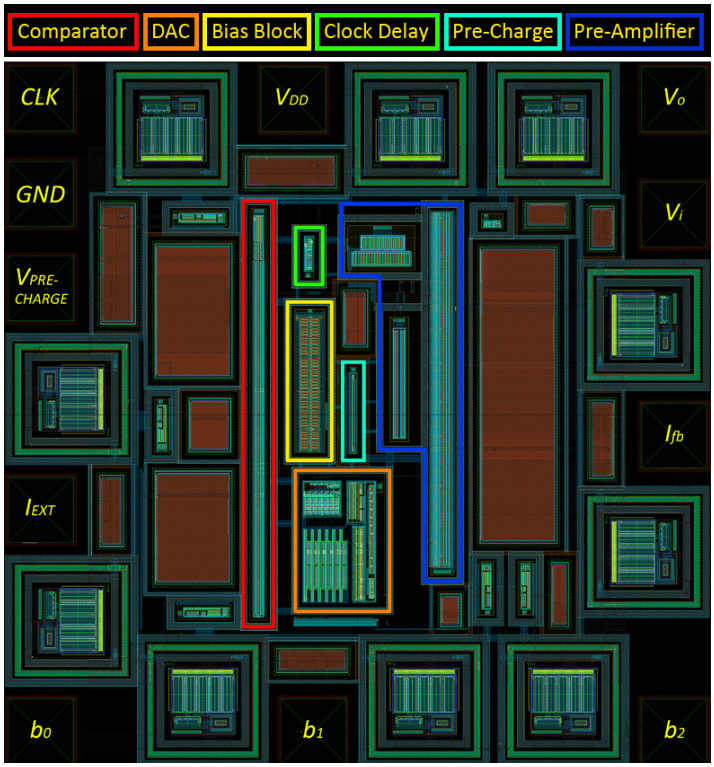

Figure 5-9: Layout of the total system, with a total size of 1 x 1 mm.

The size of the layout had to be pre-allocated early in the project and could not be changed later. Because not all area was used, 15 (grounded) capacitors were added to fulfill the density requirements.

5-2 Circuit Level Simulation

In this section we present some simulation results that show that the circuit is working as intended. We start by doing a noise analysis and finish with a simulation that shows the overall system performance.

5-2-1 Circuit Noise Simulations

The thermal noise level of the system may limit the correct readout of smaller signals, effectively reducing the maximum range of the system. In order to determine the thermal noise level of the system, the largest feedback resistor was selected and the bitstream output spectrum was generated. The result is displayed in Figure 5-10.

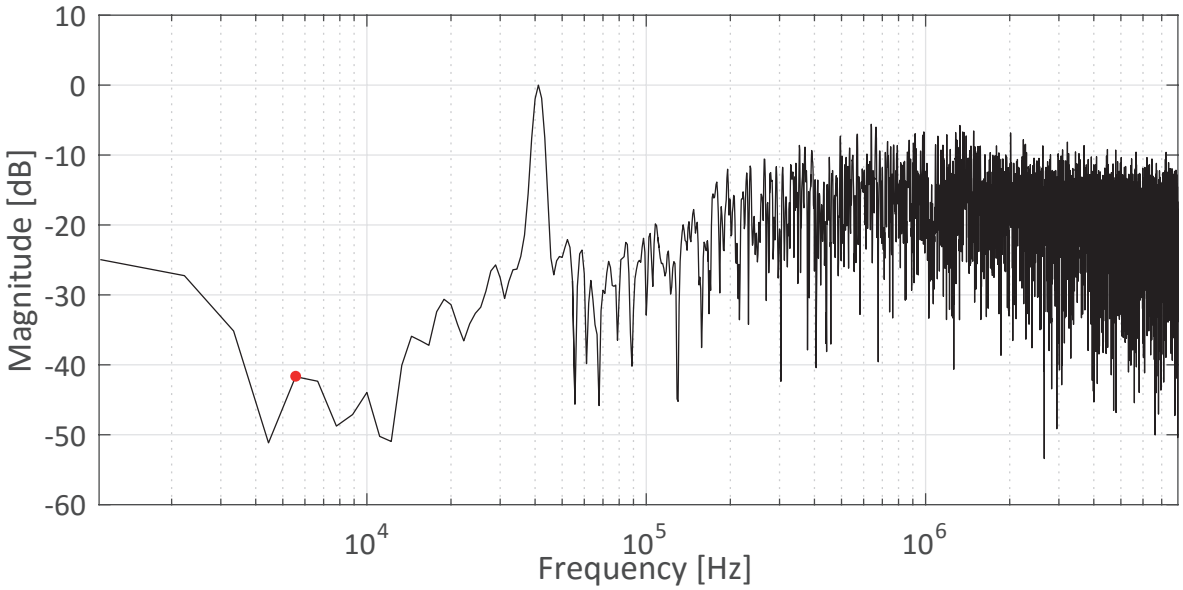

Figure 5-10: Output spectrum for the system for the lowest current feedback setting.

Due to limited simulation time however, the thermal noise level was not reached in simulations. We can however see that the system is quantization noise limited.

In order to show that the system was designed correctly in terms of thermal noise, we derive the maximum thermal noise level of the system. The input amplitude of the 40 kHz input signal is known, which can be used to calculate the energy contained in the input signal. The relationship between the energy contained in the input signal and the energy in the FFT output is given by:

$$\frac{1}{2}V_{in}^2 = \alpha \sum_{signal-bins} FFT[bin] \tag{5-2}$$

Where α is a gain factor introduced by the system and the FFT.

The relationship between the PSD S_n at the input of the system relates to the noise energy in the FFT as:

$$S_n \cdot \Delta f_{noise-bins} = \alpha \sum_{noise-bins} \cdot FFT[bin] \tag{5-3}$$

where $\Delta f_{noise-bins}$ is the bandwidth of the summed noise bins. Rewriting Equation 5-2 and inserting it in Equation 5-3 and taking the square root leads to the PSD:

$$\sqrt{S_n} = \sqrt{\frac{\frac{1}{2}V_{in}^2}{1/f_{noise-bins}} \cdot \frac{\sum_{noise-bins} FFT[bin]}{\sum_{signal-bins} FFT[bin]}} \tag{5-4}$$

Since the thermal noise level has not been reached, we use just one point (marked by the red dot in Figure 5-10) for the noise bin and sum the total energy of the signal bins to contain the total signal energy. The input signal level used was 83 μV and the bin-size was 1111 Hz. Using these values, we can calculate that the PSD of the system generated noise is at least lower than:

$$\sqrt{S_{n,SYS,max}} < 83 \text{ nV}/\sqrt{\text{Hz}} \tag{5-5a}$$

This calculated PSD of the circuit thermal noise is higher than the PSD of the transducer noise (4nV/$\sqrt{\text{Hz}}$. However, if the simulation time is increased, we are likely to find a much lower thermal noise level. We expect the thermal noise level of the circuit to be of the same order of magnitude as the thermal noise generated by the transducer since we designed the channel resistance of the pre-amplifier to be the same as the series resistance in the transducer.

5-2-2 Overall System Performance

In order to verify that the system has an adequate SNR for the complete signal range a transient simulation with transient noise was run. The ΣΔ-ADC's output bitstreams were exported to MATLAB. In MATLAB the bitstreams were passed through the cross-correlation filter. A spectrum generated before and after the cross-correlation operation is shown in Figure 5-11.

The red line represents the slope of the quantization noise for a first-order modulator. It is clearly visible that the system has a first-order behaviour. The cross-correlation filter was implemented by a FIR filter of 200 1's followed by 200 -1's, showing first-order filtering, hence reducing the out-of-band quantization noise. The filter and its frequency response are shown in Figure 5-12

In order to show that the system is capable of digitizing echoes, a continuous-wave sinusoidal signal was used as an input for the system. The same simulation was repeated for all DAC feedback settings. Because each consecutive feedback setting decreases the feedback current by a factor of 4, the input signal was reduced by a factor of 4 as well. The results are shown in Figure 5-13.

These simulation results demonstrate that the implemented system is functional and is capable of digitizing echoes.

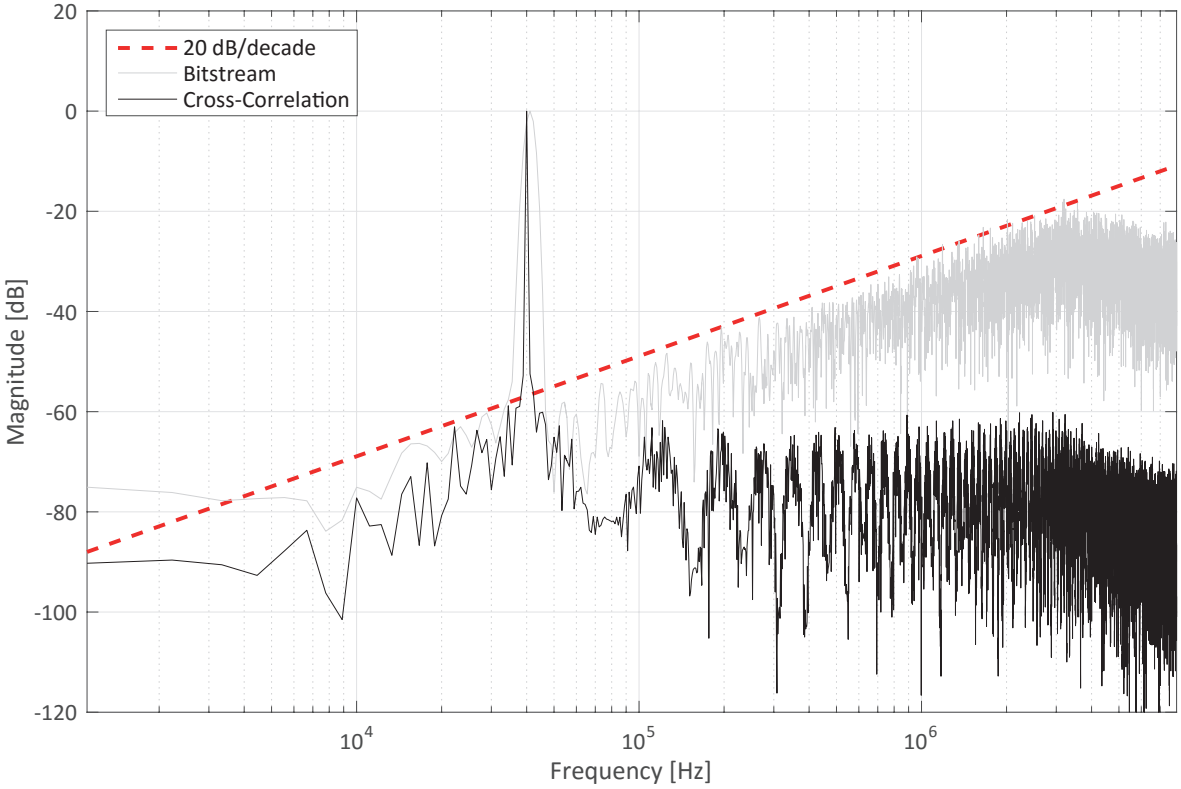

Figure 5-11: Output spectrum for the system before and after the cross-correlation operation.

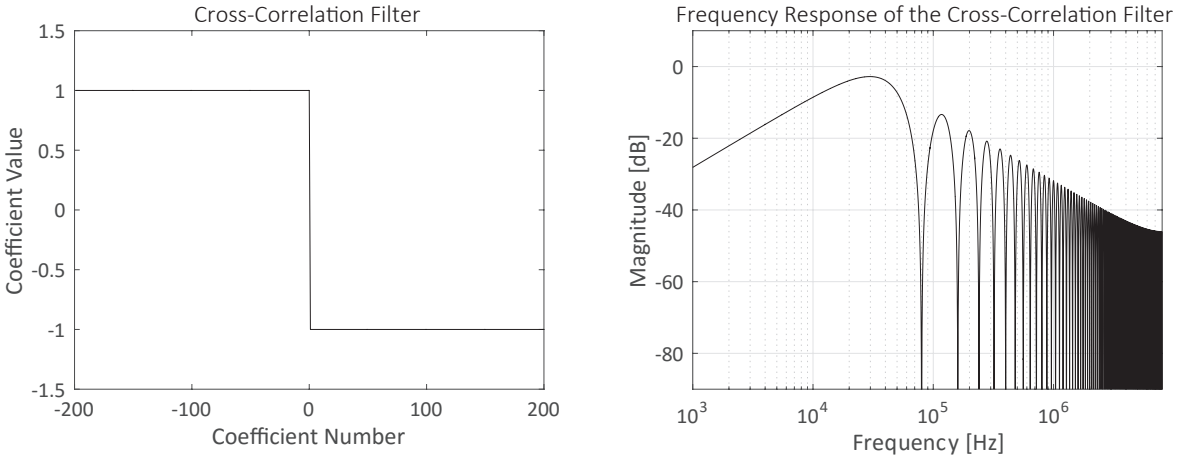

Figure 5-12: The cross-correlation filter and its frequency response.

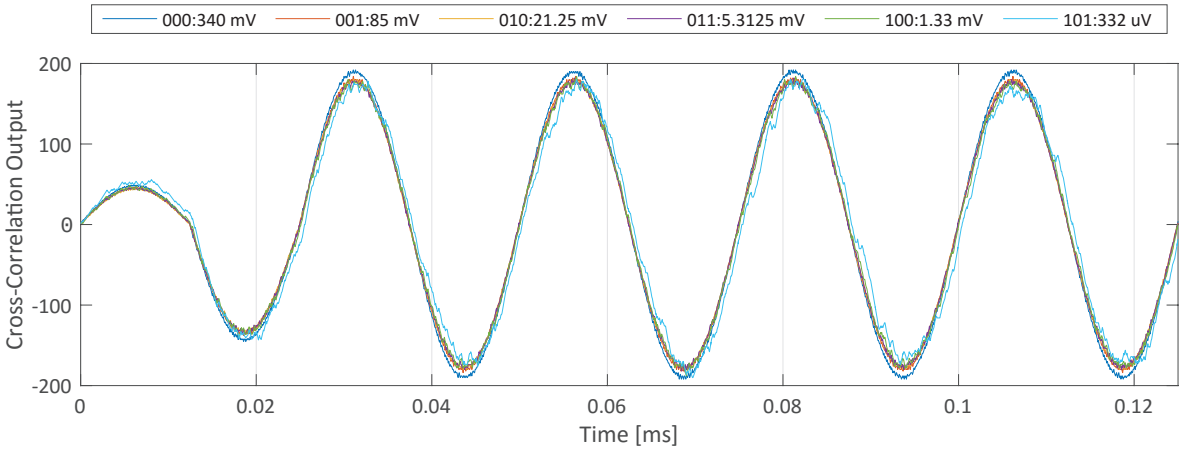

Figure 5-13: Cross-correlation output for a continuous-wave sinusoidal wave applied at the input of the system.

Chapter 6

Measurements

In this section we discuss the measurements that were done on the designed chip. First we discuss the measurement setup. We then present and discuss the measurements results.

6-1 Measurement Setup

The chip is built in a Texas Instruments 0.5μm CMOS process with 3 metal layers and 5.0 V devices. The chip has 11 pins and is packaged in a DIP package and mounted on a PCB. In order to control the chip, some control signals are needed. These control signals are provided by a Cyclone II FPGA development board, which is programmed in VHDL. The FPGA is programmed to sample the bitstream when the transmit button is pressed. The FPGA stores 500000 samples, which corresponds to a range of 10.7 m. When all samples have been stored, they are transferred to a laptop through the UART module. A block-diagram of the measurement setup is displayed in Figure 6-1.

The Cyclone II FPGA code contains several modules. The CLK GEN module generates all the clocks from a crystal oscillator located on the development board. The TRX (Transmit-Receive) module controls the 20 V pulser with the *TX-Pulse* signal. It also controls the external switches of the system that protect the system during transmission. Lastly, it generates the *Pre-Charge* signal that charges the input node up to the input bias level after transmission. The TDG (Time-Dependent Gain) module controls the feedback current generated by the DAC. The SRAM (Static Random-Acces Memory) module samples the bitstream into the SRAM memory located on the development board. The UART (Universal Asynchronous Receiver/Transmitter) module reads data from the SRAM memory and sends it to a laptop. On the laptop, the cross-correlation operation is performed in MATLAB and the final waveforms are obtained.

The piezoelectric transducer used for these measurements was the MCUSD16A40S12RO by MultiComp. From the data sheet [28], a Q of roughly 20 is calculated for this piezoelectric transducer.

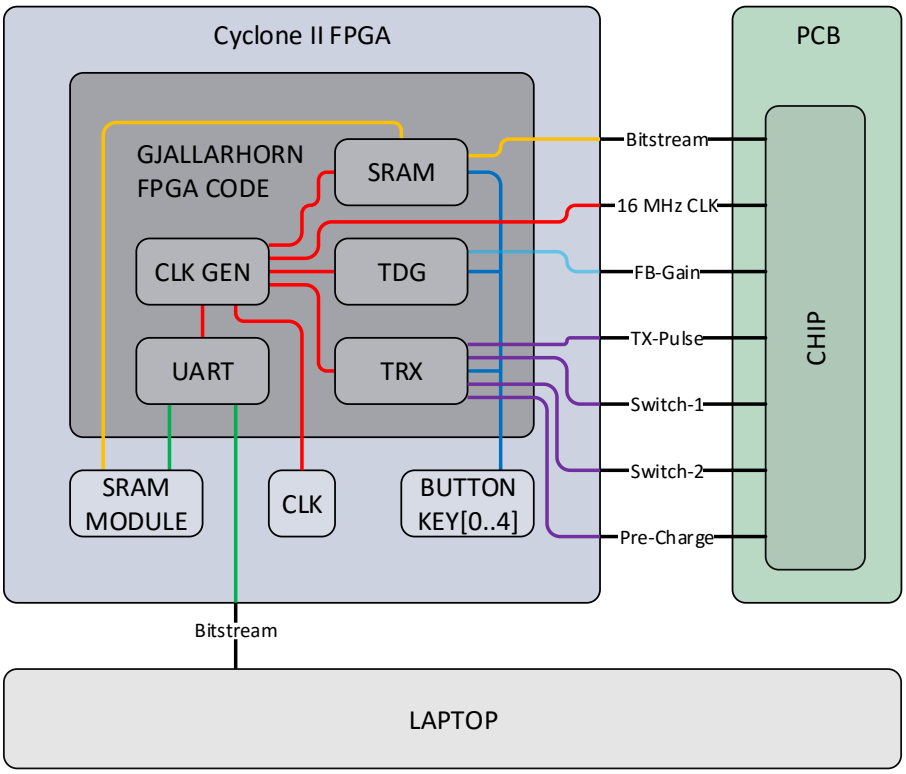

Figure 6-1: Block diagram of the measurement setup.

6-2 Measurement Results

In this section we present two resolution measurements, followed by large displacement measurements. We then show measurements results that were done to confirm the correct behaviour of the feedback DAC. In the summary we list the current consumption of the chip and briefly discuss the measurement results.

6-2-1 Resolution Measurements

Measurement One

A standard-deviation measurement was done to analyze the resolution of the system. 9 Measurements were done where a target was placed at a fixed distance of roughly 60 cm from the transducer. In Figure 6-2 we have zoomed in on the peak of the cross-correlation output that was obtained from the received signals. Calculating the actual distance from these received signals is difficult however, due to the large amount of ringing. As we have shown in chapter 4, the ToF is difficult to determine when a transducer with a high Q is used. Due to time-constraints, we were not able to do measurements with a low-Q transducer.

In order to determine the resolution of the system, we have zoomed in on a zero-crossing to determine the standard deviation of the received signals (Figure 6-3). The calculated standard

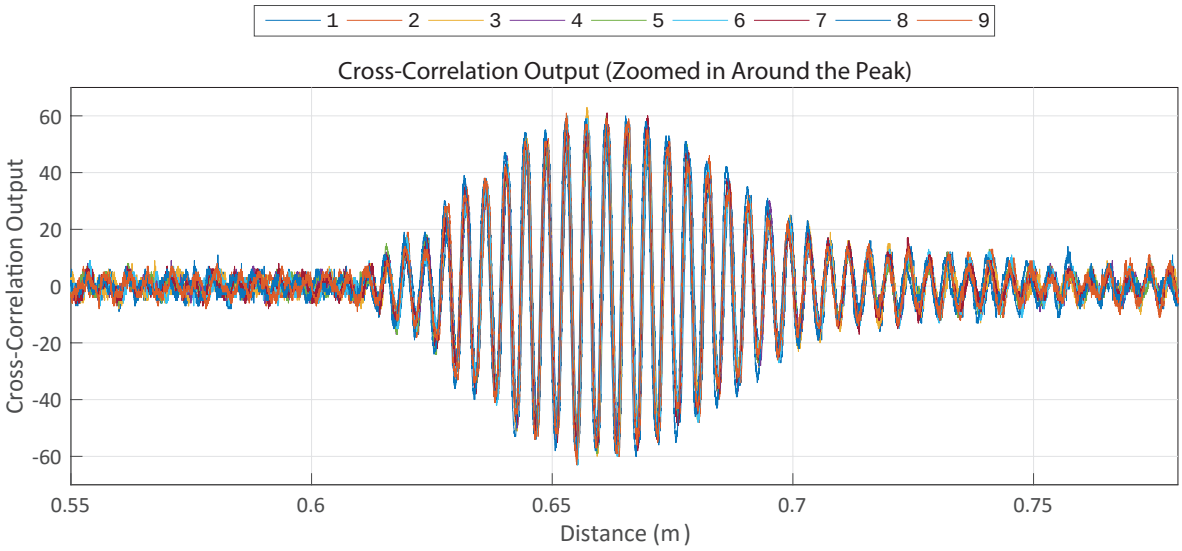

Figure 6-2: The cross-correlation output for the standard deviation measurement.

deviation $\sigma = 0.169$ mm. The system is able to detect zero-crossings with a resolution of far below one wavelength (8.57 mm).

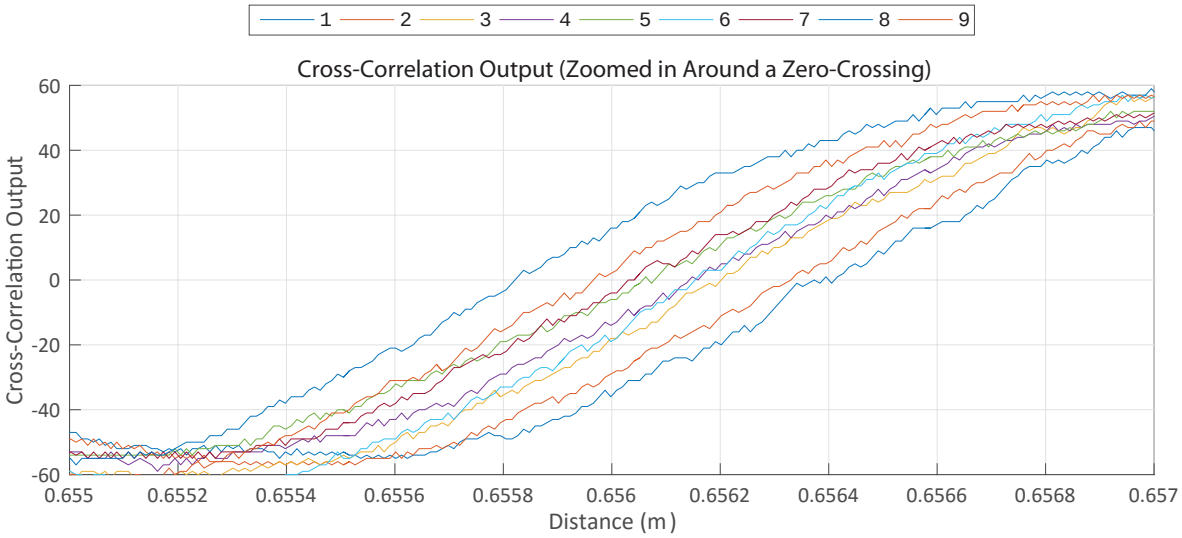

Figure 6-3: The cross-correlation output for the standard deviation (zoomed in on a zero-crossing).

Measurement Two

As stated above, it is difficult to measure the actual ToF due to the high Q of the transducer used. In order to verify the results of the first measurement, a second measurement was carried out in which the target was displaced in steps of 1 mm. The resulting cross-correlation output is shown in Figure 6-4, the extracted displacement data is plotted in Figure 6-5 and the errors are plotted in Figure 6-6.

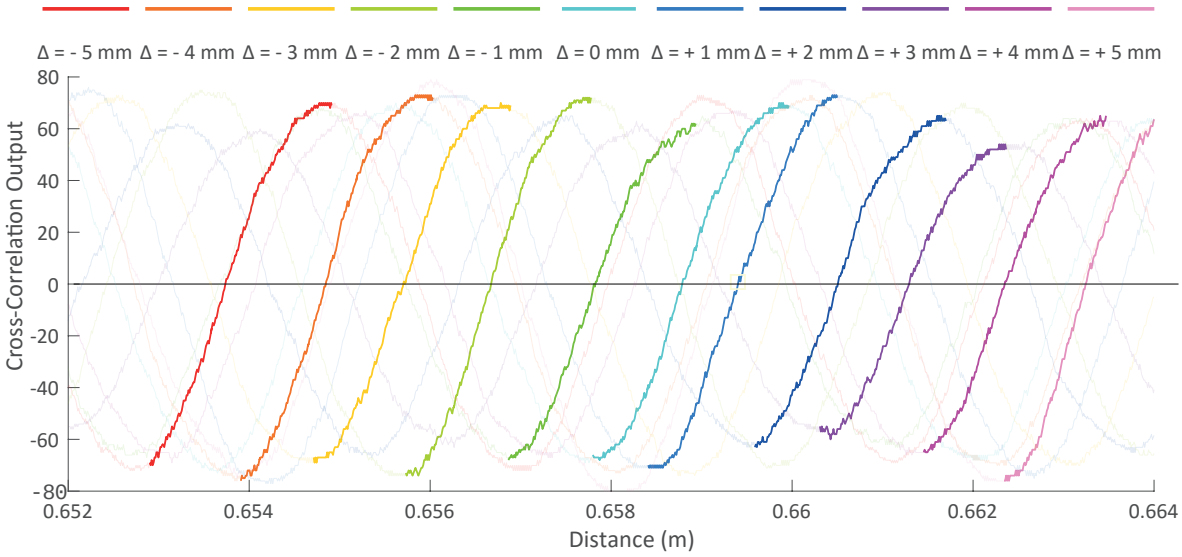

Figure 6-4: The cross-correlation output for the small displacement measurements.

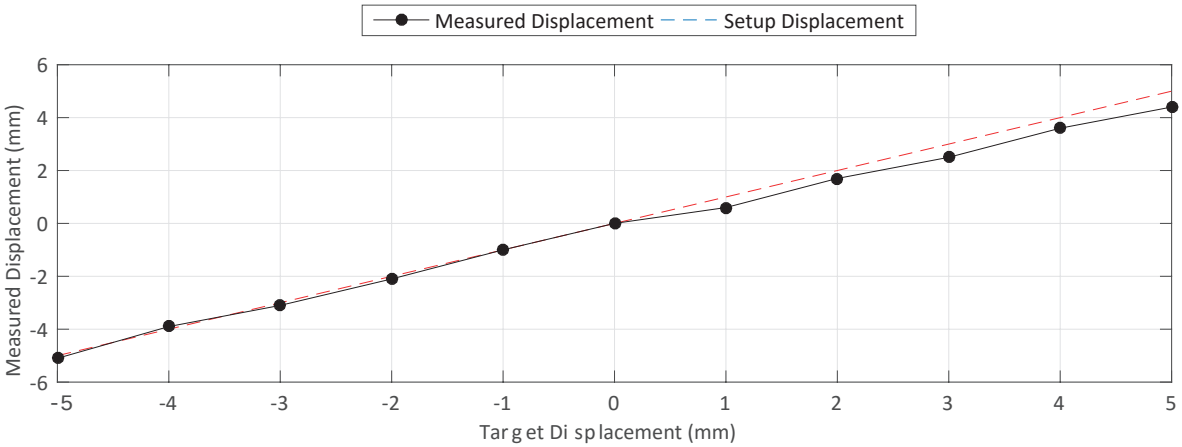

Figure 6-5: The extracted displacement results.

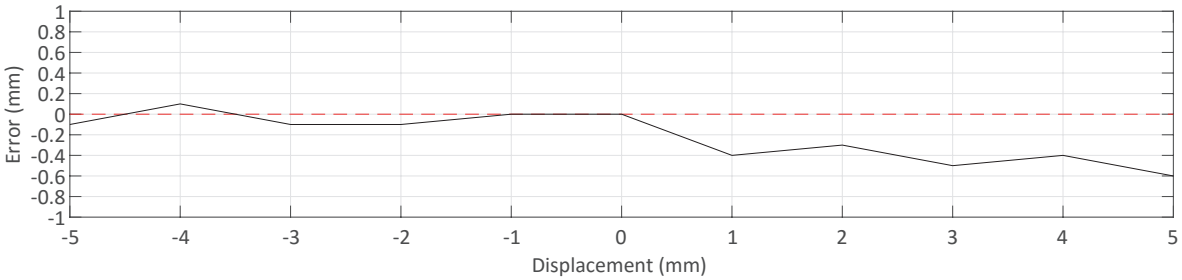

Figure 6-6: The error corresponding to the displacement measurements.

The maximum error that was measured is 0.6 mm. We believe that these measurement errors are mainly caused by the inaccurate measurement approach. Due to time-constraints, the target was displaced by hand in steps of roughly 1 mm, resulting in the displacement errors

seen in Figure 6-6. It seems as if the target was not displaced by the correct distance of 1 mm, between the 0 mm and 1 mm position since the next displacement measurements are all roughly offset by the same amount. In order to measure the actual resolution, future measurements should be carried out in which the displacement is accurately controlled.

6-2-2 Large Displacement Measurements

In order to show that the system is able to measure larger displacements, measurements were carried out in which the target was displaced in steps of 10 cm. The corresponding cross-correlation waveforms are plotted in Figure 6-7.

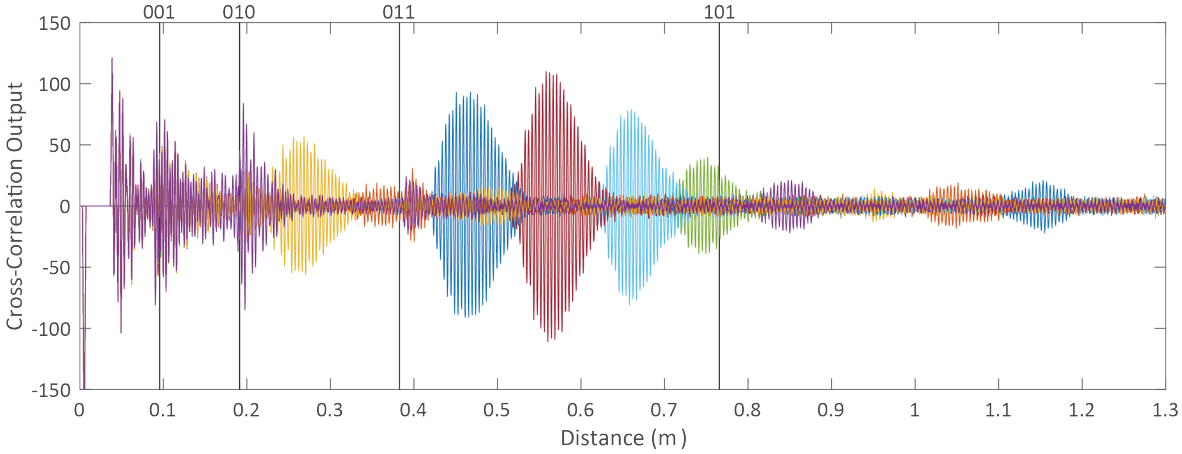

Figure 6-7: The cross-correlation output for the large displacement measurements

It is very difficult to extract the exact distance from Figure 6-7 because of the ringing associated with the high-Q transducer used. By observing the results, we can however see that the distance between the peaks in the pulses is roughly 10 cm. The system is unable to detect signals below 20 cm because the transducer is still ringing after being excited. By using a lower drive voltage, the system should be able to detect these shorter ranges.

The vertical black lines in Figure 6-7 represent the moments in time at which the DAC feedback strength is adjusted. It should be noted that due to an erroneous variable in the FPGA, the final two feedback settings (100 and 101) were not used at the moment of measuring. Instead, the 011 feedback setting was used for distances of roughly 40 cm and upwards. We can clearly see that the DAC works however as the cross-correlation output increases when the DAC switches to the next feedback setting.

6-2-3 Feedback DAC

In order to see the effect of the programmable feedback gain, a measurement was carried out in which the target was placed at a fixed distance and the feedback setting was altered. The results are shown in Figure 6-8.

The cross-correlation output is clearly dependent on the DAC feedback setting. If a feedback current is chosen that is too high, the system is unable to detect the signal. As the feedback

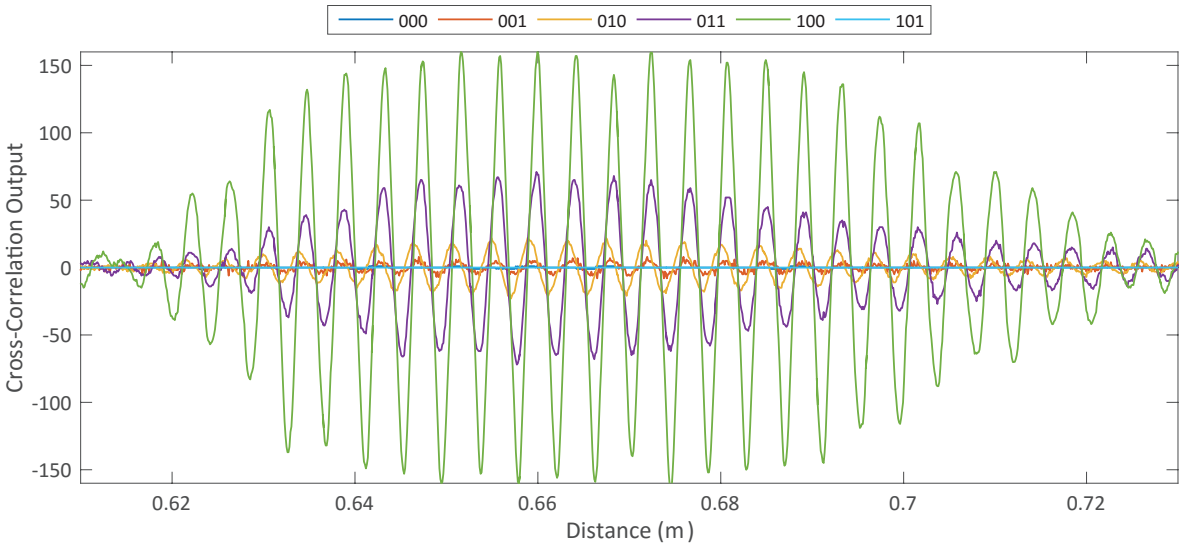

Figure 6-8: The cross-correlation output for the feedback DAC simulations.

current decreases, the output of the cross-correlation filter improves. If the feedback current is however chosen too high, the resulting waveforms will show slewing (100). No cross-correlation output is available for the 101 feedback setting because the pre-charge circuit was disabled during this experiment. This has lead the pre-charge time of the input node to increase beyond the ToF associated with a distance of 60 cm. We expect the cross-correlation output to look like a triangle in that setting if the pre-charge circuit is enabled, because there will be even more slewing than for the 100 feedback setting.

6-2-4 Summary

The measured current consumption of the chip was 1.28 mA.

Although the cross-correlation operation is able to determine the zero-cross points with a resolution of far below one wavelength, we are as of yet unable to determine the actual ToF within an accuracy of one wavelength due to the heavy ringing associated with high-Q transducers. However, we believe that our system will be able to accurately determine the ToF if a low-Q transducer is used. Future measurements should therefore be carried out with a low-Q transducer.

The large displacement measurements show that the transducer is capable of measuring bigger displacements. The actual resolution of the system remains to be determined. In order to do that, a low-Q transducer should be used and the FPGA code should be adjusted to include the 100 and 101 feedback settings in order to detect echoes from longer distances as well.

The scalable feedback DAC is working as intended.

Chapter 7

Conclusion

In this project, a cross-correlation based ultrasonic ranging system has been discussed. The designed chip targets the readout of piezoelectric ultrasound transducers for distance estimation. The system architecture was designed to be compact by arranging it in a voltage readout configuration where the designed $\Sigma\Delta$-ADC uses the transducer to passively integrate the difference between the analog input signal and the comparator's output. A compact pre-amplifier architecture was implemented that offers a low input common-mode level to reduce the time required to pre-charge the transducer.

The implemented single-bit cross-correlation filter is able to determine the zero-crossings of the received echoes with a σ of 0.169 mm. The worst-case error in zero-crossings that was measured is 0.6 mm. This error is thought to be caused by an inaccurate measurement approach where the target was moved by hand in steps of roughly 1 mm.

Accurate ToF estimation was not possible for high-Q transducers because of the large amount of ringing. Due to time constraints we were not able to measure the performance of the system for low-Q transducers. Simulations have however shown us, that the system will be able to accurately determine the ToF for low-Q transducers.

In future work, further measurements should be done to determine the performance of the system for low-Q transducers. Furthermore, a more accurate target displacement scheme needs to be used to accurately control the distance between transducer and target. Also, measurements could be done with a chirp signal. Intuitively, this should produce accurate results because the zero-crossings can be determined with high accuracy.

Finally, there are some things that can be improved in the design in order to further reduce the size of the complete ranging system. First of all, the external high-voltage switches should be implemented internally to reduce area requirements. The cross-correlation filter can be implemented in hardware using one of the described methods in subsection 4-2-4. Furthermore, an R-2R-type architecture should be implemented to reduce the total size of the feedback DAC.

Appendix A

MATLAB Code

Appendices are found in the back.

A-1 Functional part of GUI Code

```
function run_Callback(hObject, eventdata, handles, varargin)
% hObject    handle to run (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%Signal definitions
fin             = handles.d_in.fin;                     %Signal input frequency
amp             = handles.d_in.amp;                     %Signal amplitude
phase_delay     = handles.d_in.phase_delay;             %Signal phase delay
offset          = handles.d_in.offset;                  %Signal offset
noise           = handles.d_in.noise;                   %Signal noise level

%Resolution definitions
OSR             = handles.d_in.OSR;                     %Over-sampling ratio
fs              = OSR*fin*2;                            %Sampling frequency
nr_periods      = handles.d_in.nr_periods;              %Number of sinal periods
ppp             = fs/fin;                               %Points per period.

%Environment variables
meas_time       = 4e-3;                                 %Measurement time
nr_samples      = fs*meas_time;                         %Total amount of samples
c               = 343.6;                                %Speed of sound
d               = handles.d_in.d;                       %Distance to receiver
temp            = handles.d_in.temp;                    %Temperature
humidity        = handles.d_in.humidity;                %Humidity
tof             = d/c;                                  %Simple time-of-flight
    calculation
delta_t         = 1/fs;                                 %Time step
t               = delta_t:delta_t:meas_time;            %Time-domain

%Quantizer definitions
vref            = 1;                                    %Quantizer reference voltage
q_out           = zeros(1,nr_samples);                  %Quantizer output
%Reference definitions;
nr_ref_bits     = handles.d_in.nr_ref_bits;             %Number of reference bits
nr_ref_levels   = 2^nr_ref_bits;                        %Number of reference levels
```

```
35 delta_ref       = 2/(nr_ref_levels-1);                  %2 because it's between -1 and
       1
36 nr_ref_periods  = handles.d_in.ref_periods;             %The number of reference
       periods; By default equal to the number of transmitted periods
37 N               = ppp*nr_ref_periods;                   %Number of reference points
38 ref_pd          = handles.d_in.ref_pd;                  %Reference phase delay w.r.t.
       the transmitted signal
39 set(handles.N,'String',N);                              %Display the number of
       reference points
40 %Additional settings
41
42 ref_type        = handles.d_in.ref_type;                %Single-bit or multi-bit
       reference
43
44
45 % Calculate X-correlation
46 [REF,echo]            = get_echo(t,ppp,nr_periods,fin,delta_t,amp,phase_delay,fs,tof,
       meas_time);
47 assignin('base','signal',REF);
48 %REF_pts          = round(tof*fs)+(nr_periods-nr_ref_periods)*ppp+round(ref_pd*ppp/(2*
       pi)):round(tof*fs)+nr_periods*ppp+round(ref_pd*ppp/(2*pi));
49 REF               = quantizer(REF,length(REF), delta_ref, vref);    %Create a multi-bit
       reference
50 %REF = quantizer(kaiser(length(REF),20),nr_ref_periods*ppp,delta_ref,vref);
51 %REF = ones(1,400);
52 %REF(201:400) = -1;
53 %REF_top = REF(round((ref_pd/(2*pi))*ppp)+1:end);
54 %REF_bottom = REF(1:round((ref_pd/(2*pi)*ppp)));
55 %REF = [REF_top, REF_bottom];
56 length(REF)
57 switch (ref_type)
58     case 'MULTI'
59     case 'SQUARE'
60         REF             = sign(REF);
61     case 'TRIANGLE'
62         slope           = 4*fin;
63         t_half          = 1/(2*fin);
64         REF             = slope*mod((0:ppp*nr_ref_periods-1)*delta_t,t_half)-1;
65         for a1 = 1:nr_ref_periods
66             REF((a1*2-1)*ppp/2+1:a1*ppp) = REF((a1*2-1)*ppp/2+1:a1*ppp)*-1;
67         end
68         shift_amount    = round(phase_delay*ppp/(2*pi))+round(ref_pd*ppp/(2*pi));
                                    %Add phase lead to the reference
69         shift_amount    = round(shift_amount + 3*ppp/4);
                                                          %Standard phase shift
               to move the peak to the middle of the square wave instead of the end
70         REF             = [REF(length(REF)-shift_amount+1:length(REF)) REF(1:length(
               REF)-shift_amount)];    %Phase Delay
71         REF             = quantizer(REF,ppp*nr_ref_periods, delta_ref, vref);
                                      %Quantized values
72 end
73
74 absorption_coef = get_absorption(fin,temp,humidity);
75 air_absorption  = amp/(absorption_coef^d);
76
77 echo            = air_absorption*echo/(4*pi*d^2) + noise*randn(1,length(echo))+offset;
78 q_out           = get_q_out(nr_samples,vref,echo);
79 XCORR           = -1*conv(q_out,fliplr(REF));
80 assignin('base','REF',REF);
81 assignin('base','q_out',q_out);
82 assignin('base','XCORR',XCORR);
83 assignin('base','echo',echo);
84 %XCORR           = XCORR/max(abs(XCORR));   %normalize
85
86
87 %% Generate plots
```

```
88  axes(handles.TX);
89  NFFT             = 2^nextpow2(nr_samples);
90  ffty             = fft(XCORR,NFFT)/nr_samples;
91  f                = fs/2*linspace(0,1,NFFT/2+1);
92  ffty_magn        = abs(ffty)/max(abs(ffty));
93  ffty_dB          = 20*log10(ffty_magn);
94
95  assignin('base','ffty_dB',ffty_dB(1:NFFT/2+1));
96  assignin('base','f',f);
97
98  semilogx(f,ffty_dB(1:NFFT/2+1));
99  xlim([1e3 fs/2]);
100 ylim([-100 10]);
101 grid on
102
103 %Plot the echo
104 axes(handles.echo);
105 plot(t,echo);
106 hold on;
107 plot(t(round(fs*tof)+1:round(fs*tof)+length(REF)),REF,'g','Linewidth',2);
108 line([t(round(fs*tof)+1) t(round(fs*tof)+1)], [-2 2],'Color','r');
109 line([t(round(fs*tof)+ppp*nr_ref_periods) t(round(fs*tof)+ppp*nr_ref_periods)], [-2
        2],'Color','r');
110 hold off;
111 %set(handles.echo,'xlim',[tof-5*ppp*delta_t tof+(5+nr_ref_periods)*ppp*delta_t]);
112 set(handles.echo,'XMinorTick','on');
113 grid on
114
115 %Plot the ADC output
116 axes(handles.adc_output);
117 bar(t,q_out);
118 hold on;
119 line([t(round(fs*tof)+1) t(round(fs*tof)+1)], [-2 2],'Color','r');
120 line([t(round(fs*tof)+ppp*nr_ref_periods) t(round(fs*tof)+ppp*nr_ref_periods)], [-2
        2],'Color','r');
121 hold off;
122 %set(handles.adc_output,'xlim',[tof-5*ppp*delta_t tof+(5+nr_ref_periods)*ppp*delta_t])
        ;
123 set(handles.adc_output,'ylim',[-1.5 1.5]);
124 set(handles.adc_output,'XMinorTick','on');
125 grid on
126
127 %Plot Correlation function output
128 axes(handles.xcorrelation);
129 bar(t,XCORR(1:length(t)));
130 hold on;
131 line([t(round(fs*tof)+ppp*nr_ref_periods   +round(ref_pd*ppp/(2*pi))      ) t(round(fs
        *tof)+ppp*nr_ref_periods  +round(ref_pd*ppp/(2*pi)))], [-400 400],'Color','r');
132 hold off;
133 %set(handles.xcorrelation,'xlim',[tof-5*ppp*delta_t tof+(5+nr_ref_periods)*ppp*delta_t
        ]);
134 set(handles.xcorrelation,'xlim',[0 4e-3]);
135 set(handles.xcorrelation,'ylim',[-400 400]);
136 set(handles.xcorrelation,'XMinorTick','on');
137 grid on
138
139 %% Calculation Function Definitions
140
141 %************************************************************************
142 % Function to simulate the receiver and signal processing
143 %************************************************************************
144 function q_out = get_q_out(nr_samples,vref,echo)
145
146     %ADC definitions
147     adc_type        = handles.d_in.adc_type;            %ADC type (SD or COMP)
148     order           = handles.d_in.order;               %Sigma Delta order
```

```
149     nr_adc_bits     = handles.d_in.nr_adc_bits;         %Number of bits
150     nr_levels       = 2^nr_adc_bits;                    %Number of quantizer levels
151     delta           = 2/(nr_levels-1);                  %2 because it's between -1 and
            1
152
153     switch lower(adc_type)
154         case{'comparator'}
155                     q_out = quantizer(echo,nr_samples, delta, vref);
156         case{'sigma delta'}
157             switch (order)
158               case(1)
159                     q_out = sigmadelta1(echo,nr_samples,vref,delta);
160                 case(2)
161                     q_out = sigmadelta2(echo,nr_samples,vref,nr_levels,delta);
162                 otherwise
163                     error('Only first and second order Sigma-Deltas have been
                            implemented');
164             end
165     end
166 end
167
168 %************************************************************************
169 % Quantization function
170 %************************************************************************
171 function [q_out, on_bits] = quantizer(in, nr_samples, delta, vref)
172     q_out = zeros(1,nr_samples);
173     for a = 1:nr_samples
174         q_out(a) = quant(in(a)/vref + 1, delta) -1;          %Normalize input range,
                then shift back
175         q_out(a) = sign(in(a))*min(abs(q_out(a)),1);        %Clip quantized input
                value so it stays smaller than 1 and larger than -1
176         on_bits = round((q_out+1)/delta);                   %Return the amount of
                bits that are on
177
178     end
179 end
180
181 %************************************************************************
182 % Full adder that produces the ouput sequence with the first value on the first
        position
183 %************************************************************************
184 function ff_out = flipflop(FF, pointer, N)
185     ff_out = FF(mod(pointer+(1:N)-1,N)+1);
186 end
187
188 %************************************************************************
189 % Function that generates the echo from the sent signal
190 %************************************************************************
191 function [REF,echo] = get_echo(t,ppp, nr_periods,fin,delta_t,amp, phase_delay,fs,tof,
        meas_time)
192     Q = handles.d_in.damping_coef;              %Quality factor
193
194     N1 = round(tof*fs);
195     N2 = round(tof*fs+nr_periods*ppp);
196     N3 = fs*meas_time;
197     echo1 = 1:N1;                               %No signal has yet arrived
198     echo2 = N1+1:N3;                            %Signal has arrived
199
200     chirp_dev = handles.d_in.chirp_dev;         %Chirp range
201     sig_type = handles.d_in.sig_type;           %Chirp signal Y/N?
202
203     f_low = fin - chirp_dev;
204     f_high = fin + chirp_dev;
205
206     %% Filter definition for the specified Q
207     %         ----
```

```
208     %        / Ls |
209     %       / ---
210     %    \/   Cs
211     % Q = --------
212     %          Rs
213     %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
214
215     Ls = 159e-3;
216     Cs = 99e-12;
217     Cp = 3e-9;
218     Rs = sqrt(Ls/Cs)/Q;
219
220     %% Transducer Transfer Function
221     norm = (Cs + Cp)/Rs;              %normalize
222     num = Rs*(norm);
223     den = [Ls*Cs*Cp Rs*Cs*Cp (Cs + Cp)];
224     H = tf(num,den);
225     [~,fpeak] = getPeakGain(H);
226 %    fin = fpeak/(2*pi);                %% Adjust to resonance frequency
227
228
229     TX = t;
230     switch sig_type
231         case{'const'}
232             %% The transmitted Pulse
233             pulse = sin(2*pi*fin*t(ppp+1:ppp+ppp*nr_periods))      % The pulse itself
234             REF = pulse;
235             assignin('base','pulse',pulse);
236             signal = zeros(1,length(t));                            % Complete signal
                    definition
237             signal(N1+1:N1+ppp*nr_periods) = pulse;                 % Pulse is put in
                    the signal here.
238
239             %% Transmitted Wave
240             echo(echo1) = 0;                                        % Pulse not yet
                    arrived
241             TX(echo2) = lsim(H,signal(echo2),t(echo2));             % Pulse sent
242             echo(echo2) = lsim(H,TX(echo2),t(echo2));               % Pulse arrived
243 %             echo(echo2) = lsim(H,signal(echo2),t(echo2));       % Pulse sent
244         case{'chirp_lh'}
245             %% The transmitted Pulse
246             t_chirp = delta_t:delta_t:nr_periods/f_high;            % Time-vector used to
                    generate the chirp signal
247             f_chirp = f_low:(f_high-f_low)/length(t_chirp):f_high-(f_high-f_low)/
                    length(t_chirp); %frequency vector
248             pulse = sin(2*pi*f_chirp.*t_chirp-phase_delay);         % The pulse
249             REF = pulse;
250             length(pulse)
251             signal = zeros(1,length(t));
252             signal(N1+1:N1+length(pulse)) = pulse;              % The pulse inserted
253
254             %% Transmitted Wave
255             echo(echo1) = 0;
256             TX(echo2) = lsim(H,signal(echo2),t(echo2));             % Pulse sent
257             echo(echo2) = lsim(H,TX(echo2),t(echo2));               % Pulse arrived
258         case{'chirp_hl'}
259             %% The transmitted Pulse
260             t_chirp = delta_t:delta_t:nr_periods/f_high;            % Time-vector used to
                    generate the chirp signal
261             f_chirp = f_low:(f_high-f_low)/length(t_chirp):f_high-(f_high-f_low)/
                    length(t_chirp); %frequency vector
262             pulse = sin(2*pi*f_chirp.*t_chirp-phase_delay);         % The pulse
263             pulse = fliplr(pulse);
264             REF = pulse;
265             length(pulse)
266             signal = zeros(1,length(t));
```

```
267             signal(N1+1:N1+length(pulse)) = pulse;          % The pulse inserted
268         end
269             %% Transmitted Wave
270             echo(echo1) = 0;
271             TX(echo2) = lsim(H,signal(echo2),t(echo2));         % Pulse sent
272             echo(echo2) = lsim(H,TX(echo2),t(echo2));           % Pulse arrived
273 end
274
275
276 %************************************************************************
277 % Function that simulates a first order Sigma Delta
278 %************************************************************************
279 function y = sigmadelta1 (in, nr_samples,vref,delta)
280
281     % Pre-allocates and initializes array variables
282     w = zeros(1,nr_samples);
283     y = zeros(1,nr_samples);
284     w(1) = 0.5;
285     y(1) = 1;
286     alpha = max(abs(in)); %10% leakage
287    % alpha = 1;
288
289     % The main loop simulating the accumulator and the quantizer
290     %************************************************************************
291     for a = 2:nr_samples
292         w(a) = w(a-1) + in(a-1) - alpha*y(a-1);
293         y(a) = quantizer(w(a), 1, delta, vref);
294     end
295     %************************************************************************
296 end
297
298 %************************************************************************
299 % Function that simulates a second order Sigma Delta
300 %************************************************************************
301 function y = sigmadelta2 (in,nr_samples,vref,nr_levels,delta)
302
303     elements = ones(1,nr_levels-1) ;%+ 0.0001*randn(1,7); %DAC elements
304     % Coefficients and Scaling
305     a2 = 2;%2/1.7/2;
306     b1 = 1;%1/1.7/2;
307     b2 = 1;%1/0.29014/2;
308
309     % Pre-allocate and initialize array variables => faster code
310     y  = zeros(1,nr_samples); y(1) = 1;
311     DAC = zeros(1,nr_samples);
312     w1 = y;  w1(1) = 0;
313     w2 = y;  w2(1) = 0.1;
314     for a = 2:nr_samples
315          w1(a) = w1(a-1) + b1*(in((a-1)) - DAC(a-1));
316          w2(a) = w2(a-1) + b2*(w1(a-1) - a2*DAC(a-1));
317          [y(a), index] = quantizer(w2(a), 1, delta, vref);
318          DAC(a) = thermoDAC(index, elements, nr_levels); % standard DAC
319     end
320 end
321
322 %************************************************************************
323 % Function that simulates a multi-bit DAC
324 %************************************************************************
325 function DAC = thermoDAC(index, elements, nr_levels)
326     % Realizes a thermometer DAC implementing "nr_levels" levels
327     % "elements" is a vector of NORMALIZED unit DAC elements
328     % "index" = [0 .. nr_levels-1] is the thermometer code provided by the quantizer
329
330     nr_elements = nr_levels - 1;       % number of DAC elements
331     DACvect = -ones(1,nr_elements);    % Reset all elements to -1
332     DACvect(1:index) = 1;              % Set "index" DAC elements to +1
```

```
333     set(handles.fin,'String',length(elements));
334     set(handles.amp,'String',length(DACvect));
335       set(handles.phase_delay,'String',nr_elements);
336     DAC = (elements*DACvect')/nr_elements; % Build the DAC output from the given unit
            elements
337 end
338
339 function absorption_coef = get_absorption(F,T,hr)
340     % % This function was written by Edward L. Zechmann
341     T=273.15+T; % To convert from Celsius
342     T01=273.16; %triple point in degrees Kelvin
343     T0=293.15;
344     % atmospheric pressure ratio is the ambient pressure/standard pressure
345     ps0=1;  % ps0= standard pressure/standard pressure which is unity
346
347     % Bass formula for saturation pressure ratio
348     psat_ps0=10^( 10.79586*(1-T01/T) -5.02808*log10(T/T01) +1.50474*10^(-4)
            *(1-10^(-8.29692*(T/T01-1))) -4.2873*10^(-4)*(1-10^(-4.76955*(T01/T-1)))
            -2.2195983);
349
350     ps_ps0=1/ps0;
351
352     h = hr*psat_ps0/ps_ps0; % h is the humidity in percent molar concentration
353     % Bass formula
354
355     Fr0=1/ps0*(24+4.04*10^4*h*(0.02+h)/(0.391+h));
356     FrN=1/ps0*(T0/T)^(1/2)*(9+280*h*exp(-4.17*((T0/T)^(1/3)-1)));
357
358     % Calculate the air absorption in dB/meter using the Bass formula
359     absorption_coef = 20*log10(exp(1))*F^2*( 1.84*10^(-11)*(T/T0)^(0.5)*ps0+((T/T0)
            ^(-5/2))*( 0.01275*exp(-2239.1/T)/(Fr0+F^2/Fr0)+0.1068*exp(-3352/T)/(FrN+F^2/
            FrN) ));
360     absorption_coef = 10^(absorption_coef/20);
361 end
362
363 end
```

Bibliography

[1] "Basic principle of medical ultrasonic probes (transducer)." http://www.ndk.com/en/sensor/ultrasonic/basic02.html.

[2] H. Eriksson, P. Borjesson, P. Odling, and N.-G. Holmer, "A robust correlation receiver for distance estimation," *Ultrasonics, Ferroelectrics, and Frequency Control, IEEE Transactions on*, vol. 41, pp. 596–603, Sept 1994.

[3] D. Marioli, C. Narduzzi, C. Offelli, D. Petri, E. Sardini, and A. Taroni, "Digital time of flight measurement for ultrasonic sensors," in *Instrumentation and Measurement Technology Conference, 1991. IMTC-91. Conference Record., 8th IEEE*, pp. 198–201, May 1991.

[4] R. Queiros, F. Alegria, P. Girao, and A. Serra, "Cross-Correlation and Sine-Fitting Techniques for High-Resolution Ultrasonic Ranging," *Instrumentation and Measurement, IEEE Transactions on*, vol. 59, pp. 3227–3236, Dec 2010.

[5] S. Hirata, M. Kurosawa, and T. Katagiri, "Real-time ultrasonic distance measurements for autonomous mobile robots using cross correlation by single-bit signal processing," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pp. 3601–3606, May 2009.

[6] R. Nave, "Sound Speed in an Ideal Gas." http://hyperphysics.phy-astr.gsu.edu/hbase/sound/souspe3.htmlc2.

[7] "IEEE Standard on Piezoelectricity," *ANSI/IEEE Std 176-1987*, 1988.

[8] "IEEE Standard Definitions and Methods of Measurement for Piezoelectric Vibrators," *IEEE Std No.177*, 1966.

[9] T. Singal, *Wireless Communications*. Tata McGraw Hill, 2010.

[10] H. E. Bass, L. C. Sutherland, J. Piercy, and L. Evans, "Absorption of sound by the atmosphere," in *Physical acoustics: Principles and methods. Volume 17 (A85-28596 12-71). Orlando, FL, Academic Press, Inc., 1984, p. 145-232.*, vol. 17, pp. 145–232, 1984.

[11] Texas Instruments, "Ultrasonic Distance Sensing: Pulse Skipping." unpublished.

[12] C.-C. Tong, J. Figueroa, and E. Barbieri, "A method for short or long range time-of-flight measurements using phase-detection with an analog circuit," *Instrumentation and Measurement, IEEE Transactions on*, vol. 50, pp. 1324–1328, Oct 2001.

[13] F. Gueuning, M. Varlan, C. Eugne, and P. Dupuis, "Accurate distance measurement by an autonomous ultrasonic system combining time-of-flight and phase-shift methods," *Instrumentation and Measurement, IEEE Transactions on*, vol. 46, pp. 1236–1240, Dec 1997.

[14] C. Cai and P. P. Regtien, "Accurate digital time-of-flight measurement using self-interference," *Instrumentation and Measurement, IEEE Transactions on*, vol. 42, pp. 990–994, Dec 1993.

[15] X. Wang and Z. Tang, "Optional Optimization Algorithms for Time-of-Flight System," *Instrumentation and Measurement, IEEE Transactions on*, vol. 60, pp. 3326–3333, Oct 2011.

[16] E. Cabral and I. Valdez, "Airborne ultrasonic sensor node for distance measurement," in *SENSORS, 2013 IEEE*, pp. 1–4, Nov 2013.

[17] E. W. Weisstein, "Cross-Correlation." From MathWorld–A Wolfram Web Resource.

[18] C. Chen, "CSA vs. TIA: A Comparison of Noise-Efficient Ultrasound LNA Candidates." unpublished, 2015.

[19] O. Feely and L. Chua, "The effect of integrator leak in Σ-Δ modulation," *Circuits and Systems, IEEE Transactions on*, vol. 38, pp. 1293–1305, Nov 1991.

[20] S. Kashmiri, S. Xia, and K. Makinwa, "A Temperature-to-Digital Converter Based on an Optimized Electrothermal Filter," *Solid-State Circuits, IEEE Journal of*, vol. 44, pp. 2026–2035, July 2009.

[21] R. Schreier and G. C. Temes, *Understanding Delta-Sigma Data Converters*. New Jersey, NJ: Wiley, 2005.

[22] R. Schreier, "Understanding Continuous-Time, Discrete-Time Sigma-Delta ADCs and Nyquist ADCs," 2009.

[23] L. Breems and J. H. Huijsing, *Continuous-Time Sigma-Delta Modulation for A/D Conversion in Radio Receivers*. Kluwer Academic Publishers, 2001.

[24] E. Hogenauer, "An economical class of digital filters for decimation and interpolation," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 29, pp. 155–162, Apr 1981.

[25] L. Breems, "Lecture slides in Over-Sampled Data Converters." 2014.

[26] B. Razavi, *Design of Analog CMOS Integrated Circuits*. McGraw-Hill, first ed., 2001.

[27] B.-M. Min, P. Kim, I. Bowman, F.W., D. Boisvert, and A. Aude, "A 69-mW 10-bit 80-MSample/s Pipelined CMOS ADC," *Solid-State Circuits, IEEE Journal of*, vol. 38, pp. 2031–2039, Dec 2003.

[28] MULTICOMP, "MCUSD16A40S12RO."