Motional Feedback in a Bass Loudspeaker

Digital Implementation

by

Sybold Hijlkema & Bishwas Regmi

to obtain the degree of Bachelor of Science
at Delft University of Technology,
to be defended on Monday July 2, 2018

Student number: 4449487 & 4467655
Project duration: April 23, 2018 – July 6, 2018
Thesis committee:

Prof. Koen Bertels	TU Delft, chairman
Dr. ir. G. J. M. Janssen,	TU Delft, supervisor, jury member
Dr. S. Izadkhast,	TU Delft, jury member

TU Delft Delft University of Technology

Abstract

This thesis describes the digital implementation of a motional feedback system for a bass loudspeaker. Motional feedback is used to suppress the linear and non-linear distortions produced by the loudspeaker, especially at the low frequencies. An accelerometer is mounted on the cone of the loudspeaker to provide the feedback signal. The controller which consists of a PI controller and an equalizer are implemented on an FPGA. The equalizer, which is the inverse of the linear model of the loudspeaker, is used to compensate for the linear distortion. The PI controller with negative feedback is used to suppress the non-linear distortion. Not all measurement results are available at the moment of submission of this thesis. However, simulations were carried out on the model of the loudspeakers which show that the linear distortion is fully suppressed. The reduction of the non-linear distortion due to the controller can not be seen in the simulations.

Preface

This thesis is written as part of the Bachelor Graduation Project of B.Sc. Electrical Engineering at Delft University of Technology. The primary focus of this thesis is the development of a digital motional feedback system for a bass loudspeaker. The assignment was provided to us by Dr. Ir. G.J.M Janssen, who had built an analog motional feedback system himself.

We would like to thank Dr. Janssen, not only for providing the assignment, but also for reviewing our weekly progress and guiding us to the right direction during the project. We would also like to thank Dr.Ir. T.G.R.M. van Leuken for providing us with all the necessary digital hardware. Furthermore, we would like to thank the jury members for taking their time to assess our work.

Sybold Hijlkema
Bishwas Regmi
Delft, July 2018

Contents

Abbreviations

ADC	Analog to Digital converter
$ASIC$	Application specific integrated circuit
$BIBO$	Bounded Input Bounded Output
CS	Chip select
DAC	Digital to Analog converter
$DF-I$	Direct Form-I
$DF-II$	Direct Form-II
DSP	Digital Signal Processor
FFT	Fast Fourier Transform
FIR	Finite impulse response
$FPGA$	Field-programmable gate array
IIR	Infinite impulse response
MCU	Micro-controller unit
MFB	Motional Feedback
$MISO$	Master Input Slave Output
$MOSI$	Master Output Slave Input
OS	Operating System
SCK	Serial clock
SNR	Signal-to-noise ratio
SPI	Serial Peripheral Interface
THD	Total harmonic distortion
$VHDL$	VHSIC Hardware Description Language

1 Introduction

A loudspeaker, like any real electro-mechanical transducer, is a non-ideal device with physical properties and limitations. For low amplitude input signals, where its behaviour is approximately linear, the speaker manifests distortion of the input signal in the form of a non-flat transfer function. For high amplitude input signals and especially when reproducing lower frequencies, where the excursions of the cone are higher to generate the same audio power as during the reproduction of higher frequencies, non-linearities in the electrical and mechanical properties of the speaker cause additional deformation of the sound in the form of harmonic and inter-modulation distortion.

One way to reduce the detrimental effects of both the linear and non-linear behaviour of the speaker is by using feedback to correct for the distortions. Some sources of feedback signals that have been used are the back EMF of the speaker voice coil [23] and that of a secondary voice coil mounted on the diaphragm [6], but these methods are not sufficient to produce the best possible results.

In 1968, a motional feedback system was proposed by Philips [19] to suppress linear and non-linear distortion in bass speakers. The system used a piezoelectric accelerometer mounted on the speaker cone to measure its acceleration. The recorded signal was fed back to a control system which compensated for the distortion and improved the performance of the speaker.

The concept of Motional Feedback will be studied and applied in our Bachelor's Graduation Project. The team working on this project consists of three sub-groups each working on a different implementation of this concept, namely: the analogue, digital and theory group. It is meant that the digital group will try to come up with a digital implementation of this motional feedback controller. The analogue group is expected to design an analogue implementation of this system. The theory group will parameterize the speaker in order to create a model of the non-linear loudspeaker and work towards designing an optimal controller. This model as well as the measurement setup and code used to measure the performance of the speaker will be used to validate the controller designed by the other two subgroups. This thesis will cover the work of the digital group.

2

Problem Definition

The following section describes the linear model for the loudspeaker as well as the cause and effects of non-linearities that arise in the loudspeaker response. These are needed for both the design of the digital and analogue implementation, as well as being a foundation for the derivation of the nonlinear model and the design of the ideal controller.

2.1. Loudspeaker Model

A schematic of a generic loudspeaker can be seen in Fig. 2.1. In short, it simply works by sending a current through the voice coil, which is then attracted or repelled by the permanent magnet. This causes the cone to move and thus creating sound.

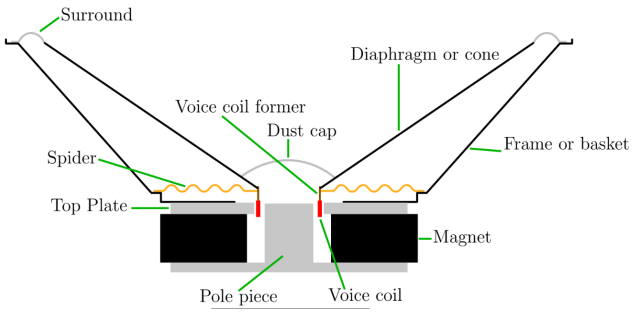


Figure 2.1: A cross-section of a typical loudspeaker. Courtesy of [12]

The electric, magnetic, mechanical and acoustic behaviour of the loudspeaker can be modelled using an electric circuit [29], which has the benefit of being able to calculate how the system reacts to an input signal. An example of such a circuit can be seen in Figure 2.2, in which an acoustic load has been neglected. The acoustic load can be neglected due to its very small magnitude [22]. This circuit only holds for low frequencies. However, since only a bass speaker is concerned, this constraint is no issue. The part of the circuit before the gyrator corresponds to the actual electrical part of the loudspeaker and that which comes after the gyrator corresponds to the mechanical components of the loudspeaker, hence the gyrator symbolises the transformation from the electrical domain to the mechanical domain with a factor $B \cdot l$. L_E is the selfinductance of the voice coil, R_E is its resistance and i_s is the current flowing through the voice coil. The current i_c is the electrical representation of the velocity of the cone, R_b represents the damping of the speaker cone, C_m represents the compliance of the speaker cone and M_m represents the mass of the moving speaker cone. The voltage across all the elements to the right of the gyrator represents the force acting upon the speaker cone.
From this simplified circuit a linear estimation of the state-space model (see Equation 2.1) and a transfer function of the speaker can easily be derived, which can give a basic illustration of how the speaker behaves.

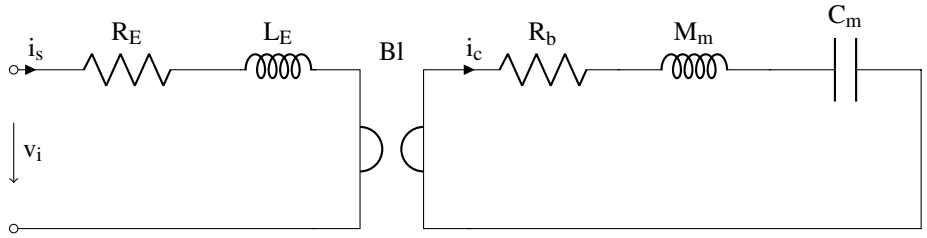


Figure 2.2: A schematic of the equivalent circuit for a loudspeaker. The left side of the gyrator represents the electrical domain, and the right side represents the mechanical domain.

But this was not used during the design process since determining the individual parameters of the speaker would take far too long. Also the derived transfer function will not provide a usable response, as the model only represents the impedance of the loudspeaker. However, the acoustic response of the loudspeaker is more important than the impedance when designing the controller. Thus instead the measured response will be used.

$$\dot{\mathbf{x}} = \begin{bmatrix} \frac{-R_b}{R_m} & \frac{-1}{M_m \cdot C_m} & \frac{B \cdot l}{M_m} \\ 1 & 0 & 0 \\ \frac{-B \cdot l}{L_e} & 0 & \frac{-R_e}{L_e} \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L_e(x)} \cdot v_i \end{bmatrix} \tag{2.1}$$

Where the vector $\mathbf{x}$ contains the state variables of the system, which are: $\dot{i}_c$, i_c and i_s. Which means the system is fully described by: i_s, the current through the voice coil, i_c, the velocity of the cone and $\dot{(i)}_c$, the acceleration of the cone.

2.2. Nonlinearities of a Loudspeaker

Loudspeakers in general tend to express non-linear behaviour, especially at lower frequencies. These non-linear behaviours are caused by the physical limitations of such a transducer as well as the geometry and material properties of the loudspeaker components, several nonlinearities are present in the system. Most of these nonlinearities are prominent at higher amplitudes, and they can have detrimental effects on the quality of the sound produced. Klippel [3], Bai and Huang [4] give an overview of the main causes of nonlinearities in loudspeaker systems. The suspension, the force factor and the voice coil inductance are the main sources of these non-linearities and all depend on the displacement of the cone. The non-linear behaviour of the parameters affects the loudspeaker transfer at different frequencies inducing harmonic distortion. The subsequent sections will further elaborate the separate sources of nonlinearity and their effect on the sound quality.

2.2.1. Suspension of the loudspeaker

The stiffness of the suspension of a loudspeaker K_m is related to the mechanical properties of the two suspension components of the speaker cone: the spider and surround (Fig. 2.1). For small displacements, K_m is constant and the suspension can be modelled as a linear spring. At higher displacements, the restoration force becomes larger as a function of x_d, the displacement of the cone, and a nonlinearity is introduced. The restoration force is given by:

$$F = K_m(x_d)x_d \tag{2.2}$$

The frequency dependency of the stiffness is linear. A related parameter to K_m is the compliance, $C_m = \frac{1}{K_m}$, which is the inverse of the stiffness.

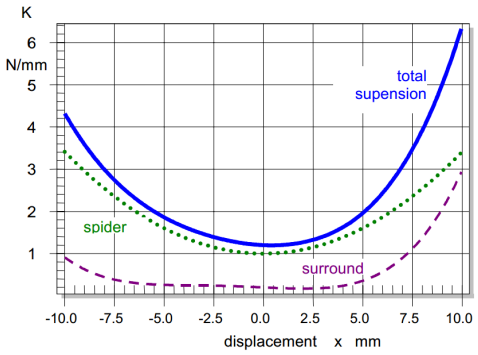


Figure 2.3: Force-Deflection curve showing nonlinearity of the spider and surround stiffness for large displacements.[20]

2.2.2. Force Factor $Bl(x_d)$

The force factor $Bl(x_d)$ is the integral of the flux density B over the effective wire length l of the voice coil in the air gap. It describes coupling between the magnet and voice coil of the loudspeaker. For a small displacement Bl is constant but for large displacement the voice coil leaves the gap and $Bl(x_d)$ decreases as function of displacement. This variation in the force factor introduces two nonlinearities:

- The back EMF u_{EMF} generated by the movement of the coil becomes dependent on displacement:

$$u_{EMF} = Bl(x_d)v \tag{2.3}$$

  where v is the velocity of the speaker cone. This has the effect of variation in the electrical damping.

- The Lorentz force also becomes displacement dependent:

$$F = Bl(x_d)i \tag{2.4}$$

  where i is the current in the voice coil. The force factor does not vary with frequency.

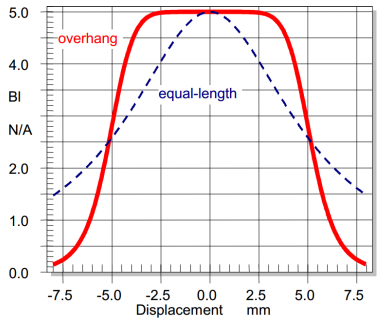


Figure 2.4: Plot showing non-linearity of the force factor for large displacements. The form of the Bl characteristic depends on the width of the voice coil, in the direction of movement, compared to the width of the gap. A coil which is wider than the gap will allow it to behave linearly for larger displacement.[20]

2.2.3. Voice Coil Inductance $L_E(x_d)$ and $L_E(i)$

The voice coil inductance L_E is also dependent on x_d. Because of the geometry of the loudspeaker motor, for positive displacement the magnetic field produced by the coil penetrates mainly the surrounding air, increasing magnetic reluctance thus decreasing the voice coil inductance. For negative displacement the magnetic field penetrates the steel surrounding the magnet (as well as the magnet) which has much higher permeability. This causes the reluctance to increase and $L_E(x_d)$ to increase.

2.2.4. Harmonic and Intermodulation Distortion

In Fig. 2.5, an arbitrary nonlinear function is shown, with the linearisation in the origin, which, in fact, according to [20], should resemble the restoration force that acts upon the loudspeaker cone. The slope of the graph is proportional to the suspension stiffness K_m, but its shape clearly indicates that it is dependent on the position x_d.

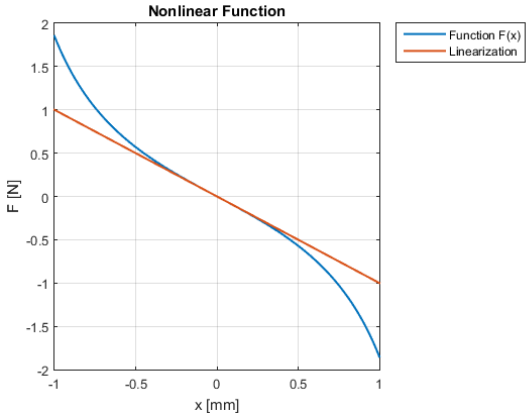


Figure 2.5: Nonlinear function with linearised graph through the origin. The nonlinear function is a primitive model for the restoration force, which is related to the suspension stiffness $K_m(x_d)$.

The nonlinear function of Fig. 2.5 can be expanded using Taylor expansion, which will give further insight in the distortion this will cause. The resulting expression is given in Equation 2.5. A third order polynomial is used by [1] for the suspension stiffness. It is stated by [16] that a Gaussian sum may be the preferred choice over polynomial expansion, because it is more accurate outside the initial range.

$$F(x_d) = a_0 + a_1 x_d + a_2 x_d^2 + a_3 x_d^3 + a_4 x_d^4 + a_5 x_d^5 \ldots \tag{2.5}$$

The offset term a_0 in the above equation is not of major concern, since it does not produce an audible frequency. Nevertheless, all terms except the a_1 term contribute to distortion. In order to understand the consequence of the higher order terms, it is assumed that the nonlinear stiffness is an important contributor to distortion. As mentioned previously, this is the case below the resonance frequency. Thus, the output y of the system may be written as in Equation 2.6 as a function of the input i.

$$y = y_0 + \alpha_1 i + \alpha_2 i^2 + \alpha_3 i^3 + \alpha_4 i^4 + \alpha_5 i^5 + \ldots \tag{2.6}$$

It may be assumed that the input is sinusoidal function, e.g. $i = \cos\omega_0 t$. This assumption is very reasonable, since the input signal can be decomposed into an infinite set of sinusoids by the Fourier transform. The higher order terms α_n with $n \geq 2$ in equation 2.6 will now generate harmonic distortion (HD). This can be understood by the notion that $i^2 = \cos^2(\omega_0 t) = \frac{1}{2} + \frac{1}{2}\cos(2\omega_0 t)$. The second order term therefore yields a spectral component with twice the frequency of the input signal. Table 2.1 gives an explicit expression for the powers of the input signal for $i = \cos(\omega_0 t)$ and the spectral components that are introduced.

Table 2.1: Explicit expression for i^n for $i = \cos(\omega_0 t)$. The spectral components that are generated by the higher order terms are listed. The frequency is given as $f = \frac{\omega}{2\pi}$ and DC corresponds to a frequency $f = 0$

n	i^n	spectral components
1	$\cos(\omega_0 t)$	f_0
2	$\frac{1}{2}\cos(2\omega_0 t) + \frac{1}{2}$	DC, $2f_0$
3	$\frac{1}{4}\cos(3\omega_0 t) + \frac{3}{4}\cos(\omega_0 t)$	f_0, $3f_0$
4	$\frac{1}{8}\cos(4\omega_0 t) + \frac{1}{2}\cos(2\omega_0 t) + \frac{3}{8}$	DC, $2f_0$, $4f_0$
5	$\frac{1}{16}\cos(5\omega_0 t) + \frac{5}{16}\cos(3\omega_0 t) + \frac{5}{8}\cos(\omega_0 t)$	f_0, $3f_0$, $5f_0$

As indicated in Table 2.1, the higher order terms introduce frequencies that are an integer multiple of the original frequency. These frequencies are commonly referred to as harmonics, hence the name harmonic distortion. It is suggested by [16] that harmonic distortion does not sound so bad. Unfortunately, the nonlinear system introduces another type of distortion known as intermodulation distortion (IMD) when two or more frequencies are played simultaneously. Supposing the input now consists of two sinusoids with the same amplitude, but different frequency: $i = \cos(\omega_0 t) + \cos(\omega_1 t)$. The second order term now yields: $i^2 = \cos((\omega_0 + \omega_1)t) + \cos((\omega_0 - \omega_1)t) + \cos^2(\omega_0 t) + \cos^2(\omega_1 t)$. The cosine squared terms produce harmonic distortion as seen before. However, additional spectral components with frequencies $(f_0 + f_1)$ and $|f_0 - f_1|$ are created also. These are the intermodulation frequencies, which may be perceived as unpleasant according to [16]. Table 2.2 lists the additional spectral components that are introduced for all nonzero order terms.

Table 2.2: Inter harmonic spectral components that are introduced by the i^n term of the nonlinear transfer, if the input is defined as: $i = \cos(\omega_0 t) + \cos(\omega_1 t)$.

n	spectral components
1	f_0, f_1
2	$f_0 + f_1, f_0 - f_1$
3	$2f_0 + f_1, 2f_0 - f_1, 2f_1 + f_0, 2f_1 - f_0$
4	$2f_0 + 2f_1, 2f_0 - 2f_1, 3f_0 + f_1, 3f_0 - f_1,$ $3f_1 + f_0, 3f_1 - f_0, f_0 + f_1, f_0 - f_1$
5	$4f_0 + f_1, 4f_0 - f_1, 4f_1 + f_0, 4f_1 - f_0,$ $3f_0 + 2f_1, 3f_0 - 2f_1, 3f_1 + 2f_0, 3f_1 - 2f_0,$ $2f_0 + f_1, 2f_0 - f_1, 2f_1 + f_0, 2f_1 - f_0$

Sixth or higher order terms in the nonlinear transfer may generate additional spectral components, but usually these components are quite small. In [35], the higher order spectral components can be seen, but they are below the measurement uncertainty and may therefore be neglected. In the measurements of e.g. [14], the third harmonic component is the most dominant. This implies that the odd terms of Equation 2.6 contribute significantly to the non-linearity. Intuitively, this means that the nonlinear function is more or less odd symmetric. It is stated in [20] that asymmetrical non-linearities generate primarily even-order distortion. It is also mentioned that even-order distortion is perceived as especially unpleasant. The graph in Fig. 2.6 shows the frequency spectrum when distortion is introduced.

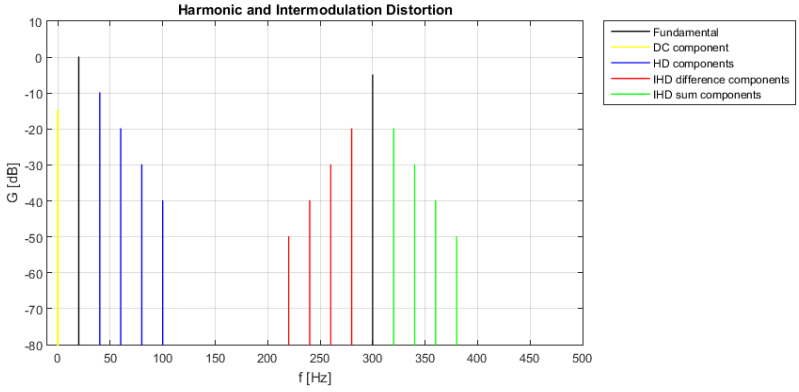


Figure 2.6: Frequency domain visualisation of harmonic (HD) and intermodulation distortion (IMD). Note that the frequency axis is linear. Additional spectral components may arise at higher frequencies, e.g. around 600 Hz but these are not indicated here.

2.3. Situation Assessment

Several solutions exist to tackle the phenomenon of distortion. The solution this project essentially uses is by means of a Motional Feedback controller (MFB). as mentioned in the Introduction. Before designing any type of implementation of this MFB controller, it is needed to know about the past of this concept,

what currently has been designed and what should still be done.

2.3.1. First developments

Phillips introduced motional feedback loudspeakers for the first time in 1968 [19]. The idea was to reduce the linear and non-linear distortion caused by a loudspeaker by including the loudspeaker in a feedback loop. Linear distortion is caused by the uneven distribution of intensity over the frequency spectrum (especially at lower frequencies) because the force needed to move the cone at low frequencies becomes larger and is offset by the stiffness (restorative force). The nonlinear distortion generates higher harmonics. The radiated power was measured using an accelerometer attached to the cone and compared to the input voltage signal. Another advantage of motional feedback was that the speakers could be made smaller in volume without affecting the low frequency bass reproduction. The development of the motional feedback (MFB) loudspeakers was soon discontinued because it was too expensive at that time to compete with conventional high-end loudspeakers [11]. MFB loudspeakers had excellent bass reproduction, however the overall tonal balance and cabinet coloration of the conventional loudspeakers was better. This was the first analogue implementation of the MFB-controller.

2.3.2. Modern Implementation

Motional feedback loudspeakers are no longer being produced by Phillips or any other big brands. However, there is research being done to make a digital implementation of motional feedback possible on loudspeakers. In 2013, R. Valk [35] wrote a thesis on enhancing loudspeaker performance at low frequencies by increasing the bandwidth and decreasing the total harmonic distortion (THD) using motional feedback. The topology used is shown in Fig. 2.7, where P represents the loudspeaker and C represents the controller. Firstly, an accurate linear model of the loudspeaker expressed in multiplications of a series of transfer function has been acquired. The controller is then also implemented as a series of transfer functions in order to place zeros and poles at desired locations to compensate for the distortion caused by the loudspeaker. The controller was implemented on a DS1103 PPC Controller Board which was mounted on a PC. The resolution of the used DAC/ADC was 16-bit with a sampling frequency of 100 kHz. With the controller the THD was reduced by a factor of 11. At 20 Hz with a high level reference signal, the measured THD was under 4%.

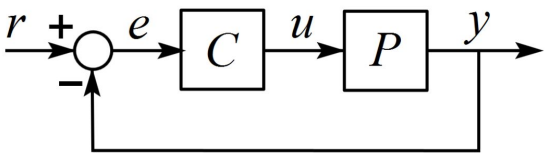


Figure 2.7: Feedback topology used by R. Valk[35]

In 2015, P.L. Torraca [34] derived a theoretical technique for digitally controlling a loudspeaker. Linear and non-liner state-space models were derived from the Beranek and Klipper model, respectively. Using the Klippel Distortion Analyzer the models were characterized and the required parameters were extracted. Using a secondary coil wound over the main voice coil as a sensor, a feedback signal was created. The control system used this feedback and a model-based technique as seen in Fig. 2.8. To elaborate further, the full controller had two controller stages, the first controller used a pole placement technique to control the linearized loudspeaker. The second stage, the compensator, linearizes the loudspeaker. Inside the compensator an estimation of the loudspeaker state is calculated from the output and the loudspeaker model, which is used to pre-distort the input signal. The technique showed good overal theoretical performance, achieving levels below any threshold for high quality audio reproduction. However, the performance degrades when delays are introduced from the digital implementation. It was stated that a delay smaller than 0.5 milliseconds is needed while at the time the available DSP had a fixed delay of 2.8 milliseconds. Also, the loudspeaker has to be accurately characterized and parameters need to be known accurately.

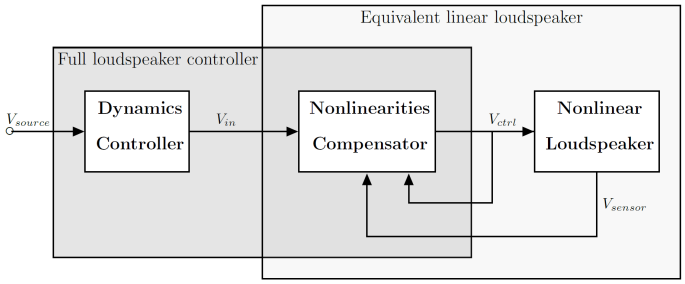


Figure 2.8: Block diagram of the controller [34]

Previously, by one of the groups who also worked in the same project [15], a motional feedback system was designed as seen in 2.9. In this system the analogue power amplifier D provides the loudspeaker G

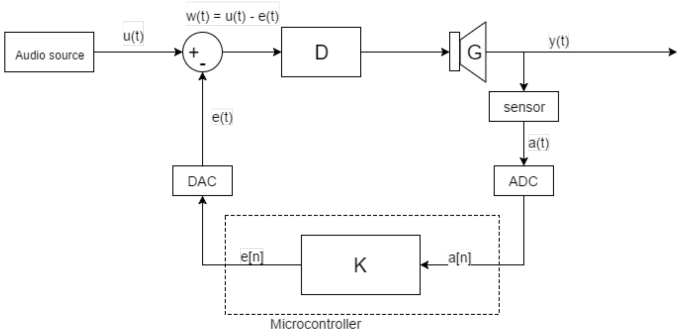


Figure 2.9: Block diagram of the controller. Courtesy of [12].

with energy while the feedback loop uses a low pass filter implemented with controller K to improve the system. The controller K was meant to be implemented with a micro controller. The controller uses the inverse transfer function to calculate the correct correction for the input. This inverse transfer function is implemented with a IIR and calculated using system identification. The controller would also contain a low pass filter in order to add zeros in the transfer function to get the same amount of zeros and poles to enable inversion of the transfer function. From this thesis it is however clear that the introduced delay from a microcontroller is currently still too large to be feasibly used in a feedback loop.

2.3.3. Future work

In the past, great THD reduction was achieved, however the implementations were too expensive for production. With recent advancements in technology,the production costs can reduced by using digital hardware to implement the controller.

3

Programme of Requirements

The products to be developed are an analog and a digital implementation of a motional feedback system for bass loudspeakers, using the feedback signal of a piezo-electric accelerometer mounted on the speaker cone. The system is a low-cost, small format implementation which can easily be adjusted to be used for different speakers with different characteristics. The system is aimed towards commercial loudspeaker manufacturers to be included in active loudspeaker systems. The consumer good must meet or improve on the specifications listed in section 3.0.1, as given by the supervisor, when using motional feedback and improve on the specifications of the loudspeaker when it does not use motional feedback. It also has to be cheaper than other motional feedback loudspeaker systems with similar specifications available on the market.

3.0.1. Requirement formulation

1. **MR:** mandatory requirements

    - A woofer loudspeaker diaphragm is equipped with a piezo-electric accelerometer. The signal thereof is to be included in a negative feedback loop; this principle is known as Motional Feedback (MFB);
    - The system should operate in a bandwidth from $10-300Hz$, however, a $1kHz$ bandwidth is highly desirable. The highest attainable bandwidth is $2kHz$ due to sensor limitations;
    - The cost of the system should be no more than € 100 .
    - The volume of the controller should be 0.5 l maximum.
    - The Total Harmonic Distortion (THD) should be reduced to 0.1%;
    - The largest acceptable delay that is introduced as a result of the controller is 120 ms. This is the delay that the user may experience when playing sound through the system.
    - The power consumption of the controller should be 100 mW.

2. **ToRs:** Trade-off requirements

    - The desired Signal to Noise Ratio (SNR) is at least 100 dB. Nevertheless, a 16 bit digital system may offer some advantages due to lower delays and lower cost. The SNR of a 16 bit system is at most 98 dB, but this acceptable also according to the client;
    - The system is optimized for the specific loudspeaker and amplifier that have been made available for this project. The system should ideally be also applicable to other configurations, considering the typical amplifier gain is $20-30$ dB.
    - The system must be stable, which implies that both the gain and phase margins must be reasonable. Precise minima were not given, but a phase margin of 45 degrees was proposed, alongside a gain margin of 3 dB.

3.0.2. Study-case

1. *Functional Requirements*
    1.1. The MFB system must operate whenever the loudspeaker system is turned on without requiring additional steps from the user to turn on motional feedback.
    1.2. The loudspeaker system's user interface may contain a switch to turn motional feedback on and off.
2. *System Requirements*
    2.1. *Utilisation features*
        2.1.1. The lifespan of the feedback controller and accelerometer must be at least as long as the lifespan of the loudspeakers in which it is included.
        2.1.2. If support and/or maintenance is provided for the loudspeaker system, this must include support for the MFB system.
    2.2. *Production and putting into use features*
        2.2.1. Inclusion of the MFB system must take place during the development of the loudspeaker system in cooperation between the loudspeaker manufacturer and our company e-motion.
        2.2.2. The loudspeaker must undergo testing by e-motion before and after the inclusion of the MFB system to ensure MFB meets the specifications.
        2.2.3. e-motion will provide the piezo-electric accelerometer and controller to the loudspeaker manufacturer. The manufacturer must install the MFB hardware into the consumer product during assembly. Placement of the controller inside the loudspeaker will be discussed with the manufacturer on a case by case basis.
    2.3. *Discarding features*
        2.3.1. If the hardware of the MFB system is enclosed in a casing, the casing must be made from recyclable materials.
        2.3.2. In case the MFB system's lifespan exceeds that of the speaker itself, the manufacturer must provide to the consumer the option of returning the MFB hardware for use in a refurbished product
3. *Development of manufacturing methodologies*
    3.1. The digital version of the MFB controller will be implemented as an ASIC.
    3.2. The ASIC must be adjustable after manufacturing to meet specifications in any loudspeakers in which it is included; Only one version of the ASIC will be developed and manufactured.
    3.3. A protocol and measurement setup will be developed for quick testing and validation of the loudspeaker system before and after the inclusion of MFB. Testing on a loudspeaker must not take longer than 1 hour.
4. *Liquidation/recycling methodologies*
    4.1. At the end of the product's lifespan, the discarding thereof must comply to the norms referring to processing of small chemical waste.
5. *Business strategies, marketing an sales opportunities*
    5.1. The manufacturer of the loudspeaker must explicitly state the inclusion of the MFB feature in the packaging and documentation of the final product
    5.2. the logo of e-motion must be included on the packaging and casing of the final product by the manufacturer.

4

Topology

Possible controller topology solutions to the problem stated in the introduction, chapter 1, are highlighted in this section. At the end, one topology is chosen.
An abstract overview can be achieved when the entire system to be controlled is captured in a single block. This single block called "plant" includes the amplifier, loudspeaker and accelerometer in one. In control engineering, the plant is a system that needs to be controlled. The system identification will, later on, also be performed for the entire plant, as seen in the next chapter, chapter 5.

4.1. Feed-forward

When the plant has been correctly identified and a plant model has been derived. This plant model can be used internally in the controller for the computation of the correct control signal, and thus feed-forward can be used. The controller supplies an input to the plant in such a way that the plant produces the desired output [25].

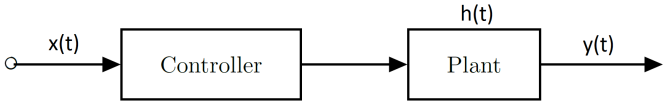


Figure 4.1: Feed-forward control system structure. Adapted from [34]

With feed-forward no information about the real plant is gathered in real time, only a model is used. This model should therefore approximate the plant sufficiently. As mentioned in the technical report by Robert-H Munnig [28], in mechanics three dynamic elements can be defined. They are spring, damper and mass. They will provide passive feedback for respectively the displacement, velocity and acceleration. When there are real dynamic elements in the plant, in this case the speaker, this will cause non-linearities. The first step to compensate those can be to model the dynamic elements. After that, with feed-forward the inverse of the plant can be provided to the plant with the controller to compensate them. In a more mathematical notation:

$$y = H * x \tag{4.1}$$

$$y = H * H^{-1} * x = x \tag{4.2}$$

In which x is the input signal, y is the output and H is the transfer function of the plant. Note that H needs to be invertible, or in other words it should be possible to derive the input from the output. Thus mathematically, matrix H should be square with a non-zero determinant

The big advantage of feed-forward is that no noise from the sensor is introduced and the reaction speed of the system is faster than with a feedback system.

It is however, noted in a paper [34] that an error in the model, will *easily cause the compensation to work out of phase of the problem*. Which can increase the error instead of decrease it. An error in the model can originate from e.g. temperature change, position of the cone and non-linear self inductance.

But, in the same paper [34] it is also noted that not requiring information about the real plant has the advantage that the plant does not have to be disturbed or impractical measurements have to be taken. When however, the real plant and the model, used in the controller, are misaligned with each other this can cause loss of robustness, performance degradation and even instability of the system.

4.2. Adaptive Feed-forward

From paper [28] it is seen that with adaptive feed-forward the same basic idea of feed-forward is used. There is however an addition of an observer which observes the behaviour of the plant. When the plant behaviour changes, the observer will correct the model parameters in the controller in a process called innovation [28]. Note that the observer is included in the controller in Fig. 4.2. This adaptive feed-forward works well with repeating actions such as the loudspeaker membrane motion as noted in the same paper.

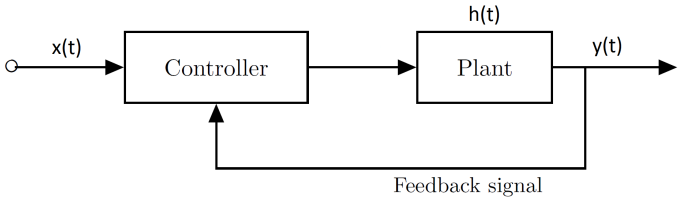


Figure 4.2: Adaptive Feed-forward control system structure. Adapted from [34]

A trade-off for corrections of the controller between speed and estimation can be made. The optimal trade-off is called a Kalman-filter. It is stated in [21] that there are two ways to adjust the parameters of the controller in a feed-forward configuration to the real plant:

- Indirect updating using an additional adaptive system which compares the model of the plant with the real plant and infers the required parameters from it, which it thereafter transfers to the controller. Indirect updating can be done in series or in parallel to the plant. With both indirect and direct methods the error signal is directly calculated using the output. Using an inverse plant model the parameters can be calculated. In the parallel plant modelling two algorithms of interest can be distinguished. The filtered-gradient algorithm and the filtered-error algorithm.
- Direct updating which uses the non-linearities of the plant in the update process. It is shown that for small error signals some complicated calculations can be ignored. Which is very useful when implementing an observer in a DSP (Digital Signal Processor) because it keeps the required processing power lower. Without the complicated calculations, two algorithms can be used. Intermittent filtered-error LMS and intermittent filtered-gradient LMS as also stated in [21].

4.3. Motional Feedback

The idea behind motional feedback is that a feedback signal is generated from a measurement on the cone movement. One of the first implementations of this is discussed in the article by Philips [19]. In this paper, an accelerometer was used to generate the feedback. Since then, a lot of different implementations of motional feedback have been made, both using just analogue circuits as well as digital. Some of these implementations will be discussed below.

4.4. Full State Feedback

In full state feedback (pole-placement), not the output of the system is taken into account but the internal state of the system itself, by measuring a variable of the plant. The state space representation of a linear

system is given in equation 4.3, where **x** represents the states and the eigenvalue of **A** represents the poles of the system. By placing the poles of a system in desired locations, the system response can be controlled. The linear distortion of the system (in this case a loudspeaker) can be compensated by adding pre-distortion to the input signal. By placing poles in the appropriate positions, such a pre-distorted signal can be created. When inputted to the loudspeaker this will cancel out the distortion of the loudspeaker. A more powerful version of state feedback is observer based state feedback, where the output and the input signal are also used to estimate the full state of the system.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + f(\mathbf{x}, \mathbf{u}), \\ y &= \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} + g(\mathbf{x}, \mathbf{u}) \end{aligned} \tag{4.3}$$

4.5. Observer Based State Feedback

Observer based state feedback is similar to full state feedback but with an added state observer. An observer allows access to more state variables, which cannot be measured with identification, by also measuring the output and input signals at real time to determine the system state more accurately. This method is especially useful to implement due to the fact that a loudspeaker is a time invariant system. The state of the system is observed using a transducer and is compared to the acquired estimated state using a state-space model. The observed state will however be inaccurate due to measurement noise and due to the fact that the estimated state is inherently inaccurate because not all variables can be accessed in the loudspeaker. To acquire the optimal state estimation with minimal error, Kalman filtering is used. A similar method, model-following control, has been presented and simulated by Kadowaki and Samejima [18] to reduce non-linear distortion in loudspeakers. A reference model and an observer (velocity sensor) were used to estimate the state of the system. The method was verified using simulations and the THD was reduced from a maximum of 55% to under 5%. It was concluded that the method can be implemented on a DSP using an IIR filter of around tenth order.

The loudspeaker being a non-linear system, a non-linear space-state model has to be implemented for an accurate state estimation. A method called Polynomial Nonlinear State-Space (PNLSS) modelling has been studied by Brunet [5], with application to nonlinear modelling of loudspeakers. The modelling procedure consists of two parts, frequency domain identification for the linear parameters and time domain nonlinear optimization for the nonlinear parameters of the model.

A similar method as the Observer based State Feedback has also been implemented by La Torraca [34] for a Loudspeaker. The sensor used is a magnetic velocimeter to measure the velocity of the voice coil. In the Nonlinear Dynamics Compensation Controller block in Fig. 4.3, the observed state has been used to create a pre-distorted signal by pole-placement (Full State Feedback) such that the input and output signals of the whole system is linear. After implementing the controller, La Torraca concluded that the delay introduced by the DSP must be no larger than 0.7 ms in order to keep THD uder 1%. The DSP La Torraca used was a Cirrus Logic CS47024 and the total delay introduced was 2.8 ms.

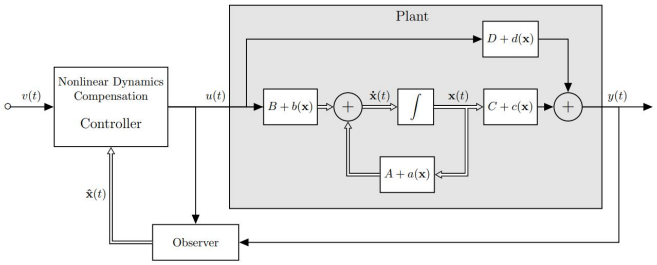


Figure 4.3: Controller topology implemented by La Torrace [34]

4.6. Final Topology Choice

Feed-forward has the advantage that a phase delay is irrelevant because there is no direct feedback. Therefore, a better analysis (which requires more computation time) can be performed, from the supplied audio, before an input is supplied to the plant. For example, samples can be buffered and analyzed (e.g. using a FFT to analyze the frequency domain spectrum). With feed-forward however, the non-linear distortions will not be suppressed. This will therefore not achieve the required suppression of distortion as stated in chapter 3.0.1. Moreover, an initial test with pre-distorting a complete audio sample before inputting it to the plant failed (i.e. produced noise). It is expected that the plant model is insufficient at the moment. Because the distortions will not be suppressed and the initial test failed, it was decided to not research (adaptive) feed-forward any further.

Feedback does, contrary to feed-forward, provide suppression of non-linear distortion. Care should however be given to the (introduced) phase shift. A phase shift will cause performance degradation and in extreme cases, when the phase shift exceeds 180 degrees while the gain is equal or larger than 0 dB, the system will become unstable. Thus, feedback can only be used when the delay introduced by the digital controller is small enough.

No accurate model of the loudspeaker is available at the moment because the theory group, which intended to supply the model, had too many complications. Therefore, full state feedback and observer based state feedback are both not feasible in this case.

Because negative feedback will inherently improve the system, even without a good model, it is chosen to use the (basic) feedback topology. In order to improve the response further, it is also decided to use a controller and a filter (i.e. equalizer) along with negative feedback.

5

System identification

In order to design a proper controller for the loudspeaker, an accurate model of the loudspeaker needs to be made. The loudspeaker is a non-linear system. The sources of non-linearities of the loudspeaker is investigated by another group [27] to create a non-linear model. In this thesis, a linear model of the loudspeaker has been used to develop a digital controller for the loudspeaker.

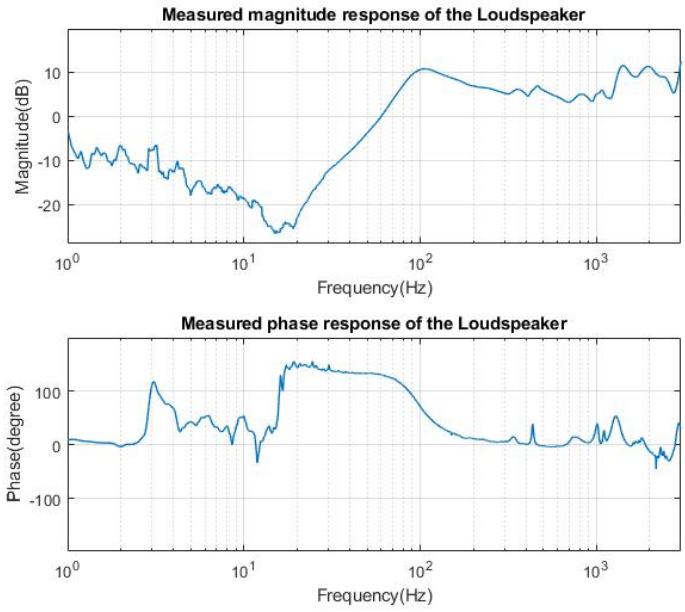


Figure 5.1: Measured frequency response of the loudspeaker acquired using LS-measure [17]

The linear model has been derived from the measured frequency response of the loudspeaker (see Fig. 5.1). To acquire the measurement data, LS-measure [17] has been used. Uniform white noise is used as input signal to the loudspeaker and the output has been measured with the accelerometer. In LS-measure, magnitude and phase responses are computed using the input and output data, and are stored in separate vectors. The separate vectors have been combined in Matlab into a single complex matrix that represents the frequency response using equation 5.1.

$$\vec{H} = |\vec{H}| \cdot e^{\angle \vec{H}} \tag{5.1}$$

Using the complex vector data, a transfer function is estimated in Matlab using 'tfest'. Tfest [33] is a tool in Matlab that estimates a transfer function using time or frequency domain data. Once the order of the transfer function is specified by the user, the parameters of the transfer functions are determined using the least-square method.

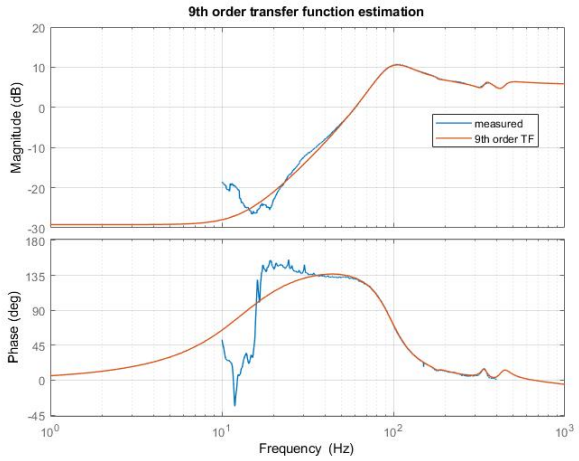


Figure 5.2: Bode plot of the 9^{th} order transfer function and measured data

5.1. Continuous transfer function estimation: 9^{th} order

The best fit to the measured data has been acquired with a 9^{th} order transfer function. The accuracy achieved between the transfer function and measured data was 97.04% in the frequency range of 10-400 Hz. In Fig. 5.2, the bode plot of the transfer function can be visually compared to the measured frequency response of the loudspeaker.

The model represented by the transfer function is BIBO stable as all the poles and zeros lie in the left half of the s-plane, see Fig. 5.3. The ninth pole can not be seen in the figure because it lies far to the left of the poles and zeros shown. In Fig. 5.4, the stability margins of the system can be seen. The gain margin of the system is infinite as the phase response never crosses 180°. The phase margin is -48.3°, which means the system will be unstable if a phase shift of more than +48.3° is introduced. Fig. 5.4 shows that the system only has a positive phase response, which means that the output of the system leads the input for frequencies starting at 1Hz till 500Hz.

5.2. Continuous transfer function estimation: 2nd order

Looking at the measured frequency response, the magnitude response shows a slope of about 40 dB/dec between frequency 10 and 100 Hz. Above 100 Hz, the response seems to be flat. This system could thus also be estimated using a second order transfer function. Using a second order model instead of the ninth order would reduce the complexity of the controller. Again, 'tfest' was used to acquire the second order transfer function and the frequency response of the acquired transfer function is shown in Fig. 5.5.The phase response of this transfer function is incorrect because the measured phase response of the loudspeaker is 0° at 1 Hz (see Fig. 5.1). However, the 9th order transfer function acquired using 'tfest' did give the correct phase response, which is why the 9th order transfer function is taken as a reference to make the second order model.

In the pole zero map of the 9th order estimated system several poles and zeros cancel out each other as they lie almost on the same points on the s-plane (see Fig. 5.3). Table 5.1 contains the same poles and zeros with their location in frequency domain. The corresponding frequencies of poles and zeros have been

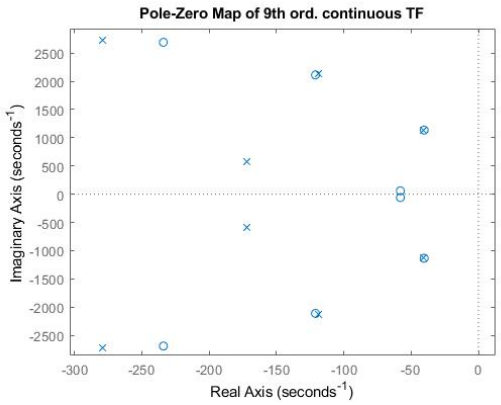


Figure 5.3: Location of poles and zeros in complex s-plane

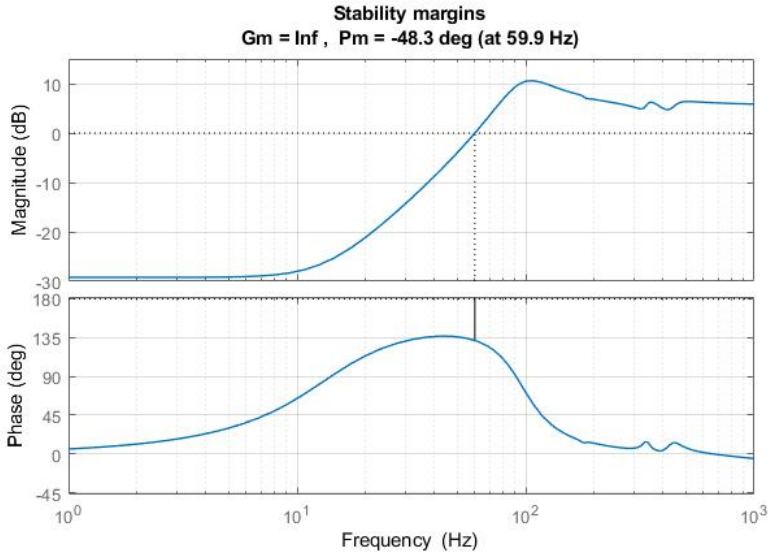


Figure 5.4: Gain margin and phase of 9^{th} order transfer function

calculated using equation 5.2. There are two poles and two zeros at 181 Hz which cancel each other out. There are also two poles and two zeros between 337 Hz and 341 Hz, which can also be canceled out. The two zeros at 430 Hz, two poles at 437 Hz and one pole at 6.59 kHz can be left out as these fall outside of the required 10-300 Hz range.

There are two zeros at 13.2 Hz, meaning that the magnitude response has a slope of +40 dB/dec starting at 13.2 Hz. The two poles at 96.8 Hz indicate that the slope of 40 dB/dec will be made flat after 96.8 Hz. This behaviour of the loudspeaker model can be confirmed by looking at Fig. 5.2, where the magnitude response at 10 Hz is approximately -30 dB and at 100Hz approximately 10 dB, thus an increase of 40 dB in one decade. The phase response can also be confirmed and the phase has a slope of 180°per decade at around 10 Hz and -180°per decade at around 100 Hz.

$$s = j\omega = j2\pi f \tag{5.2}$$

Using table 5.2, a second order transfer function has been derived and is represented by equation 5.3. The value of K is determined to be 1.853 from the magnitude shift between the measured frequency response

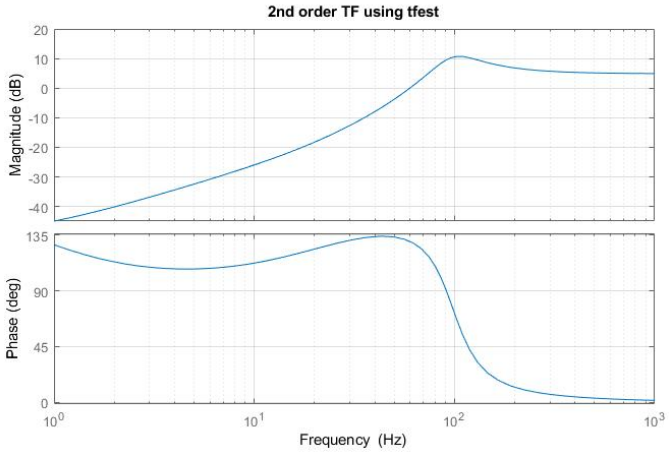


Figure 5.5: Bode plot of the 2nd order transfer function acquired using 'tfest'

Table 5.1: Zeros and poles of 9th order TF and their corresponding frequencies

zeros	$f_{zeros}(Hz)$
-57.9 - 59.7j	13.2
-57.9 + 59.7j	13.2
-40.2 - 1135.1j	181
-40.2 + 1135.1j	181
-121 - 2111.9j	337
-121 + 2111.9j	337
-234 - 2689.7j	430
-234 + 2689.7j	430

poles	$f_{poles}(Hz)$
-172 - 583j	96.8
-172 + 583j	96.8
-41 - 1133j	181
-41 + 1133j	181
-119 - 2137j	341
-119 + 2137j	341
-279 - 2731j	437
-279 + 2731j	437
-4.1405	6590

and the response of the derived transfer function with $K = 1$. This is the constant gain factor of the accelerometer at the output. Thus, when a 1 V signal is fed into the system, the sensor output is 1.853 V. The bode plot of this transfer function is shown in Fig. 5.6. The accuracy of the 2nd order transfer function estimation is 91.6% between 10-300 Hz.

$$H(s) = \frac{K(s+57.9+59.7j)(s+57.9-59.7j)}{(s+172+583j)(s+172-583j)} \tag{5.3}$$

$$H(s) = \frac{1.853s^2 + 214.7s + 1.283 \cdot 10^4}{s^2 + 344.3s + 3.697 \cdot 10^5} \tag{5.4}$$

5.3. Discrete transfer function estimation: 2nd order

The tuning of the controller will be done using the 2nd order continuous transfer function as the loudspeaker itself is a continuous system. However, simulating a continuous system on an FPGA is not possible (please refer to chapter 7 for the choice of hardware). To simulate the working of the controller on an FPGA, a discrete version of the transfer function has to be made. The controller will be simulated in both Matlab (Simulink), where the controller is also designed, and in Modelsim, where the FPGA design is simulated. The purpose of the discrete transfer function is to allow us to compare the simulation results of Matlab and Modelsim and verify that the controller designed in Matlab behaves correctly when implemented on the FPGA.

Table 5.2: Remaining Zeros and poles of 9th order TF and their corresponding frequencies

zeros	$f_{zeros}(Hz)$
-57.9 - 59.7j	13.2
-57.9 + 59.7j	13.2

poles	$f_{poles}(Hz)$
-172 - 583j	96.8
-172 + 583j	96.8

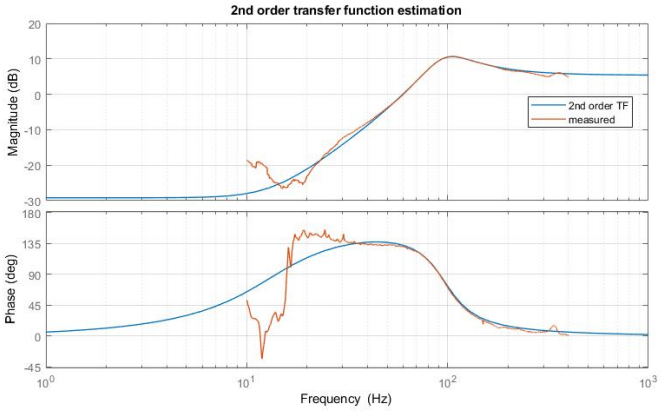


Figure 5.6: Bode plot of the 2nd order transfer function and measured data

The continuous transfer function is discretized i.e., transformed from s-domain to z-domain. In Matlab, this is done using the 'c2d' function [31]. The principle behind the discretization is given by equation 5.5, where F_s is the sampling frequency. The discrete transfer function is given by equation 5.6. It is confirmed that the frequency response of the discrete transfer function corresponds to the continuous transfer function.

$$z = e^{\frac{s}{F_s}} \tag{5.5}$$

$$H(z) = \frac{1.853 - 3.701z^{-1} + 1.848z^{-2}}{1 - 1.993z^{-1} + 0.9929z^{-2}} \tag{5.6}$$

6

Concept Design

After acquiring the model of the loudspeaker, a controller was to be designed. The chosen topology in chapter 4 is negative feedback with a controller. The effect of only feedback is shown in Fig. 6.1. The resonance peak has been suppressed and the frequency response above 100 Hz is flat at 0 dB. However, the attenuation of the lower frequencies is not compensated. A PID controller was chosen because it has a desirable frequency response as it can be seen in Fig. 6.2. The integrator part of the PID is responsible for the amplification at lower frequencies and the differentiator for amplification at higher frequencies. Also, an integrator acts as a lag-compensator, adding -90° phase shift, and the differentiator as a lead-compensator, adding +90° phase shift. As already shown in Fig. 6.1, the amplitude response for higher frequencies is already corrected by just negative feedback. A PI controller with a negative feedback loop is thus sufficient for the loudspeaker because it corrects both the magnitude response and phase response at low frequencies.

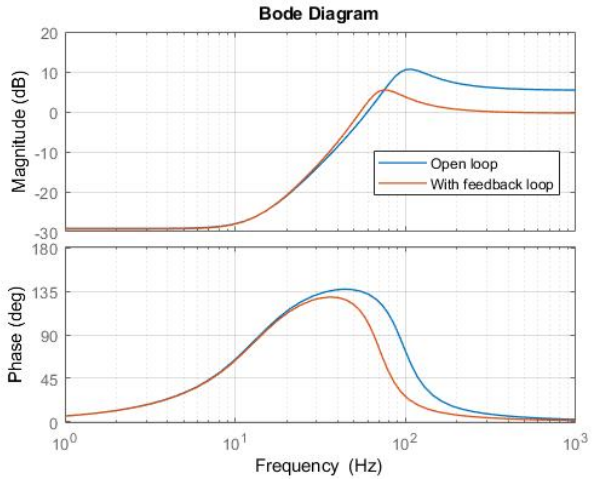


Figure 6.1: Frequency response of the model with and without feedback

6.1. PI controller

A PI controller is represented by equation 6.1, where $u(t)$ is the output of the controller, $e(t)$ is the error, K_p is the proportional gain and K_i is the integrator gain. There are several discretization methods of the PI controller, but the discretization method that is used is the same as that Matlab uses, which is the backward Euler discretization method. The discrete time PI controller is represented by equation 6.2.

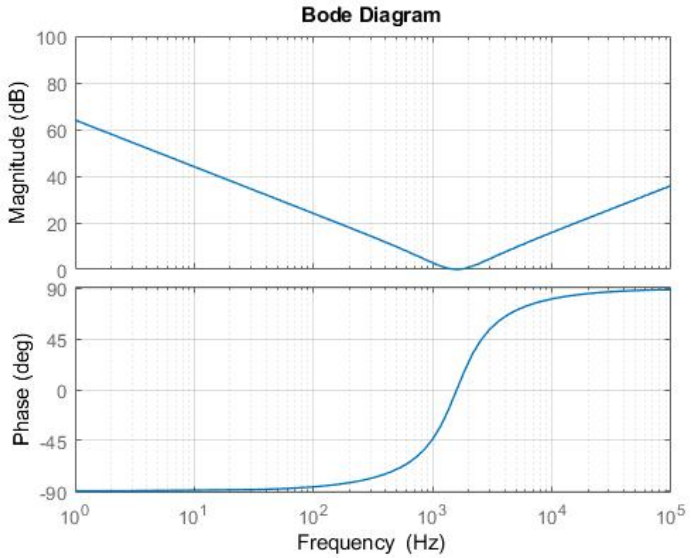


Figure 6.2: Frequency response of a PID controller

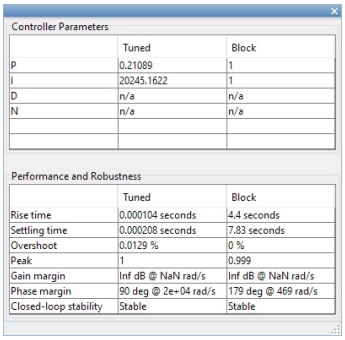
$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt \tag{6.1}$$

$$u(k) = K_p e(k) + K_i T_s \frac{1}{1 - z^{-1}} e(k) \tag{6.2}$$

To tune the PI controller for the loudspeaker, the parameters K_p and K_i are determined using the model. The tuning of the controller is done in Simulink using the PID Tuner App [32]. As reference a step function has been used. The controller has been tuned for the following performance requirements : settling time, overshoot and minimum phase margin.

Settling time is the time that the system requires to settle within 95% of the input reference. In our case, the signal at the input of the system is sampled using an ADC at a certain sampling frequency. The sampling frequency is chosen to be 4.8 kHz (the choice of this sampling frequency has been motivated in chapter 7). This means that the system gets a new reference voltage at the input every $1/4800 = 0.000208$ seconds. Thus, the system has a maximum of 0.000208 seconds to settle at the correct output, so that the output of the systems follows the input precisely. The overshoot, given in percentage, is by how much the output exceeds the input reference. This has to be 0% in order for the output and input to be the same. The minimum phase margin that the system should have is at least 45°. This means that the system remains stable for an added phase shift of up to 45° by unforeseen disturbances.

In Fig. 6.3 the parameters of the PI controller and the performance comparison after tuning the controller is given. With $K_p = 0.21089$ and $K_i = 20245.1622$, the settling time has been reduced from 7.83 to 0.000208 seconds. The overshoot has become 0.013% from 0%, which is caused due the reduction of settling time from by almost 40,000 times. Given the huge improvement in settling time, the 0.013% overshoot which is a factor of 0.00013 had to be accepted. The phase margin has become 90° from 179°. The reduction in phase margin is caused by the integrator. However, the phase margin of 90° still satisfies the requirement. In Fig. 6.4, the reference tracking of a step input is shown. The frequency response of the whole system with the tuned PI controller is shown in Fig. 6.5.



Controller Parameters

	Tuned	Block
P	0.21089	1
I	20245.1622	1
D	n/a	n/a
N	n/a	n/a

Performance and Robustness

	Tuned	Block
Rise time	0.000104 seconds	4.4 seconds
Settling time	0.000208 seconds	7.83 seconds
Overshoot	0.0129 %	0 %
Peak	1	0.999
Gain margin	Inf dB @ NaN rad/s	Inf dB @ NaN rad/s
Phase margin	90 deg @ 2e+04 rad/s	179 deg @ 469 rad/s
Closed-loop stability	Stable	Stable

Figure 6.3: PI controller parameters and the performance comparison. Column 'Tuned' represents the system with the tuned PI controller and column 'Block' represents the systems with the untuned controller.

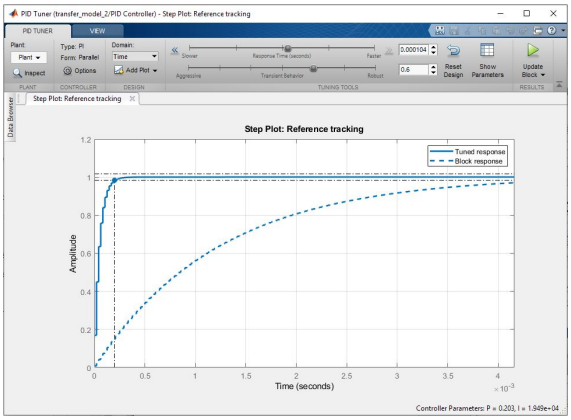


Figure 6.4: Response of the system with tuned controller to input step reference

6.2. Linkwitz transform

The Linkwitz transform circuit is an analog solution widely used to improve the low frequency response of a loudspeaker. It is a second order filter that equalizes the -40 dB/decade slope of the loudspeaker at low frequencies and suppresses the cone resonance. In the digital domain, such an equalizer can be made by taking the inverse of the loudspeaker model and implementing it as an IIR filter. The effects of the equalizer is shown in Fig. 6.6. Only the linear distortion is suppressed with the equalizer. In order to suppress the non-linear distortion, which can not be seen in the linear model, negative feedback is necessary. A combination of the equalizer and PI controller can be used to effectively reduce the linear and non-linear distortions. The advantage of adding the equalizer is that more loop gain of the PI controller is available for the suppression of non-linear distortion as the linear distortion is (mostly) suppressed by the equalizer.

6.3. Simulink model

The controller topology has been implemented along with the loudspeaker model in Simulink, see Fig. 6.7. The second order transfer function represents the loudspeaker and the discrete PI controller and equalizer make up the digital controller. The input of the system is a sinusoidal signal generator which supports frequency sweep. Two Analog-to-Digital converters are placed at the input and a Digital-to-Analog converter at the output of the digital controller.

6.4. Simulink Simulation

Simulations of the system have been done in Simulink with and without the controller. The simulation results of the loudspeaker model only are shown in Fig. 6.8. The frequency of the input signal increases from 10 Hz at t = 0 to 150 Hz at t = 1. The simulation shows the attenuation of low frequencies and a

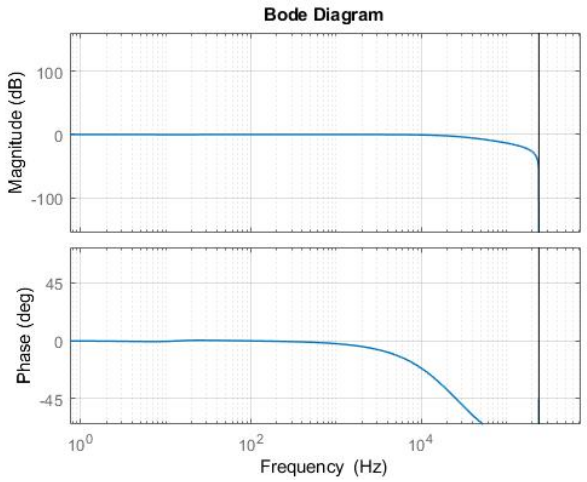


Figure 6.5: Frequency response of the whole system with PI controller

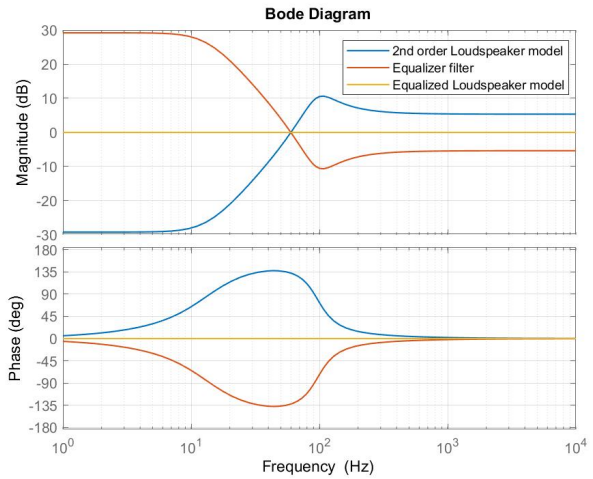


Figure 6.6: Equalization by using the inverse of the loudspeaker model

resonance peak at about 100 Hz, which corresponds to the real loudspeaker.

Fig. 6.9 shows the simulation results of the loudspeaker model with the controller. The output of the loudspeaker model follows the input signal almost perfectly even at the lowest frequencies. At 100Hz, the resonance peak has also been suppressed.

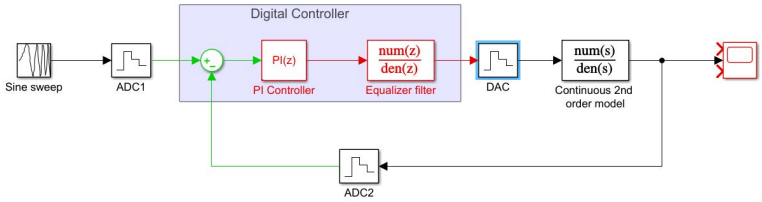


Figure 6.7: Simulink model of the loudspeaker and controller

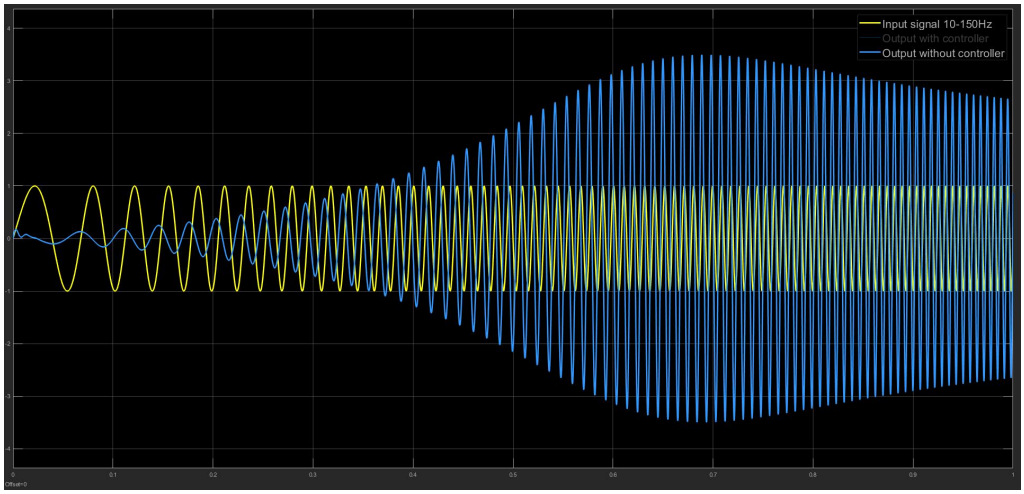


Figure 6.8: Simulation of the loudspeaker model in Simulink

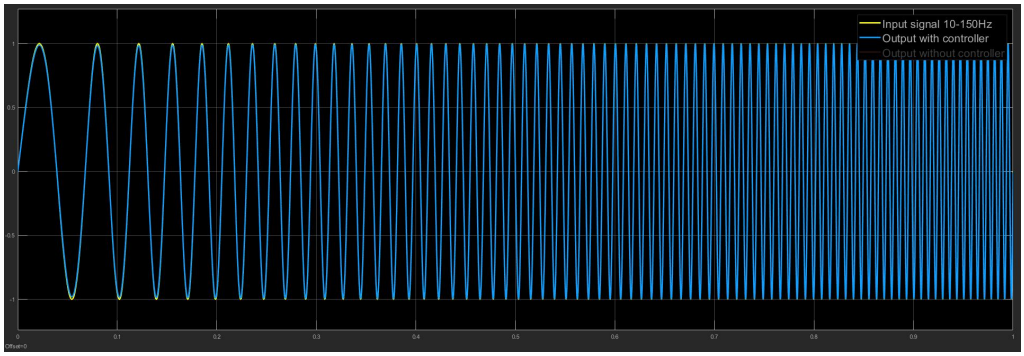


Figure 6.9: Simulation of the loudspeaker model with the controller and feedback loop in Simulink

7

Hardware choice

Multiple devices are available which can be used to implement a controller. Four of the major ones are highlighted below.

7.1. Microprocessor

The microprocessor is a small computing device consisting of a processor and peripherals on one board. A special kind of software is required to handle all the components on the board (i.e. an Operating System). Using this OS, programs written in a high level language can be run on top of it. Therefore, the microprocessor can be quickly programmed. The OS however, will produce overhead running the program, which negatively influences the speed of the calculations and events that can be handled. Therefore, the microprocessor will produce a significant amount of delay between the input and output ports.

7.2. Micro controller

The micro controller also has a board with a processor on it, similar as the microprocessor. However, this board is designed to be programmed with a low level language like C and a lot more control and responsibility is left to the programmer. From [3] it is seen that the device has potential to be used. However as seen in [24], the device will likely introduce too much phase delay and thus cannot be used in the feedback topology.

7.3. DSP

The Digital Signal Processor (DSP) can be regarded in some ways to be similar to a microprocessor. However, an important distinction should be made in that it has dedicated hardware for signal processing and often it has a vector processor. With these, the DSP can be faster than a regular micro controller. But this can also be a major drawback, this is because a vector processor will require more time and knowledge to program. Also, from [34] it is clear that the phase delay can still be too large. Moreover, after looking at currently available DSPs, it became apparent that two kinds of DSP devices are available right now. The first one is a barebone only DSP processor square without a board and without support components while the other kind is a full featured DSP which only allows the user to change some values with a dedicated Graphical User Interface (GUI). Lastly, the DSP processors will be expensive in development and production.

7.4. FPGA

The Field Programmable Gate Array (FPGA) is an integrated circuit which is designed to be configured after it is manufactured. An FPGA is in general faster than a microcontroller because the FPGA will use dedicated hardware, which is also parallelizable, for a task while the microcontroller uses a general purpose processor running at least an assembly level programming language, causing overhead. Designing

and implementing the dedicated hardware with for example VHDL will however require more development time compared to writing high level code (i.e. C). Therefore, the development time can be relative longer for the FPGA compared to a microcontroller. Moreover, an FPGA board is in general more expensive than a microcontroller board thus increasing development costs even more. In production however, the hardware design for the FPGA can be implemented in hardware as an ASIC and thus will be cheaper than a microcontroller. Note that the costs are lower for large quantities only.

7.5. Final Hardware Choice

When choosing for a feed-forward topology (e.g. Observer based feed-forward) a microprocessor or micro controller can be used because there is no direct closed loop and thus there are no phase delay problems. When storing multiple samples, analyzing them and then outputting, a total delay should however be taken into account. The delay is required to be small (i.e. $<$ 120 ms as given in chapter 3) to prevent desynchronization between video and sound.

In chapter 4 it is however, chosen to use a feedback topology. A major concern for the hardware will thus be the phase delay and not the total delay. In literature it is noted that a microprocessor and microcontroller are not sufficient in a feedback topology because of their phase delay. (e.g. [24]).

Other research showed that digital signal processors (DSP) that are available right now also have drawbacks which make them unsuitable for our purpose. For example, some DSPs do not have surrounding hardware which is required to function, others cannot be freely programmed or require an external computer connected to function.

In one paper where a DSP was used to implement a digital Motional Feedback (i.e. [34]) it is recommended to use an FPGA because of the lower phase delay compared to other devices (i.e. microprocessor, microcontroller and DSP).

An FPGA will provide small phase delays but will also require more development time to program than the other mentioned devices. Taking also into account the importance of a cheap product, less than 100 euros, when selling as stated in chapter 3.0.1. The choice was made to use an FPGA for development and aim for selling an ASIC when producing in large editions. The Zybo Zynq-7000 development board FPGA from Digilent [26] is made available to us by Rene van Leuken.

As for the analog to digital converters (ADCs), the minimum sampling frequency had to be determined. The required bandwidth of the loudspeaker with the controller is 10-300 Hz. However, the loudspeaker being a non-linear system, higher harmonics will be produced even if the input signal is band-limited to the required frequency range. The accelerometer, which is used to measure the output signal of the loudspeaker, has a maximum bandwidth of 2 kHz. Thus, harmonics of frequencies up to 2 kHz can be sampled using an ADC. In order to satisfy the Nyquist-Shannon sampling criterion, ADC must have a sampling frequency equal or higher than 4 kHz.

The peripherals which are available with the provided FPGA include a 4.8 kHz 24 bit ADC, 1 MPS 12 bit ADC and a 16 bit DAC. From the requirements, chapter 3.0.1, it is seen that a minimum SNR of 100 dB is required.

Equation 7.1 can be used to calculate the required bits according to a desired SNR.

$$N = \frac{\text{SNR} - 1.76}{6.02} \tag{7.1}$$

with N as number of bits required and SNR the Signal to Noise Ratio requirement. From this it is clear that at least 17 bits are required to achieve the required SNR. The obvious choice will therefore be to use the 24 bit ADC.

Do note however, that with oversampling the resolution in number of bits can be increased [2]. However, using equation 7.2 as stated in the same paper, it is seen that using this technique the 1 MPS 12 bit ADC can barely achieve around 16 bits resolution when targeting a sampling frequency of around 4 kHz. Moreover, the sample averaging also requires more logic to be implemented, this will increase the required development time.

$$f_{os} = 4^{w} * f \tag{7.2}$$

with f_{os} the oversampling frequency, w the extra bits of resolution and f the target sampling frequency.

Thus, it was decided to use an FPGA with a 4.8 kHz 24 bit ADC and a 16 bit DAC.

8

Detailed design

The concept design has been further worked out so that it can be implemented on the chosen hardware. Firstly, the standard components that were available in Simulink, like the PI controller block and filter blocks, have been recreated using simpler components like adders, multipliers and delay buffers. Secondly, the recreated design with simpler blocks was implemented in Modelsim. Modelsim is a software program in which the FPGA code, written using VHDL, is compiled and simulated before uploading the code to the FPGA. Finally, the simulation results of Simulink and Modelsim are compared to confirm that the controller behaves as intended on the FPGA.

8.1. Realization of digital components

Implementation of digital filters on hardware can be done in four structures: Direct Form-I (DF-I), Direct Form-II (DF-II), transposed Direct Form-I and transposed Direct Form-II. DF-I and DF-II realizations of a second order filter, with transfer function given by equation 8.1, are shown in Fig. 8.1. The advantage of using DF-II over DF-I is that it uses less delay elements which means less hardware resources are used. However, its main disadvantage is that internal overflow can occur when using fixed-point arithmetic, which is never the case when DF-I is used [30]. The transposed DF-I and DF-II are acquired by reversing the direction of the signal and interchanging branch-points and summers. The transposed DF-II does not have the property of having internal overflows, however for simplicity DF-I is used as it is straight-forward and is numerically robust.

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \tag{8.1}$$

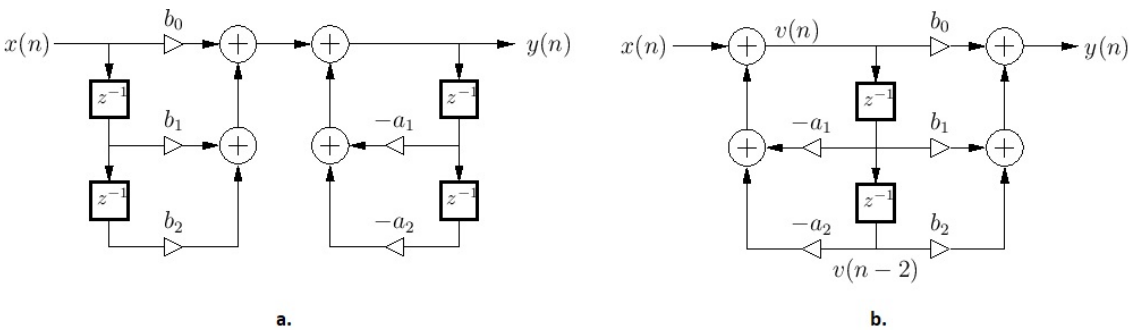


Figure 8.1: (a.) Direct form-I realization of second order IIR Filter. (b.) Direct form-II realization of second order IIR Filter. [30]

The different components are designed and tested below and thereafter combined into one system. Each component will have test results of a Simulink and a Modelsim simulation. From now on, the name "loud-

speaker" and "plant" are used interchangeable, referring to the system to be controlled as defined in the system identification in chapter 5.

Unlike with Simulink, it is not trivial to produce a frequency sweep with Modelsim. This is because a sine wave has to be manually generated with an algorithm or with a Read Only Memory (ROM). Because of development time, it was chosen to use a ROM with predefined sine values calculated with Matlab. Different frequencies are chosen using a defined step size when reading the values from ROM. Therefore, only one frequency is shown graphically with a Modelsim simulation and at some specific frequencies a relative amplitude value is shown in a table below it.

8.1.1. Loudspeaker model

It is not possible to simulate a continuous system model in Modelsim so the model had to be discretized. The derivation of the discrete transfer function of the loudspeaker model is given in chapter 5, and is represented by equation 8.2. The direct form-I implementation of the loudspeaker model is shown in Fig. 8.2.

$$H(z) = \frac{1.853 - 3.701z^{-1} + 1.848z^{-2}}{1 - 1.993z^{-1} + 0.9929z^{-2}} \tag{8.2}$$

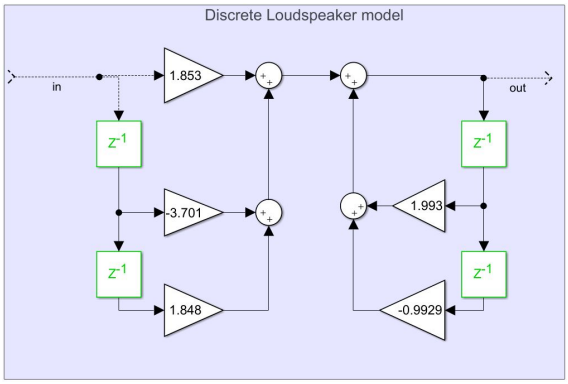


Figure 8.2: Direct form-I realization of discretized loudspeaker model

Simulink Test

The Direct Form-I implementation of the discrete loudspeaker model has been simulated in Simulink. For a frequency sweep from 10 Hz to 150 Hz at the input, the output is shown in Fig. 8.3. The amplitude of the input signal is 1 V. It is confirmed that the DF-I implementation of the loudspeaker model (Fig. 8.3) has the same response as the continuous model (Fig. 6.8).

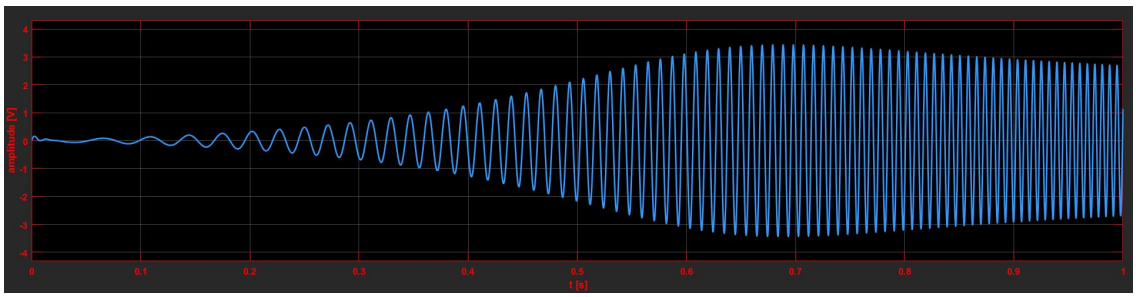


Figure 8.3: Output of DF-I implementation of discrete plant for frequency sweep from 10Hz at t=0 to 150Hz at t=1

Modelsim Test
The Direct Form-I has also been described in the VHDL language and simulated with Modelsim. A 100 Hz sine wave is shown in Fig. 8.4 for the plant. Some other amplitude values are shown in table 8.1.

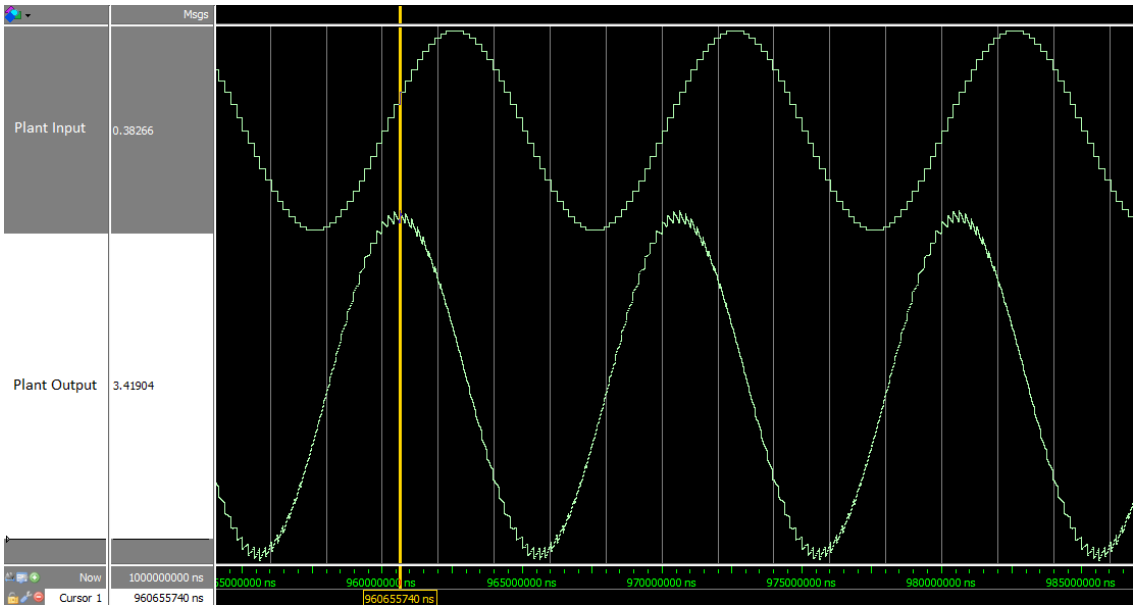


Figure 8.4: Input and output of the loudspeaker in Modelsim at 100Hz

Table 8.1: Modelsim simulation result for the implemented loudspeaker at different frequencies

Frequency (Hz)	10	20	50	80	100	120	200	300
Output / Input	0.036	0.083	0.659	2.274	3.419	3.274	2.310	2.037

8.1.2. Equalizer Filter

Similar to the discrete loudspeaker model, the equalizer filter has also been implemented in Direct Form-I. The transfer function of the equalizer filter is given by equation 8.3. The DF-I implementation is shown in Fig. 8.5.

$$H(z) = \frac{0.5397 - 1.075z^{-1} + 0.5358z^{-2}}{1 - 1.998z^{-1} + 0.9976z^{-2}} \tag{8.3}$$

Simulink Test
The DF-I implementation of the equalizer has also been tested in Simulink. The input of the equalizer is again a frequency sweep from 10Hz to 150Hz. As expected, the equalizer amplifies the lower frequency signals and attenuates the higher frequencies, since it is the inverse of the loudspeaker model. This can be seen Fig. 8.6.

Modelsim Test
Similar to the plant, A 100Hz sine is shown in Fig. 8.7 for the equalizer. Some other amplitude values are shown in table 8.2.

8.1.3. Controller

In Simulink, the standard discrete PI controller block is implemented using the backward Euler discretization method. The difference equation of the PI controller is given by equation 8.4, where $u(k)$ is the

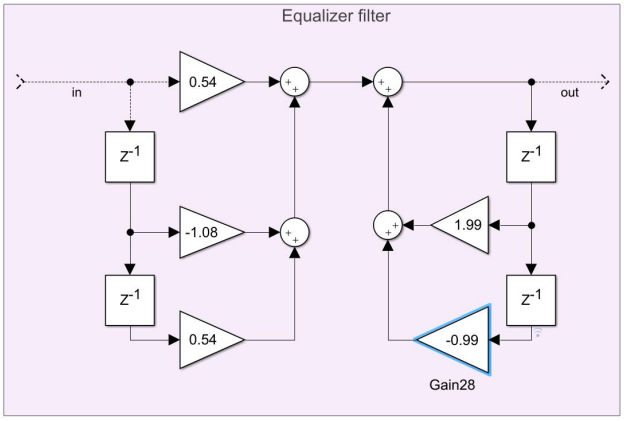


Figure 8.5: Direct form-I realization of equalizer filter

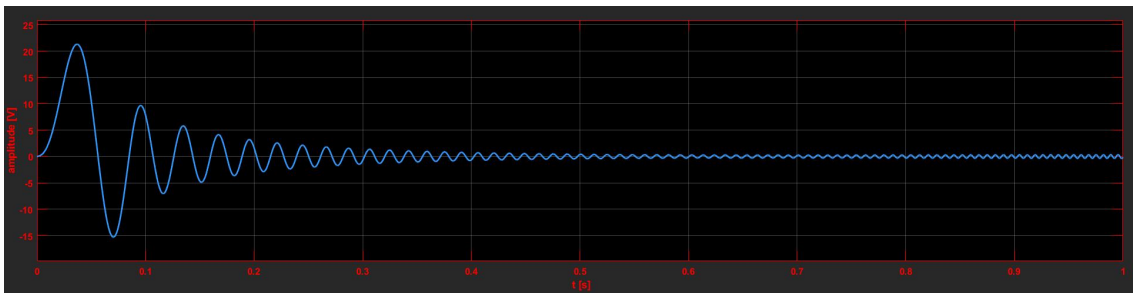


Figure 8.6: Output of DF-I implementation of equalizer filter for frequency sweep from 10Hz at t=0 to 150Hz at t=1

output of the controller, $e(k)$ is the error, K_p and K_i are controller parameters and T_s is the sampling period. This difference equation has also been realized in Direct Form-I and the structure is shown in Fig. 8.8.

From initial tests it was seen that the discretized controller did not show similar results as the continuous controller (i.e. it was unstable). It is assumed that the accumulator was the cause for this disparity. This has been solved with increasing the frequency of the accumulator 10 times, relative to the sample frequency. With this higher frequency the accumulator will represent an integrator more accurately. The 10 times increase was determined experimentally. Above this increase, no significant improvement was seen.

$$u(k) = K_p e(k) + K_i T_s \frac{1}{1 - z^{-1}} e(k) \tag{8.4}$$

Simulink Test
In order to test the working of the controller properly, it would have to be used along with the plant. However, to compare the Simulink model with the Modelsim implementation, it also had to be tested separately. In Fig. 8.9, the output of the controller is shown for a frequency sweep input from 10Hz to 150Hz. The amplitude of the input signal is 1V. As expected from the frequency response of the PI controller in Fig. 6.2, the low frequency signals are amplified. This is because for the low frequency signals, the integrator has more time to accumulate the input signal value before the input signal changes polarity.

Modelsim Test
A 100Hz sine wave is shown in Fig. 8.10 for a controller. Some other amplitude values are shown in table 8.3.

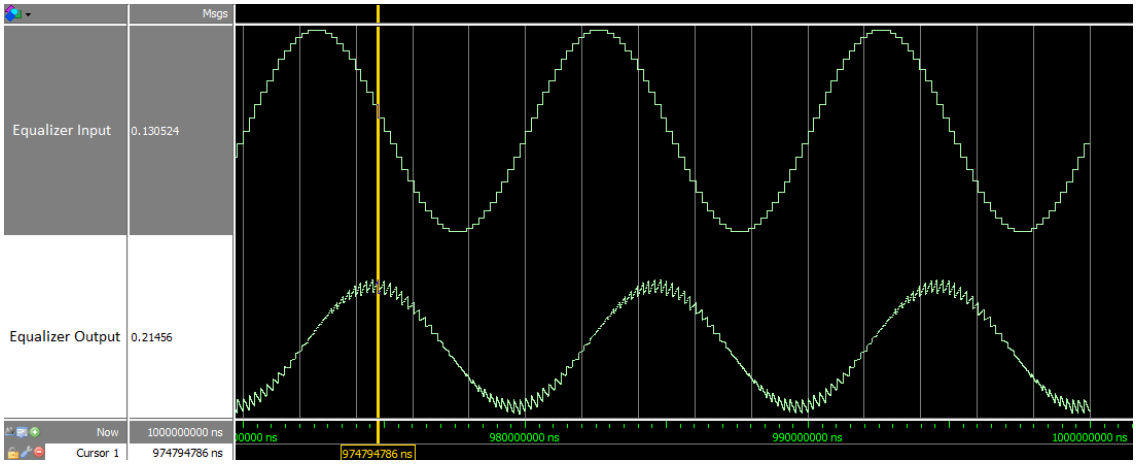


Figure 8.7: Input and output of the equalizer in Modelsim at 100Hz

Table 8.2: Modelsim simulation result for the implemented Equalizer at different frequencies

Frequency (Hz)	10	20	50	80	100	120	200	300
Output / Input	24.966	11.271	1.536	0.415	0.215	0.240	0.409	0.430

8.2. Complete Simulink design

All the components that were implemented in Direct Form-I are combined and tested.

8.3. Design Simulation matching

8.3.1. Simulink Simulation

The whole Simulink design, as shown in Fig. 8.11, has been simulated in Simulink and compared to the Modelsim simulation. The input signal is a frequency sweep from 10Hz at t=0 to 150 Hz at t=1 with an amplitude of 1 V. The output of the system is shown in Fig. 8.13. The output signal matches the input signal precisely, which is why only the output signal can be seen in the figure. The Simulink design which consists of the ADCs, DAC and continuous loudspeaker model, shown in Fig. 8.12, has also been simulated in Simulink. This also yielded the same results as shown in Fig. 8.13.

8.3.2. Modelsim Simulation

In Fig. 8.14 it is seen that the output follows the input with around 1% error while lagging slightly at 100 Hz. Other amplitude values are shown in table 8.4. The phase is not shifted significantly for all frequencies and thus is not shown.

Note that the output is smoother than the input. The cause for this is that the DAC runs at a 10 times higher frequency. This is possible because the controller and equalizer components already run 10 times faster because the accumulator has to approximate an integrator better, as explained before in this chapter. Moreover, the DAC its maximum sample frequency is 1 MHz compared to 4.8 kHz for the ADC. The output can thus be interpreted as a linear approximation of the DAC output values compared to if it would run at the same speed as the ADC.

8.3.3. Simulink versus Modelsim

In preliminary tests it was seen that simulating the components (i.e. controller, loudspeaker, equalizer) with Modelsim did not achieve the expected design results similar to Simulink. It was deduced that this originated from the fact that the accumulator did not approximate an integrator well enough. To improve the results, all the delays, represented with z^{-1} in Fig. 8.11, use a faster clock signal relative to the sampling frequency, which the ADC uses. It was experimentally concluded that the internal clock speed should be around 10 times higher for the best results.

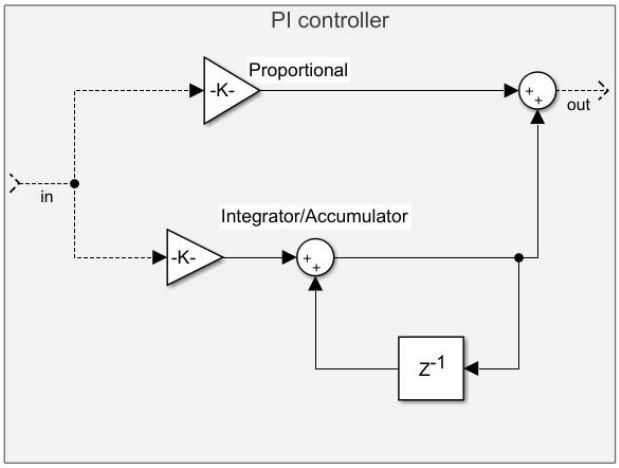


Figure 8.8: Realization of PI controller

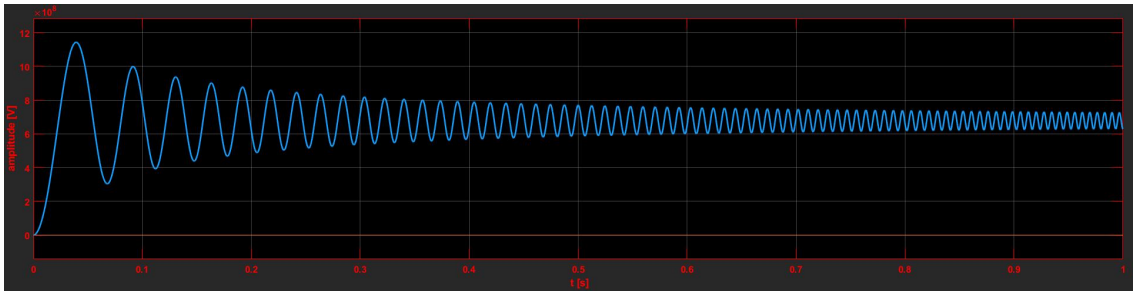


Figure 8.9: Output of PI controller for frequency sweep from 10Hz at t=0 to 150Hz at t=1

Some deviations (i.e. a small saw tooth at every sample point) from an expected sine wave can be seen in the Modelsim output for the loudspeaker and equalizer components. This is caused by the accumulator which runs at a frequency 10 times higher than the sample frequency and thus shows intermediate results. A buffer at the output of those components would produce smoother figures. The current figures however, show a more accurate representation regarding their behaviour.

In the Modelsim simulation figures, the input signal always has the same amplitude scale. The output signal however, is sometimes re-scaled in amplitude to be able to contain it in one figure.

A disparity can be seen between the Simulink and Modelsim simulation for the controller. The cause of this disparity originates from the way the frequency sweep is performed. The frequency sweep changes its frequency too quickly thus the controller's accumulator can not settle completely. This has been verified using a single frequency sine wave in the Simulink model, this removed the disparity between the models. Therefore, it is assumed that there is no real disparity between the controller implementations.

Both Simulink and Modelsim simulations give similar results regarding amplitude and phase shift. It should be noted however, that at low frequencies (i.e. 10Hz) the loudspeaker input can be around 34 times higher compared to the controller input. This can be to high depending on the absolute input value of the controller and the limit of the loudspeaker input.

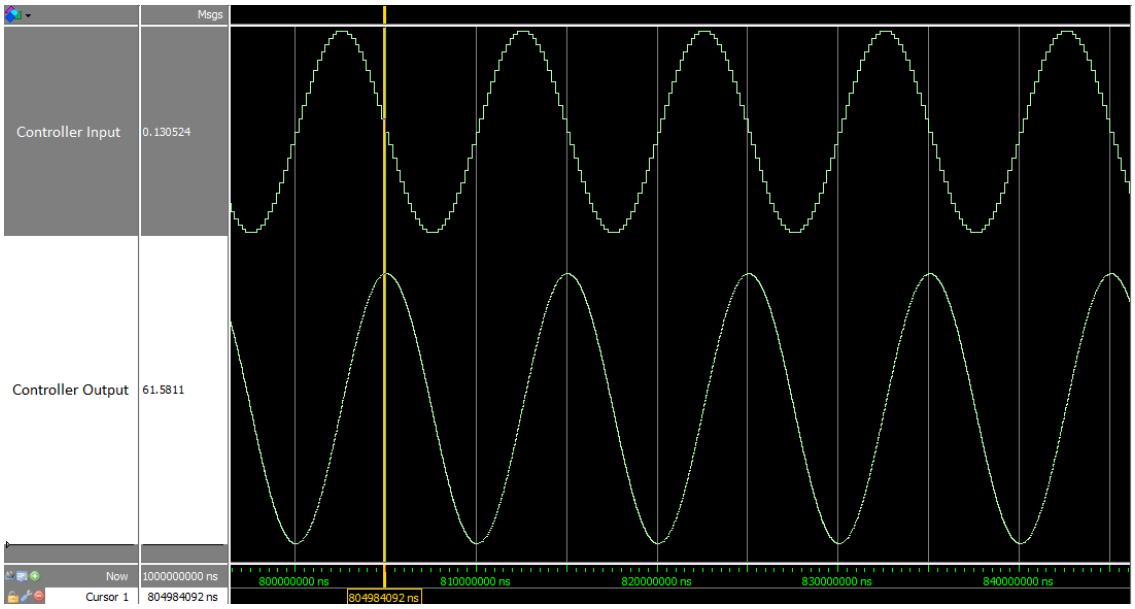


Figure 8.10: Input and output of the controller in Modelsim at 100Hz

Table 8.3: Modelsim simulation result for the implemented controller at different frequencies

Frequency (Hz)	10	20	50	80	100	120	200	300
Output / Input	619.414	309.359	123.84	76.806	61.581	51.865	31.144	20.380

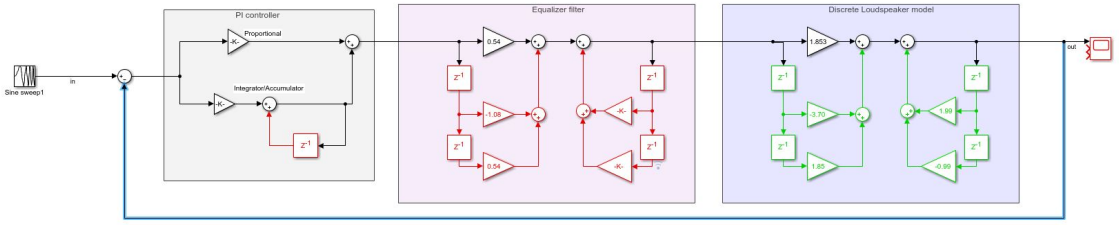


Figure 8.11: Block diagram representation of the whole system as implemented on the FPGA for simulation

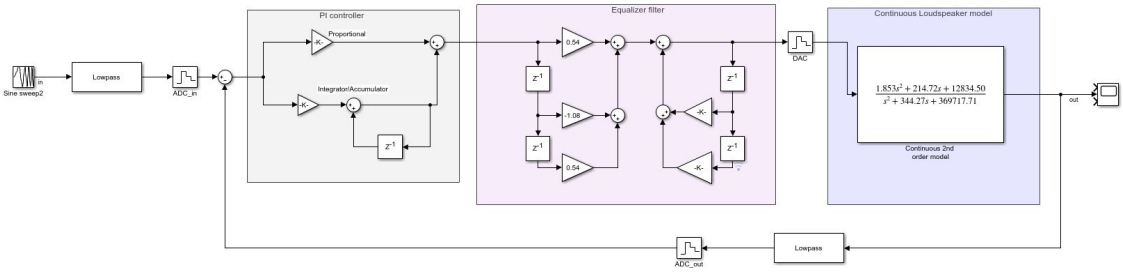


Figure 8.12: Block diagram representation of the whole system including the controller, equalizer, continuous Loudspeaker, ADCs, DAC and anti-aliasing filters

Table 8.4: Modelsim simulation result for the complete system at different frequencies

Frequency (Hz)	10	20	50	80	100	120	200	300
Output / Input	0.999	1.000	1.001	1.000	0.997	0.992	0.975	0.944

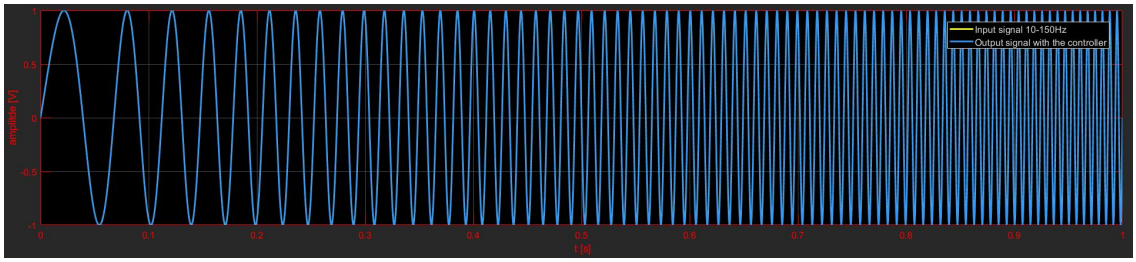


Figure 8.13: Output of whole system including the controller, equalizer and loudspeaker model for frequency sweep from 10Hz at t=0 to 150Hz at t=1

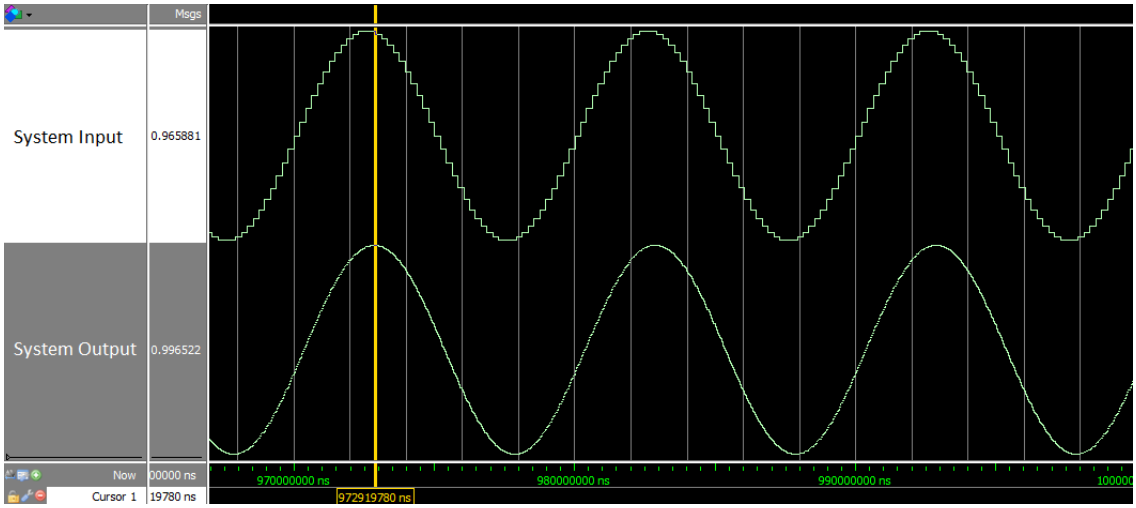


Figure 8.14: Input and output of the complete system in Modelsim at 100Hz

9

Implementation

The device on which the system will be implemented is a Field Programmable Gate Array (FPGA), as elaborated in chapter 7. The particular FPGA with peripherals was borrowed from professor Rene van Leuken, who is a staff member of the Circuits and Systems group at TU Delft. To program this device, a hardware description language (HDL) like VHDL or Verilog can be used. From multiple sources (e.g. [13]) it is seen that both languages can, and sometimes have to, be used. Because it appears that no major advantage is present from either and both are suitable, it is chosen to use VHDL because of the authors past-experience with it.

To simulate, synthesize and generate a bit stream for the FPGA, the Vivado Design Suite is used.

9.1. VHDL

With an HDL language like VHDL, hardware components can be defined and connected. VHDL enables the programmer to implement components with structural and behavioural architectures. The structural architecture is used for connecting components together, the behavioural architecture is used to describe the function in an algorithmic way. Standard structural and behavioural implementations are supported for synthesizing. The interested reader is encouraged to read the VHDL book written by Brown and Vranesic [4].

9.2. System Schematics

After the complete design is successfully tested. A complete system, as shown in Fig. 9.1, can be synthesized. Do note that there are SPI modules included. These modules are required to communicate, for every sample point, with the peripherals and also include delay buffers to keep the complete system synchronized.

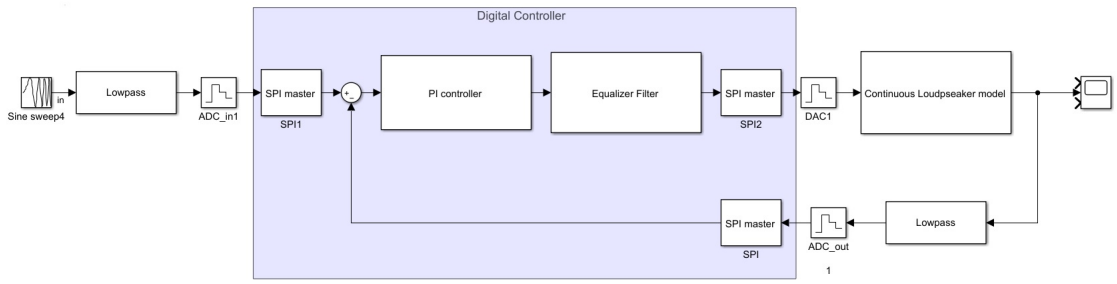


Figure 9.1: VHDL Schematics Overview

9.3. Peripherals

In order to process the analogue signals, they are first converted into digital signals. This is done with an ADC. To control the plant, the digital signal is later on converted to analogue with a DAC. The used FPGA does have an ADC component on-board but does not have a DAC [26]. Therefore, an external DAC is required as a peripheral. The chosen DAC requires the use of the SPI protocol for communication, which is highlighted below in section 9.4. The on-board ADC communication has to be done trough an Intellectual Property (IP) core, which is a pre-defined hardware component from the FPGA manufacturer. However, in order to save development time and the limited number of bits of resolution, it was decided to not learn how the IP core works but to also use the SPI protocol for the ADC. Therefore, an external ADC is also connected as peripheral. As explained before in chapter 7, the choice is made to use a 24 bit 4.8 kHz ADC.

The input range of the ADC is 0 V to 5.25 V, which means negative input voltages are not supported. An audio signal, which can also be negative, thus needs a DC offset to ensure that no negative voltages are present at the input of the ADC. The internal reference voltage of the ADC is by default 3.3 V. The input range thus becomes 0 V to 3.3 V. A DC offset of 1.65 V is added at the input of ADC to shift the equilibrium point from 0V to 1.65 V, which is in the middle of the input range of the ADC. Similar as with the ADC, the DAC only supports positive voltages at the output. The default range of the DAC is from 0 V to 2.5 V. Since the output of the DAC is only positive, a negative DC offset has to be introduced at the output. The offsets before the ADC and after the DAC are added using analog circuits.

9.4. SPI module

The standard Serial Peripheral Interface (SPI) protocol is a fast communication protocol relying in total on four parallel wires as seen in Fig. 9.2.

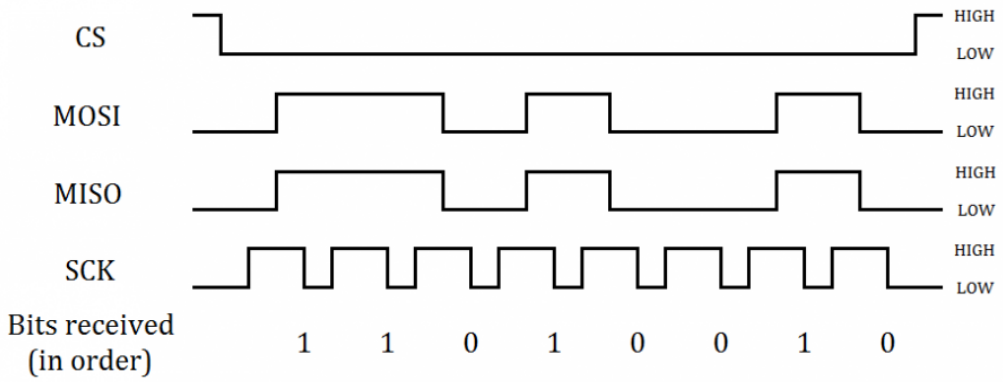


Figure 9.2: SPI Signals example[10]

The CS line should be brought low when transferring data. Then, the Master Out Slave In (MOSI) and Master In Slave Out (MISO) line both have to be used in parallel while communicating. At an event (i.e. rising edge) of the clock (SCK), data will be put on the MOSI and MISO lines. Different settings are available for reading and writing data to the MOSI and MISO lines with respect to SCK.

The particular ADC and DAC modules from Digilent however, use a slightly adapted version of the SPI protocol. As seen from [10], both the MOSI and MISO lines are used as MISO lines, one for each audio channel present on the ADC. A 24 bit signal can be read from the ADC as follows.
First CS is brought to low by the master (FPGA). After this, at every falling edge of SCK, the slave (ADC) will put a bit on the MISO line which should be read at the rising edge by the master. After 24 clock cycles all the bits are transferred and the CS line is brought to high again by the master. This is a signal to the slave that the transfer is done, so the slave can process new information for the next transfer.

The protocol is implemented in SPI components located between the ADC or DAC and the synchronized

buffers. The general state machine diagram of the ADC SPI component is shown in Fig. 9.3.

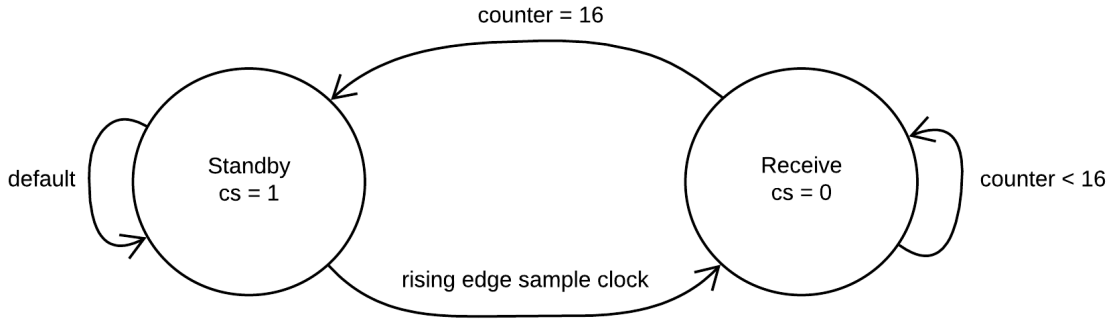


Figure 9.3: Simplified FSM diagram illustrating the SPI component logic for the ADC

It is seen that there are two states in total. In the standby state, CS is kept high and no actions are performed. When in the receive state, the CS line is kept low and at every rising edge of the clock a counter variable is incremented and the received bit is shifted in a temporary register. When the state is changed from receive to standby, the temporary register is copied into an output buffer of the SPI component and the counter is reset.

9.5. Format

The digitized values originating from the input and feedback lines will be processed using adders, multiplications and delay buffers. In order to accurately keep track of the intermediate results, as to not decrease the final accuracy, it is required to use a large enough number of bits internally. To be able to represent fractions it is moreover chosen to use fixed-point notation for the representation. An analysis can be done to calculate the required number of bits.
The ADC provides 24 bits of resolution. But note that at the final step the answer has to be truncated to 16 bits in order to fit the DAC. The FPGA has dedicated DSP slices for multiplication for 32 and 64 bit values. An optimal balance is considered to use 32 bits internally.

9.6. Timing

To correctly control the different delay buffers and SPI modules, a timing scheme as shown in Fig. 9.4 has been created to get a better overview.

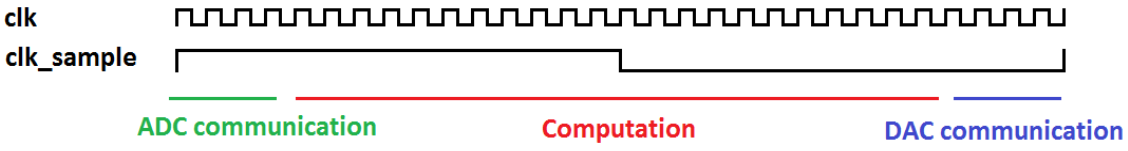


Figure 9.4: Timing scheme for the complete system

Note that Fig. 9.4 only provides an indication. Currently, the exact timing used for one sample point is the following.

- In one clk_sample there are 100 clk signals.
- At the start of the clk_sample, 24 clk signals are used to read a sample point from the ADC.
- After this, there are 60 clk signals available for computation.
- Finally, the last 16 clk signals are used to communicate with the DAC to output the sample point.

This will ensure that one sample only requires one clk_sample signal for the digital part. The controller can now be optimized in multiple small steps to a particular frequency to achieve the requirements as stated

in section 3.0.1. Note that the timing stated here is a basic guideline for the first tests. It is clear how many clk signals the ADC and DAC require. But it is not yet exactly clear how much time the computational part will require. When necessary, this should be experimentally researched. For now, the stated timing is considered useful for preliminary tests due that the timing should not be the cause of any problems.

Taking the maximum sample frequency of the ADC, 4.8kHz, it is seen that one clk_sample requires 208.3 µs. Therefore, from equation 9.1, it is clear that 125 µs are available for the computation part.

$$\text{computation time} = \text{clk\_sample} - \text{ADC communication} - \text{DAC communication} \tag{9.1}$$

The variables stated in equation 9.1 are in units of time.

It can be noted from the timing scheme that there is no concurrency between the different components, i.e. ADC communication, computation (controller and equalizer) and DAC. From the book [7], it is seen that it is technically possible to pipeline the components. This will increase the throughput but will also increase the delay for one sample. An example which illustrates this is depicted in Fig. 9.5. The colors used for the different components are similar as used in Fig. 9.4.

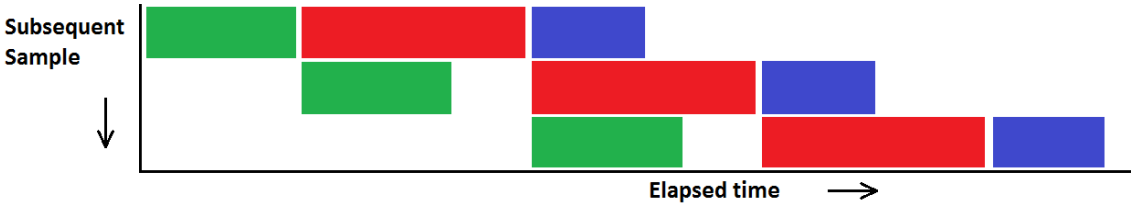


Figure 9.5: Pipelining illustration

It is seen that the first sample, top row in Fig. 9.5, does not have any gaps, white spaces, between components. This would be the case when there is no pipelining or each component has the same delay. If however, the individual component delays are not equal there will be an extra delay seen as a gap in Fig. 9.5 at the second and last row. Moreover, pipelining will produce overhead anyway because of the required synchronization buffers between the components. From this, it is concluded that the total sample throughput can be increased but the delay for each sample is also increased.

Because the sample frequency of the ADC is limited to 4.8kHz it is assumed that pipelining is not required because the FPGA is fast enough. Moreover, the delay and complexity of the design will be kept lower without pipelining. Thus no pipelining has been implemented.

10

Testing methods

The entire system is designed with Matlab (Simulink) and tested with Modelsim (written in VHDL) in chapter 8. In chapter 9, it is shown how the components written in VHDL code can be implemented on the FPGA. In this chapter, it is elaborated how the designed system components and the peripherals can be tested.

10.1. Peripherals testing

Two periperhals, ADC and DAC, are used. First, they should both be connected separately to the FPGA and the correct working of the SPI module for communication should be tested. After which, the ADC and DAC can both be connected to the FPGA and linked internally to test them together. the DAC should produce the same signal that is fed into the ADC.

10.1.1. ADC

First of all, it should be checked if the output from the FPGA Pmod pins corresponds to the expected pattern according to the datasheet [8] of the ADC. After this, a variable DC power supply can be used to feed the ADC input with different voltage levels. The input voltage level is coded by the ADC into a straight binary word. The binary word is read out with the ADC SPI module in the FPGA. The binary word that is read can be displayed with the FPGA using the on-board leds. Using equation 10.1 the binary word can be converted into a decimal value and compared to the applied input voltage.

$$\text{input voltage} = \text{to decimal (received binary word)} * \frac{v_{ref}}{2^{24}} \tag{10.1}$$

10.1.2. DAC

For the DAC it should also be confirmed if the output from the FPGA Pmod pins corresponds to the expected patterns as stated in the datasheet [9] of the DAC. After this, different binary values should be inputted from the FPGA to the DAC SPI module, dispersed over the operating range of the DAC. It can be confirmed with a voltmeter if the DAC provides the correct output. Switches can be used to efficiently test different binary words. The switches on the FPGA board can be used to control different bits in a binary word. This binary word can be supplied to the SPI module, communicating with the DAC.

10.1.3. ADC and DAC combined

Testing both the ADC and DAC with an AC signal is also important. To efficiently execute this, the ADC and DAC should both be connected to the FPGA. In the FPGA, the binary values received from the ADC are transferred directly to the DAC. The AC signal for the input of the ADC can be produced using a function generator and the output of the DAC can be displayed on the oscilloscope.

10.2. Summing circuits

Because the ADCs and DAC need an offset, analog summing circuits are used to generate and remove this offset. These circuits can be tested using a function generator, DC power source and an oscilloscope. The function generator can be used to generate an AC signal and a DC source can be used to generate the offset voltage. The inputs of a summer circuit are connected to the function generator and DC source. The output signal should be AC with an offset equal to the DC source output.

10.3. Unity feedback testing

When the ADC and DAC combined are successfully tested. The system, with the loudspeaker, is ready to be tested using unity feedback. The reason why the controller and equalizer should not yet be added is to be able to find the improvement that adding these components gives over unity feedback. Moreover it will supply information about the system (i.e. stability) that can be critical when unexpected results are found later on.

10.4. Complete system testing

Now the complete system can be connected. If the performance is not as desired. It is to be decided to which specific design step should be returned, depending on the results from unity feedback testing.

11

Measurements and Results

Using the testing methods as described in chapter 10, measurements are obtained from several components. Not all testing steps have been successfully completed due to the limited amount of time available.

11.1. Peripherals testing

After every change in code, the output of the Pmod pins to be used are checked to prevent damage to the peripherals.

11.1.1. ADC

To test the ADC, DC voltages from 0 V to the reference voltage of 3.3 V were applied at the input. Since the DAC was not tested yet, the binary values received from the ADC using a SPI module were read out using the on-board leds of the FPGA. By using equation 10.1, it was confirmed that the values displayed by the leds correspond to the input voltage.

11.1.2. DAC

A test using the switches on the FPGA board showed that the DAC produced the correct output as defined by the binary value. After this, a 16 bit binary value was used as a counter and at every rising edge of the clock incremented. This counter value was supplied to the DAC resulting in the saw tooth seen in Fig. 11.1.

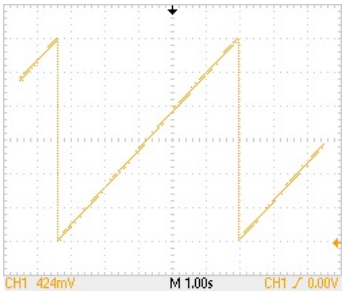


Figure 11.1: DAC output using a counter

11.1.3. ADC and DAC combined

A preliminary test with both peripherals combined resulted in Fig. 11.2a. The yellow line is the input to the ADC from a function generator and the blue line is the output from the DAC as seen with an oscilloscope. After changing the DAC peripheral to another Pmod connector, the result is as shown in Fig. 11.2b. The blue line is the input to the ADC from a function generator and the yellow line is the output from the DAC

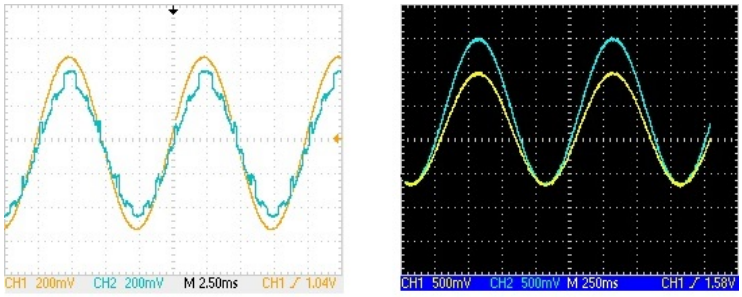


Figure 11.2: **a.** ADC and DAC combined with a bad output. **b.** ADC and DAC combined with a correct output

as seen with an oscilloscope.

11.1.4. Equalizer and PI controller on the FPGA

After successful tests with the ADC and DAC, the equalizer and PI controller on the FPGA were tested for various frequencies, see Fig. 11.3 and 11.4. In both figures the yellow line represents the input and the blue line represents the output signal. Note that the offset is not relevant, it is only present to increase readability.

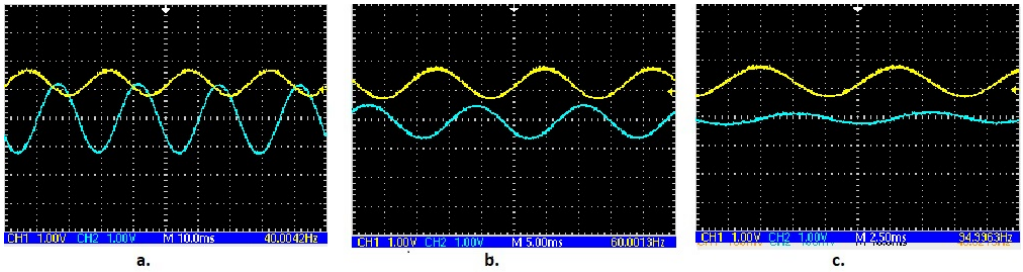


Figure 11.3: Output of the equalizer for various input signal frequencies: **a.** 40 Hz **b.** 60 Hz **c.** 95 Hz

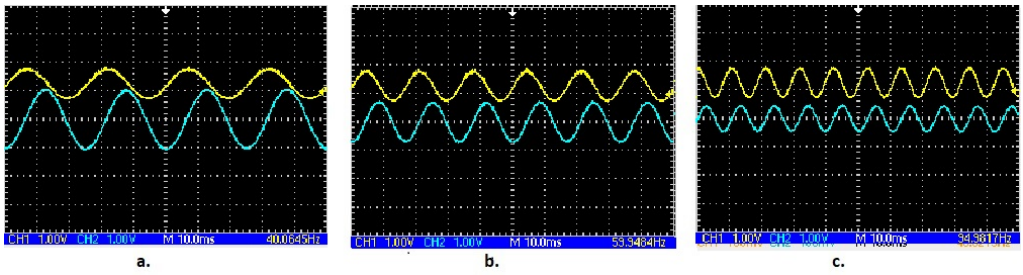


Figure 11.4: Output of the PI controller for various input signal frequencies: **a.** 40 Hz **b.** 60 Hz **c.** 95 Hz

11.1.5. Complete system

After the equalizer and PI controller were tested seperately, the complete system was tested. In Fig. 11.5, 11.6 and 11.7, the output of the loudspeaker measured with the accelerometer is shown for frequencies 40 Hz, 60 Hz and 95 Hz input signals. In the figures, the yellow line represents the input and the blue line represents the output signal.

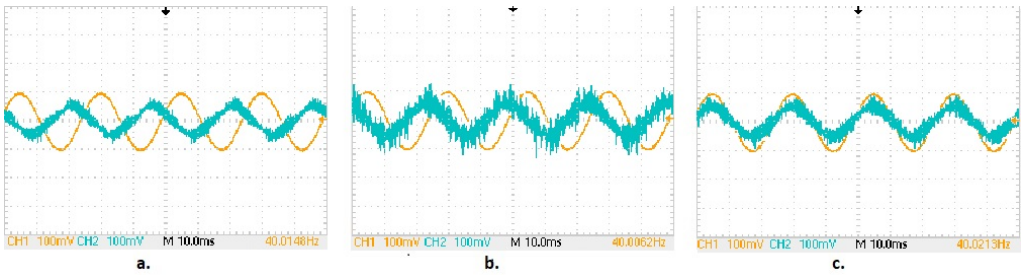


Figure 11.5: Output of the loudspeaker for 40 Hz input signal: **a.** Direct path through the FPGA **b.** Negative feedback only **c.** Negative feedback with equalizer

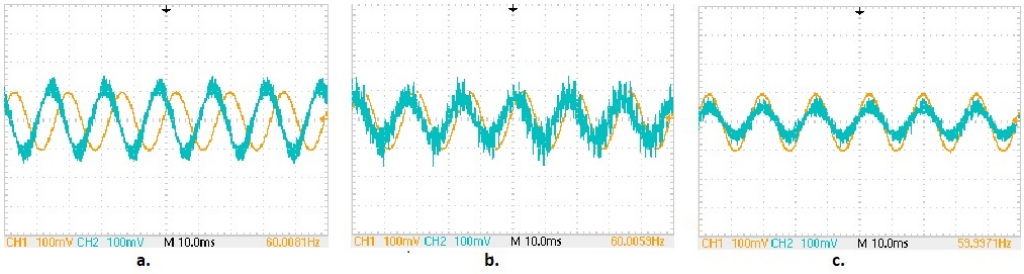


Figure 11.6: Output of the loudspeaker for 60 Hz input signal: **a.** Direct path through the FPGA **b.** Negative feedback only **c.** Negative feedback with equalizer

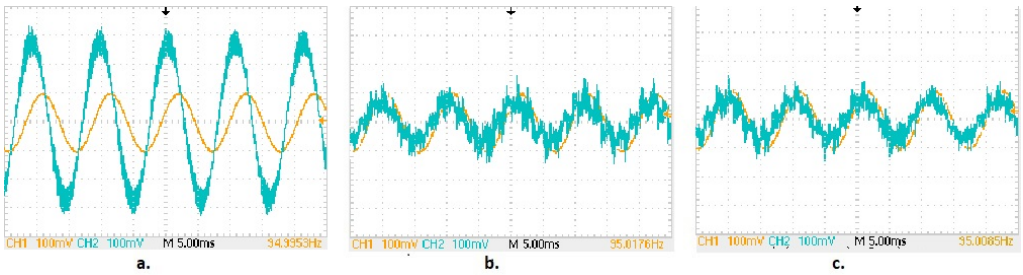


Figure 11.7: Output of the loudspeaker for 95 Hz input signal: **a.** Direct path through the FPGA **b.** Negative feedback only **c.** Negative feedback with equalizer

12

Discussion

12.1. Measurements

12.1.1. ADC and DAC separate

Both ADC and DAC produced the expected results. Therefore, it is concluded that both the ADC and DAC communication has been correctly implemented.

12.1.2. ADC and DAC combined

It should be noted that in the FPGA the most significant bits are transferred from the ADC to the DAC. While the ADC has an input range of 0 to 3.3 volts and the DAC has an output range from 0 to 2.5 volts, in Fig. 11.2, it can be seen that the output (blue) is approximately $\frac{3}{4}$ compared to the input (yellow), as expected.

More importantly however, a significant distortion was sometimes seen at the output signal. At higher voltages, which are not shown here, it was also seen that the signal sometimes, within one period, temporarily acquired an offset and suddenly reverted back. It is suspected that the random offset was caused by a certain bit flipping its value due to an unknown reason.

However, because both the ADC and DAC were successfully tested individually the inconsistency was assumed to originate from unexpected behaviour regarding the FPGA. After some consultations with a supervisor it was concluded that the clock dividers, used to control the components according to the timing scheme, were a major concern. After removing the clock dividers and implementing the timing scheme in another way, the correct output as seen in Fig. 11.2b was consistently shown.

12.1.3. Summing circuits

The summing circuits work as expected. Both positive and negative DC offset can be added to an AC signal. However, the circuit introduces a lot of noise at the output. The noise was suppressed using a capacitor at the output of the summing circuit, i.e. a first order low-pass filter was added.

12.1.4. Equalizer and PI controller on the FPGA

The controller and equalizer were tested in an open loop path without the loudspeaker. The results are shown in Fig. 11.3 and Fig. 11.4. From these results it is clear that both the controller and equalizer display their expected behaviour.

To elaborate, when taking into account for example the magnitude, it is seen that the PI controller its magnitude is inversely related to the frequency. Moreover, the equalizer has the inverse frequency behaviour compared to the loudspeaker.

12.1.5. Unity feedback testing

Connecting the complete system in a direct path, i.e. directly supplying the FPGA input to the output without doing anything, introduced a lot of noise. When using the unity feedback configuration it was seen that the magnitude and phase of the desired signal were improved significantly but the noise also increased somewhat. Some results of direct path and unity feedback are shown at 40, 60 and 95 Hz as shown in Fig. 11.5, Fig. 11.6 and Fig. 11.7 respectively. It is assumed that the noise is present because of insufficient shielding of the components.

12.1.6. Complete system

Both controller and equalizer were tested with the loudspeaker and negative feedback. Some results of the equalizer are shown at 40, 60 and 95 Hz as shown in Fig. 11.5c, Fig. 11.6c and Fig. 11.7c respectively. When enabling the equalizer, it was seen that the desired signal is improved some more compared to unity feedback and the noise is also reduced a little. A lot of noise can still be seen but it is assumed that the noise does not originate from the equalizer. Therefore, the equalizer is successfully tested.
The controller however, is not shown in the measurements because it was unstable.

12.2. Observations

12.2.1. Environmental influences

A large (i.e. 100mV) 50Hz signal, originating from the electric wall socket, can be visible on the FPGA connector pins. The signal strength depends on the proximity of external material e.g. a hand. When signals are relative small, the interference could become significant and measures should be taken to shield the necessary components.

12.2.2. FPGA

From the reference manual [26] the following is clear.
When the FPGA is idling, the board uses ~1W of power. The VCC and Ground pins can deliver up to 1A of current, but care must be taken not to exceed any of the power budgets of the on-board regulators or the external power supply.

According to the requirements seen in chapter 3.0.1, the controller should use no more than 0.1 W. This is not achievable with the current FPGA but implementing the final design in ASIC will reduce the operating power.
The FPGA cannot provide the necessary power for the loudspeaker. Therefore, a power amplifier is required behind the DAC output.
Similar as with the operating power, an FPGA is too expensive to be used for large scale manufacturing. Implementing the system in ASIC will reduce the costs in large quantities.

13

Conclusion

A loudspeaker is a non-ideal device with physical properties and limitations. One of the consequences of this is that the reproduction of audio with a loudspeaker is not perfect. The loudspeaker manifests deformation of input audio signals in the form of linear and non-linear distortions. The linear distortion can be visualized with a bode plot. From the bode plot it can be seen that there is a deviation from the ideal case, 0 dB gain and zero phase shift at every frequency in the bandwidth. This deviation can be compensated using a controller. Different topologies can be used to implement a controller but only one is chosen. The chosen digital controller topology is negative feedback with a PI controller and an equalizer in the open loop path. To design the controller, a linear model has been derived from a loudspeaker measurement. Using this derived model, parameters of the controller are determined. To confirm the parameters and topology choice, simulations were performed using the controller and loudspeaker models. The simulations were carried out in Matlab (Simulink) successfully. To use the controller, hardware is required. The chosen hardware is an FPGA with ADCs and a DAC as peripherals. The hardware implementation was successfully tested with ModelSim. The simulations show that the linear distortion of the loudspeaker is completely suppressed using the controller.

The tests in hardware have only been partially successfully performed. The peripherals, needed to combine the equalizer, controller and the loudspeaker, have been successfully tested. Moreover, the controller and equalizer were successfully tested using only the peripherals. Also, the equalizer was tested with the physical loudspeaker and showed in general results as expected but there was a lot of noise. The controller however, is unstable when used with the physical loudspeaker.

From the simulations and the partial measurements it is considered to be feasible to suppress the linear and non-linear distortion of a loudspeaker with a digital controller. Because the controller was not successfully tested with the loudspeaker and the equalizer has a lot of noise it is not possible to determine if all the requirements can been met. From the mandatory requirements it is however assumed that the power consumption, delay smaller than 120 ms, cost of the system no more than €100, volume no more than 0.5l and operation range from 10-300Hz can be achieved. Note that to achieve most of the requirements it is required to not use FPGAs but produce ASICs.

To conclude, the use of a digital controller to improve the performance of a loudspeaker is considered to be a choice which should not be overlooked.

A

Appendices

A.1. Project Deliverables

Table A.1: List of deliverables

Date	Deliverables
01-05-2018	Literature study report
29-05-2018	Green-Light Assessment
18-06-2018	Thesis report
02-07-2018	Oral presentation and thesis defense
06-07-2018	Demonstration of the prototype

A.2. System identification in Matlab

```
openfig('amplitude4_vol85_gitaar.fig'); % Opening and extracting the
h = findobj(gca,'Type','line');          % measured magnitude response
x_m=get(h,'Xdata');                      % of the loudspeaker
y_m=get(h,'Ydata');
y_m = db2mag(y_m);
x1 = find(x_m>10 & x_m<10.1);  %Setting the frequency range for the TF
x2 = find(x_m>300 & x_m<300.1);% estimation
x1=x1(1);
x2=x2(1);
x = x_m(x1:x2);
close;
openfig('phase4.fig'); % Opening and extracting the measured phase
h = findobj(gca,'Type','line'); %response of the loudspeaker
x_p=get(h,'Xdata');
y_p=get(h,'Ydata');
y_p = degtorad(y_p);

y = y_m.*exp(i.*y_p); % Combining the magnitude and phase response
y = y(x1:x2);         % into one complex vector
h = idfrd(y,2*pi*x,1/48000); % Fs = 48000 Hz

tf_est_opt = tfestOptions;
tf_est_opt.EnforceStability = true;
```

```
H_9 = tfest(h,9,tf_est_opt) % Estimating the 9th order TF

s=tf('s');  % Constructing the second order TF using poles and zeros
H_2 =(s-[-57.9387908198506 + 59.7447417477858i])*
    (s-[-57.9387908198506 - 59.7447417477858i]) /
    ((s-[-172.135279747862 + 583.169922592034i])*
    (s-[-172.135279747862 - 583.169922592034i]))
H_2 = 1.853*H_2;            % Offset in the magnitude response
H_2_d =c2d(H_2,1/48000);    % Discretizing the second order TF
G_2 = inv(H_2)              % Equalizer filter
G_2_d = c2d(G_2,1/48000)    % Discretizing the equalizer filter
```

A.3. VHDL Code for controller implementation in FPGA

The VHDL code for the implementation of the motional feedback system on the FPGA is subject to be changed during the complete system implementation and tests. To view the provisional code, one is requested to visit the GitHub repository given by the following link.
https://github.com/bswsz/BAP
After successful implementation, the provisional code shall be updated with the final code.

Bibliography

[1] Khalid Mohammad Al-Ali. "Loudspeakers: Modeling and Control". PhD thesis. University of California at Berkeley, 1999.

[2] Anonymous. *Improving ADC resolution by oversampling and averaging*. Tech. rep. Silicon Labs.

[3] A. J. Bianchi and M. Queiroz. *Real time digital audio processing using Arduino*. Tech. rep. University of Sao Paulo, 2013.

[4] Stephen Brown and Zvonko Vranesic. *Fundamentals of Digital Logic with VHDL Design*. third. Raghothaman Srinivasan, 2009.

[5] Pascal Brunet. "Nonlinear System Modeling and Identification of Loudspeakers". PhD thesis. Northeastern University Boston, Massachusetts, Apr. 2014.

[6] C-Y Chen et al. "Passive voice coil feedback control of closed-box subwoofer systems". In: *Journal of Acoustical Society of America Proceedings of The Institution of Mechanical Engineers Part C-journal of Mechanical Engineering Science* 214.7 (July 2000), pp. 995–1005.

[7] John L. Hennessy David A. Patterson. *Computer Organization and Design*. fifth. Elsevier, 2014.

[8] Digilent. *Pmod AD5 Reference Manual*. URL: `https://reference.digilentinc.com/reference/pmod/pmodad5/reference-manual`.

[9] Digilent. *Pmod DA3 Reference Manual*. URL: `https://reference.digilentinc.com/reference/pmod/pmodda3/reference-manual`.

[10] Digilent. *SPI Reference Digilent*. URL: `https://reference.digilentinc.com/learn/fundamentals/communication-protocols/spi/start?redirect=1`.

[11] Janine Elliot. *Philips Motional Feedback Speakers*. Sept. 2016. URL: `http://hifipig.com/philips-motional-feedback-speakers/`.

[12] Iain Forgusson. *Loudspeaker cross section*. May 2010. URL: `http://en.wikipedia.org/wiki/File:Speaker-cross-section.svg`.

[13] Shannon Hilbert. *Verilog vs. VHDL*. Apr. 2013. URL: `http://www.bitweenie.com/listings/verilog-vs-vhdl/`.

[14] R. Hilmisson. "Feedback Linearisation Of Low Frequency Loudspeakers". MA thesis. Technical University of Denmark, Sept. 2009.

[15] Tijs Hol and W. Bons. *Digital Control in a Motional Feedback Audio System*. Tech. rep. Delft University of Technology, 2016.

[16] M. Jakobsson D. Larsson. "Modelling and Compensation of Nonlinear Loudspeakers," MA thesis. Chalmers University of Technology, 2010.

[17] Dr.ir. G.J.M. Janssen et al. *Lab Courses EE Semester 1, Student Manual*. TU Delft. 2017-2018.

[18] Yusuke Kadowaki and Toshiya Samejima. "Nonlinear distortion reduction of an electrodynamic loudspeaker by using model-following control theory". In: *The Acoustical Society of Japan* 38 (2017), pp. 222–224.

[19] J.A. Klaassen and S.H. de Koning. "Motional feedback with loudspeakers". In: *Phillips Technical Review* 29.5 (1968), pp. 148–157.

[20] Wolfgang Klippel. "Loudspeaker nonlinearities - causes, parameters, symptoms." In: *Journal of Audio Engineering Society* 54.10 (Oct. 2006), pp. 907–939.

[21] Wolfgang J. Klippel. "ADAPTIVE NONLINEAR CONTROL OF LOUDSPEAKER SYSTEMS". In: *Journal of the Audio Engineering Society. Audio Engineering Society* (1998).

[22] James V. sanders Lawrence E. Kinsler Austin R. Frey. *Fundementals of Acoustics*. fourth. John Wiley & Sons, Inc., 2000.

[23] Yaoyu Li and G. T. C. Chiu. "Control of loudspeakers using disturbance-observer-type velocity estimation". In: *IEEE/ASME Transactions on Mechatronics* 10.1 (Feb. 2005), pp. 111–117.

[24] R. Overwater and Y. Rosema. *Digital Implementation of a Motional Feedback Audio Sytem*. Tech. rep. Delft University of Technology, 2016.

[25] Franklin Powell and Emami-Naeni. *Feedback Control of Dynamic Systems*. seventh. Pearson, 2014.

[26] *Reference Manual Zynq 7000*. URL: https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf.

[27] Aart-Peter van der Schipper and Alexandros Skourtis-Cabrera. *Motional Feedback in a Bass Loudspeaker, Theory*. Tech. rep. TU Delft, 2018.

[28] Robert-H Munnig Schmidt. *Motional Feedback Theory in a Nutshell*. Tech. rep. RMS Acoustics & Mechatronics, 2017.

[29] C. Sean. "A Direct PWM Loudspeaker Feedback System". MA thesis. Massachusetts Institute of Technlogy, 1996.

[30] Julius O. Smith. *Introduction to Digital Filters with Audio Applications*. http://www.w3k.org/books/: W3K Publishing, 2007. ISBN: 978-0-9745607-1-7.

[31] Inc. The MathWorks. *Matlab Documentation - c2d*. URL: https://nl.mathworks.com/help/control/ref/c2d.html.

[32] Inc. The MathWorks. *Matlab Documentation - PID Tuner*. URL: https://nl.mathworks.com/help/control/ref/pidtuner-app.html.

[33] Inc. The MathWorks. *Matlab Documentation - tfest*. URL: https://nl.mathworks.com/help/ident/ref/tfest.html.

[34] Paolo La Torraca. "FEEDBACK CONTROL OF A DYNAMIC LOUDSPEAKER WITH EMBEDDED SENSOR COIL". MA thesis. Politecnico di Milano, 2015.

[35] R. Valk. "Control of Voicecoil Transducers". MA thesis. Delft University of Technology, 2013.