

UC Davis

UC Davis Electronic Theses and Dissertations

Title

Scalable Tactile Sensing Architectures Utilizing Frequency-Domain Multiplexing Techniques

Permalink

<https://escholarship.org/uc/item/2t93126b>

ISBN

9798189228520

Author

Perna, Shannon Marie

Publication Date

2026-06-25

Peer reviewed|Thesis/dissertation

Scalable Tactile Sensing Architectures Utilizing Frequency-Domain Multiplexing Techniques

By

SHANNON PERNA

Masters Thesis

Submitted in partial satisfaction of the requirements for the degree of

Master of Science

in

Electrical and Computer Engineering

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Iman Soltani, Chair

Soheil Ghiasi

Hyoyoung Jeong

Committee in Charge

2026

Contents

Abstract	iv
Acknowledgments	v
Chapter 1. Introduction	1
1.1. Motivation	1
1.2. Problem Statement	1
1.3. Previous Work	2
1.4. Research Objectives	5
Chapter 2. Background	7
2.1. Tactile Sensing Systems	7
2.2. Piezoresistive Sensing Materials	8
2.3. Frequency-Domain Multiplexing	8
2.4. Passive Filters and Quality Factor	9
2.5. Active Bandpass Filters	10
2.6. Varactors and Variable Capacitance	10
2.7. Nonlinear Optimization in Resonant System Design	10
Chapter 3. Passive RLC Bandpass Architecture - AM	12
3.1. General Solution	12
3.2. System Limitations	16
Chapter 4. Active Varactor-Based Bandpass Architecture - FM	17
4.1. General Solution	17
4.2. Nonlinear Solver	23
4.3. Simulated Circuit	27

4.4. Limitations	37
Chapter 5. FPGA-Based Multiplexed Architecture - AM	38
5.1. Solution 1: Sin Waves	38
5.2. Solution 2: Square Waves	40
5.3. Delaying Impulse Trains Code	41
5.4. Limitations	44
Chapter 6. Conclusion	45
Appendix A. Code Screenshots	47
Appendix. Bibliography	51

Abstract

This thesis investigates scalable tactile sensing architectures utilizing frequency-domain multiplexing techniques to reduce wiring complexity while maximizing achievable sensing density. Three sensing architectures are explored, including passive RLC resonators, active varactor-based multiple-feedback bandpass filters, and FPGA-based amplitude modulation systems.

The passive resonator architecture investigates amplitude-modulated tactile sensing using piezoresistive materials, with emphasis placed on resonator bandwidth limitations, achievable frequency spacing, and practical component constraints. The active architecture presents a varactor-based frequency-modulated sensing system, where a nonlinear optimization framework is developed to solve for component values while constraining resonant frequency, bandwidth, gain, quality factor, and channel overlap. The FPGA-based architecture explores shared carrier generation and signal-processing techniques for increasing sensing density using both sinusoidal and square-wave carrier methods.

Simulation results demonstrate the tradeoffs between bandwidth allocation, guard-band spacing, resonator overlap, physical component limitations, and achievable sensing density across each system. Architectures ranging from 9 successful simulated nodes to 49 theoretical nodes are discussed. This work establishes scalable design methodologies relevant to electronic skin systems, flexible tactile interfaces, and large-area frequency-multiplexed sensing platforms.

Acknowledgments

I would like to recognize those who have contributed to this project as mentors, supporters, and collaborators. First, I would like to express my gratitude to Dr. Soltani, who created the framework for a project that taught me so much about the intricacies of sensor systems. Thank you for providing me with the resources, mentorship, and confidence necessary to grow as an engineer over the past two years.

I would also like to thank Dr. Naddaf for his patience and commitment to technical mentorship throughout this project, and Dr. Banuasadi for his valuable contributions to its advancement. Additionally, I extend my thanks to Dr. Jeong and Dr. Ghiasi for their aid in reviewing this thesis and their dedication to supporting and connecting with students.

I'm deeply grateful to my family for encouraging me every day over the last few years to keep pushing forward and always put my best foot forward. Finally, I dedicate a special recognition to my late grandfather, Christopher Condon, who showed me the pathway to electrical engineering through his own inspiring career.

CHAPTER 1

Introduction

1.1. Motivation

High-density tactile sensing plays an important role in emerging applications in robotics, wearable electronics, prosthetic devices, biomedical systems, and human-machine interfaces. Electronic skin systems capable of spatially resolving pressure distributions have gained significant attention due to their potential to replicate aspects of human touch [3]. Flexible tactile sensors add another multidimensional level of precision as they can replicate human tactile perception like pressure, shear, bend and strain, where pressure from 10 -100 K Pa is suitable for object manipulation and sensors are able to deduce 3 Pa, similar to a fly landing on its surface [13]. These systems can provide robots with environmental awareness, improve prosthetic hand feedback, create interactive wearable devices, and enhance human-computer interaction.

While pressure density has proved to match up to human capabilities, sensor spatial density remains a complex issue. As tactile sensing arrays increase their total sensing area, conventional architectures become increasingly difficult to scale. A dense sensor grid may require hundreds to thousands of individual sensing nodes, each requiring dedicated space in their footprints for multiple components and complex wiring [1].

1.2. Problem Statement

Scaling up a tactile sensing array creates substantial challenges in routing complexity, power consumption, physical footprint size, and customizable grid geometry. These limitations become problematic in flexible sensing materials where excessive wiring and components reduces flexibility and manufacturability.

To address these challenges, scalable sensing architectures capable of reducing overall footprint and wiring complexity while maintaining high spatial resolution are required. One promising approach is frequency-domain multiplexing, where each sensing node operates within a unique frequency region while sharing the same external wiring input and communication pathway [1]. By converting spatial information into the frequency domain, the total number of input wires required by the system can remain constant even as additional sensing nodes are added, and individual resonator footprints can be optimized for sizing.

However, frequency-multiplexed tactile sensing systems introduce several new design challenges. As the number of sensing nodes increases, available spectral bandwidth must be efficiently allocated while avoiding channel overlap and maintaining sufficient sensitivity.

1.3. Previous Work

Tactile sensing systems are designed to detect and spatially resolve physical interactions including pressure, shear, texture, and vibration across flexible surfaces, with applications spanning robotic manipulation, prosthetic feedback, wearable health monitoring, and human-machine interaction [3]. As tactile arrays increase in area and node count, wiring complexity becomes the primary challenge. This challenge is abundant in robotic glove-based systems, where covering three dimensional hand geometry while maintaining flexibility restricts on the number of wires that can be routed across the substrate.

Several sensing and readout architectures have been explored for scalable tactile sensing systems, primarily either utilizing resistive sensing arrays or capacitive sensing arrays.

Piezoresistive sensing materials are among the most widely used transduction mechanisms in flexible tactile arrays due to their simple readout circuitry, large resistance variation under compression, and compatibility with flexible substrates [4]. When pressure is applied, conductive pathways within the material change, producing a measurable variation in resistance that can be read directly through a voltage divider circuit [4][11]. Flexible piezoresistive arrays have been demonstrated for robot

skin applications capable of spatially resolving contact force distributions across curved surfaces [10], and commercial piezoresistive systems have been deployed on robotic hands for grasping feedback [1]. However, piezoresistive materials including Velostat exhibit nonlinearity, hysteresis, and manufacturing variability that complicate calibration across large arrays, and resistance variation is sensitive to temperature drift and repeated loading fatigue [4].

Capacitive tactile sensing systems have been considered due to their low power consumption and compatibility with flexible materials[10]. However, solely capacitive systems are highly sensitive to parasitic capacitance, and nonlinearly, particularly in large-area flexible systems where distributed parasitics are difficult to model. [3]

Beyond sensing material selection, the readout architecture determines how signals are acquired over a limited number of wires. For example, electrical impedance tomography (EIT) tactile sensing solves the wiring problem much differently than multiplexing. Rather than assigning each sensing node a unique channel, EIT methods apply current through a small number of boundary electrodes on a piezoresistive sheet and reconstruct the pressure distribution computationally from the resulting voltages. Kato et al. demonstrated this in a soft tactile sensor made of pressure-sensitive conductive rubber, where the resistivity of the material increases under applied pressure and 16 electrodes placed around the perimeter were used to reconstruct the full scope of pressure distribution without any wires in the sensing region [15]. While this eliminates wiring within nodes entirely, the reconstruction relies on constrained numerical methods to produce usable results. Additionally, the update rate of this prototype was under 80 milliseconds per frame, limiting responsiveness for dynamic cases [15].

Multiplexing architectures are also used to minimize the complexity of wire routing. Spike-based multiplexing architectures encode tactile information as asynchronous event transmitted over a shared conductor, similar to biological peripheral nervous systems [16]. Rather than continuously sampling all nodes in sequence or assigning frequency channels, each sensing element generates a unique spike signature that encodes both its spatial information and the applied pressure, allowing all nodes to transmit simultaneously. Kim et al. demonstrated this in their position-encoded spike

spectrum (PESS) system, where nodes constructed from a mixed ion-electron conductor generate spikes with unique relaxation times, and the sequence of these spikes across the array encodes static and dynamic touch position in real time [16]. The system required only two global wires to operate about 132 nodes per square centimeter, with latency under 5 milliseconds. However, the approach encodes pressure as a binary data rather than a continuous analog value, and the number of distinguishable nodes is constrained by the relaxation time [16].

Row-column matrix architectures reduce wire count by sharing conductors between multiple sensors, reducing wire count from $M \times N$ to $M + N$. While proven effective for moderate array sizes, these systems primarily rely on Time-division multiplexed (TDM) readout architectures, which suffer from readout delay, electrical crosstalk, signal disturbance from continuous switching, and reduced acquisition speed due to sequential scanning operation [5].

Even hybrid TDM approaches, such as Zhenxing Wang and Xuan Dou’s approach have their limitations. Their system assigns different sampling rates to different channels through FPGA-based scheduling, facing the limitations of sequential scanning, offsets in channel timing, and acquisition speed [11].

Frequency-domain multiplexing (FDM) has emerged as a promising approach that avoids the sequential scanning bottleneck of TDM while offering scalable wire counts. Rather than addressing each node directly, FDM assigns each sensing element a unique frequency channel, allowing many nodes to share a common signal path [2]. Within FDM systems, there are two distinct modulation tactics: amplitude modulation (AM), where touch changes the amplitude of a carrier at a fixed resonant frequency, and frequency modulation (FM), where touch shifts the carrier frequency.

Long et al. demonstrated AM modulation in a one-wire tonotopy-inspired sensor matrix where each node generates a unique fixed-frequency sinusoidal signal and applied pressure modulates its amplitude through a piezoresistive voltage divider, with all node signals combined onto a single shared conductor, reducing wiring from $M \times N$ connections to a single output wire regardless of array size [2].

In contrast to AM approaches, FM methods encode pressure as a continuous shift in resonant frequency rather than a change in carrier amplitude. Xu et al. demonstrated this in a wireless RF-resonator-based tactile array where applied pressure directly shifts the LC resonant frequency [14]. However, this work relies on machine learning to resolve overlapping resonant peaks rather than defining explicit channel boundaries through spectral optimization, a gap directly addressed by the varactor-based architecture in Chapter 4.

While prior frequency-domain tactile sensing systems have demonstrated scalable low-wire-count architectures through spectral encoding, this work further investigates the physical and spectral constraints limiting achievable node density across three distinct architectures, passive RLC resonators, active varactor-based bandpass filters, and FPGA-based carrier generation, including resonator bandwidth allocation, guard-band spacing, quality factor scaling, component footprint limitations, and nonlinear optimization of resonant channels.

1.4. Research Objectives

This thesis investigates scalable tactile sensing architectures that utilize resonant frequency multiplexing and modulation techniques to maximize sensing density while minimizing wiring complexity and maintaining linear sensing behavior. The primary contributions of this work include:

- (1) Development of a passive RLC resonator-based tactile sensing architecture utilizing amplitude modulation through piezoresistive materials.
 - (a) Find the resistor and capacitor values for the smallest possible bandwidth
- (2) Development of an active varactor-based multiple feedback bandpass filter architecture capable of frequency modulation sensing
 - (a) Development of a nonlinear optimization framework for calculating component values while constraining channel overlap and maintaining desired system characteristics (resonant frequency, bandwidth, gain, and channel spacing)
 - (b) Defining scalability limitations related to physical component sizes, bandwidth allocation, and quality factor growth across large sensing arrays
- (3) Develop a FPGA-based amplitude modulation technique that will allow for more nodes to be added to the system

- (a) Compare sine wave carriers to square wave carriers
- (b) Investigate carrier offset customization so the system complies with the ADC
- (c) Create equations to represent the system's node density

CHAPTER 2

Background

2.1. Tactile Sensing Systems

Tactile sensing systems are designed to detect physical interaction with the environment, including force, pressure, texture, vibration, and deformation. These systems are commonly used in robotics, prosthetics, wearable electronics, biomedical monitoring, and human-machine interaction applications.

Biological skin contains a dense distribution of mechanoreceptors capable of spatially resolving pressure and texture information across large surface areas [6]. Electronic skin systems attempt to replicate this functionality through distributed arrays of sensing nodes embedded onto or within flexible materials.

Modern tactile sensing systems typically consist of large grids of sensors capable of converting mechanical deformation into measurable electrical signals. Depending on the sensing mechanism, the electrical output may correspond to changes in resistance, capacitance, voltage, or current [3]. As sensor density increases, the amount of wiring required by the system becomes a major design limitation. This challenge motivates the development of scalable sensing architectures capable of reducing wiring complexity while maintaining high spatial resolution.

The following sections introduce multiplexing methods, resonant circuits, sensing materials, and filter architectures relevant to the solutions in chapters three, four and five.

2.2. Piezoresistive Sensing Materials

Piezoresistive materials change resistance as a function of applied mechanical stress or pressure. These materials are commonly used in tactile sensing systems due to their low cost, and compatibility with flexible substrates [1].

When pressure is applied to a piezoresistive material, conductive pathways within the material change, resulting in a measurable variation in resistance [3]. This resistance variation can be used directly as a sensing mechanism or incorporated into resonant circuits and active filters.

Velostat and other conductive polymer materials are frequently used in flexible tactile sensing applications due to their ability to produce large resistance changes under compression. However, Velostat has shown to exhibit nonlinear behavior, hysteresis, and manufacturing variability, which complicates system-level calibration and scaling [4].

In this work, piezoresistive materials are incorporated into resonant sensing architectures to modulate either resonant amplitude or resonant frequency in response to applied pressure.

2.3. Frequency-Domain Multiplexing

Frequency-domain multiplexing is a signal processing technique where multiple signals occupy different frequency regions.

Frequency multiplexing can take a frequency-modulated form or an amplitude modulated form. Frequency modulation (FM) methods display tactical changes through frequency shifts in the carrier wave while amplitude modulation (AM) methods display tactical changes through amplitude shifts in the carrier wave. [1][8]

In tactile sensing applications, whether using AM or FM, each sensing node can be assigned a unique resonant frequency or frequency channel. The output spectrum of the system then contains

information about the spatial location and state of each sensing node. Frequency-domain multiplexing provides several advantages, including simultaneous sensing of multiple nodes, minimal crosstalk, and scalable system architectures [5].

However, the total number of sensing nodes that can occupy a given spectral range is constrained by resonator bandwidth, channel spacing, quality factor, and overall spectral bandwidth being designated for the system. If neighboring resonant responses overlap excessively, distinguishing individual nodes becomes difficult.

2.4. Passive Filters and Quality Factor

Resonant circuits are widely used in filtering, sensing, and communication systems due to their frequency-selective behavior. Due to the reduced requirement for additional circuit components, passive elements are used for flexible applications and cover large areas [1].

In RLC circuits, resonant frequency f_0 is determined by inductance and capacitance values.

$$f_0 = \frac{1}{2\pi\sqrt{LC}}$$

Bandwidth determines the spectral width of the resonator response, while the quality factor describes the sharpness of resonance.

$$Q = \frac{f_0}{BW}$$

Higher quality factors produce narrower resonances and allow more channels to occupy a fixed frequency range. However, high-Q systems may also exhibit longer settling times and increased sensitivity to component tolerances. Additionally for non-passive systems using an op-amp, the acceptable range of Q for that style of op amp should be considered

The relationship between resonant frequency, bandwidth, and quality factor plays a critical role in determining the scalability limits of frequency-domain tactile sensing systems.

2.5. Active Bandpass Filters

Active bandpass filters provide frequency-selective amplification using operational amplifiers in combination with resistors and capacitors. The Multiple Feedback (MFB) bandpass topology is particularly useful for resonant sensing applications because resonant frequency, bandwidth, and gain can be independently controlled through component selection.

In this work, active MFB filters are combined with variable capacitance elements to create resonant channels whose center frequencies shift as pressure is applied to the sensing material.

2.6. Varactors and Variable Capacitance

Varactors are voltage-controlled capacitors commonly used in frequency tuning applications. Their capacitance changes as a function of applied bias voltage. By coupling a piezoresistive material to a varactor, applied pressure can indirectly modify resonant frequency through capacitance variation. This allows pressure information to be encoded into frequency shifts within a resonant sensing system.

Variable capacitance techniques provide a method for implementing frequency modulation tactile sensing for an active circuit without requiring large physical deformation of the resonant circuit itself.

2.7. Nonlinear Optimization in Resonant System Design

Designing large resonant sensing arrays for complicated systems such as the active bandpass filter requires simultaneously satisfying multiple coupled constraints including:

- resonant frequency,
- bandwidth,
- gain,
- channel spacing,
- and physical component limitations

Because these relationships are nonlinear, analytical solutions become increasingly difficult as system complexity increases.

Numerical optimization methods provide a systematic way to solve for component values that satisfy multiple design objectives simultaneously. In this work, a system of equations along with nonlinear least-squares optimization techniques are used to calculate resistor and capacitor values for resonant sensing nodes while constraining channel overlap and maintaining desired bandwidth characteristics.

Iterative optimization methods are additionally used to scale the system from a single sensing node to large arrays occupying shared spectral bandwidth.

CHAPTER 3

Passive RLC Bandpass Architecture - AM

This chapter investigates a passive resonant sensing architecture utilizing amplitude modulation within parallel RLC resonators. The objective of this architecture is to minimize wiring complexity while maximizing achievable sensing density using passive components only.

In the proposed system, a piezoresistive material varies in parallel with a fixed resistor, modulating the amplitude response of the resonant frequency within a passive RLC filter. Particular emphasis is placed on resonator bandwidth limitations, achievable frequency spacing, and the physical constraints imposed by commercially available inductors and capacitors. The chapter first introduces the resonant topology and governing equations before analyzing practical bandwidth limitations and spectral scaling behavior.

3.1. General Solution

3.1.1. Parallel Vs Series Configuration. The document "Sinusoidal Steady State Response of Linear Circuits" highlights the key differences for series and parallel combinations in Figure 3.1 below [7]. For this application we will use a parallel combination, which is unique in that it has a bandwidth constrained by capacitance and Q factor constrained by inductance. Table 3.1 below shows important equations for series and parallel configurations.

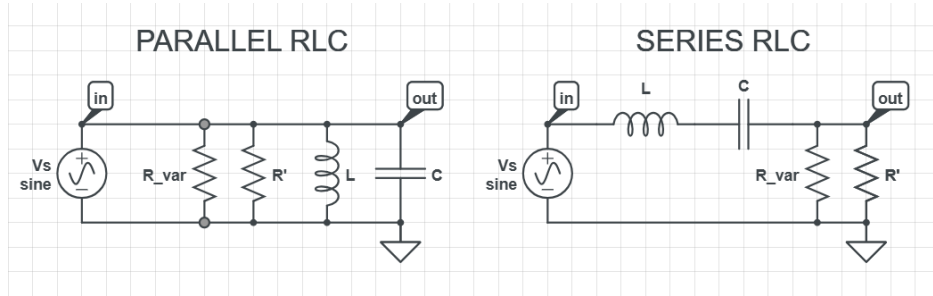


FIGURE 3.1. Parallel and Series RLC Circuits [7]

	Parallel	Series
Transfer Function	$ H(\omega) = \left \frac{I_R}{I_s} \right = \frac{\omega L}{\sqrt{(1-\omega^2 LC R)^2 + (\omega L)^2}}$	$ H(\omega) = \left \frac{V_R}{V_s} \right = \frac{\omega RC}{\sqrt{(1-\omega^2 LC)^2 + (\omega RC)^2}}$
Resonant Frequency	$\omega_0 = \frac{1}{\sqrt{LC}}$	$\omega_0 = \frac{1}{\sqrt{LC}}$
Bandwidth	$B_p = \omega_2 - \omega_1 = \frac{1}{RC}$	$B_s = \omega_2 - \omega_1 = \frac{R}{L}$
Q Factor	$Q = \frac{\omega_0}{B_p} = \omega_0 RC = \frac{R}{\omega_0 L}$	$Q = \frac{\omega_0}{B_s} = \frac{\omega_0 L}{R} = \frac{1}{\omega_0 RC}$

TABLE 3.1. Parallel and Series RLC Equations

3.1.2. Transfer Function and Circuit: The resonant frequency (Hz) of a parallel RLC circuit is:

$$f_0 = \frac{1}{2\pi\sqrt{LC}}$$

Bandwidth at -3db point, in Hz:

$$BW = \frac{1}{2\pi R_{eq} C}$$

3.1.2.1. *Substituting in a Parallel R.* Our piezoresistive material will be in parallel with a fixed resistor, R' . Therefore, R_{eq} is a the parallel combination of R' and R_{var} , our piezoresistive material:

$$R_{eq} = \frac{R' R_{var}}{R' + R_{var}}$$

Since R_{var} is at its smallest value when maximum pressure is applied, the resistances at maximum and minimum pressure would be:

$$R_{eq}(P_{max}) = \frac{R' R_{var,min}}{R' + R_{var,min}}$$

$$R_{eq}(P_{min}) = \frac{R' R_{var,max}}{R' + R_{var,max}}$$

Regardless of exact values, R_{eq} will be minimized at max pressure and maximized at min pressure.

3.1.3. Minimizing Bandwidth. In order to maximize the nodes in our system, the resonator bandwidth must be minimized, since smaller bandwidths allow resonant frequency channels to be packed closer together without overlap. Therefore, we must determine the values for R' and C at

which the bandwidth reaches its minimum, and minimize this worst-case value for the R_{var} range.

The equation for Bandwidth in section requires maximizing both R_{eq} and C to minimize BW. For the worst case situation we will assume R_{var} is minimized, assuming no pressure on the sensor. Therefore at this point, reducing the bandwidth requires keeping the fixed resistor R' as large as possible in order to keep R_{eq} high. Lets say the Velostat ranges from :

$$R_{var,min} = 1\text{ }k\Omega$$

$$R_{var,max} = 1\text{ }M\Omega$$

For availability reasons, the maximum available resistance value is capped to 1M ohms:

$$R'_{max} = 1\text{ }M\Omega$$

That would make R_{eq} equal to about 999 ohms

$$R_{eq}(P_{min}) = \frac{(R_{var,min})(R'_{max})}{R_{var,min} + R'_{max}} = \frac{(1 \times 10^3)(1 \times 10^6)}{1 \times 10^3 + 1 \times 10^6} \approx 999\Omega$$

Now, based on available parts at or below 1 mm x 1 mm, the largest capacitor is 22 uF (series GRM, 1.0mm x 0.5mm on Digikey).

$$BW = \frac{1}{2\pi R_{eq}C} = \frac{1}{(2\pi)(999\Omega)(22\text{ }uF)} = 7.24Hz$$

The smallest BW we can create is 7.2 Hz.

Next we should figure out what the minimum frequency this 7.2 Hz bandwidth can occur at. In order to do this you sub in the C at which min bandwidth occurs, and the maximum possible value of on the shelf inductors under 1 mm x 1 mm. The value of this inductor was found to be 470 nH (part 0201, 0.6mm x 0.3mm).

$$f_0 = \frac{1}{2\pi\sqrt{LC}} = \frac{1}{2\pi\sqrt{(470\text{ }nH)(22\text{ }uF)}} = 49.5kHz$$

Frequency	Inductance (with 22 μ F)	Footprint Size (A mm x B mm)
1 kHz	1.15 mH (1 mH + .15 mH)	(2.00 x 1.60) + (2.5 x 1.8)
10 kHz	11.51 μ H	7 x 7
100 kHz	115.14 nH (115 nH + .14 nH)	(1.6 x 0.8) + (0.47 x .28)
500 kHz	4.61 nH	0.6 x 0.3
1 MHz	1.15 nH (1 nH + 0.2 nH)	(0.47 x 0.28) + (0.47 x .28)

TABLE 3.2. Changing Resonant Frequencies

If we wanted to operate at different scales of resonant frequencies, Table 3.2 shows the necessary corresponding inductors. Notice that any values below 49.5k Hz have inductor footprints far outside of acceptable limits of 1mm x 1mm:

Notice that after the 49.5 kHz limit, the inductor sizing begins to fit closer to 1 mm x 1mm sizing. It should be noted that, in order to get specific frequencies, the addition of small inductors may be necessary.

3.1.4. Bandgap Sizing. Inorder for there to be no overlap between subsequent nodes, the distance between two nodes' resonant frequencies must be more than half the value of their bandwidths added.

$$f_{i+1} - f_i \geq \frac{1}{2} (BW_i + BW_{i+1})$$

$$f_{c2} - f_{c1} \geq \frac{1}{2} (BW_{max1} + BW_{max2})$$

BW_{max} occurs when R is at its highest value. This will be important in future work to consider that R', L and C values are chosen so that the equation is satisfied at both max pressure and minimum pressure scenerios.

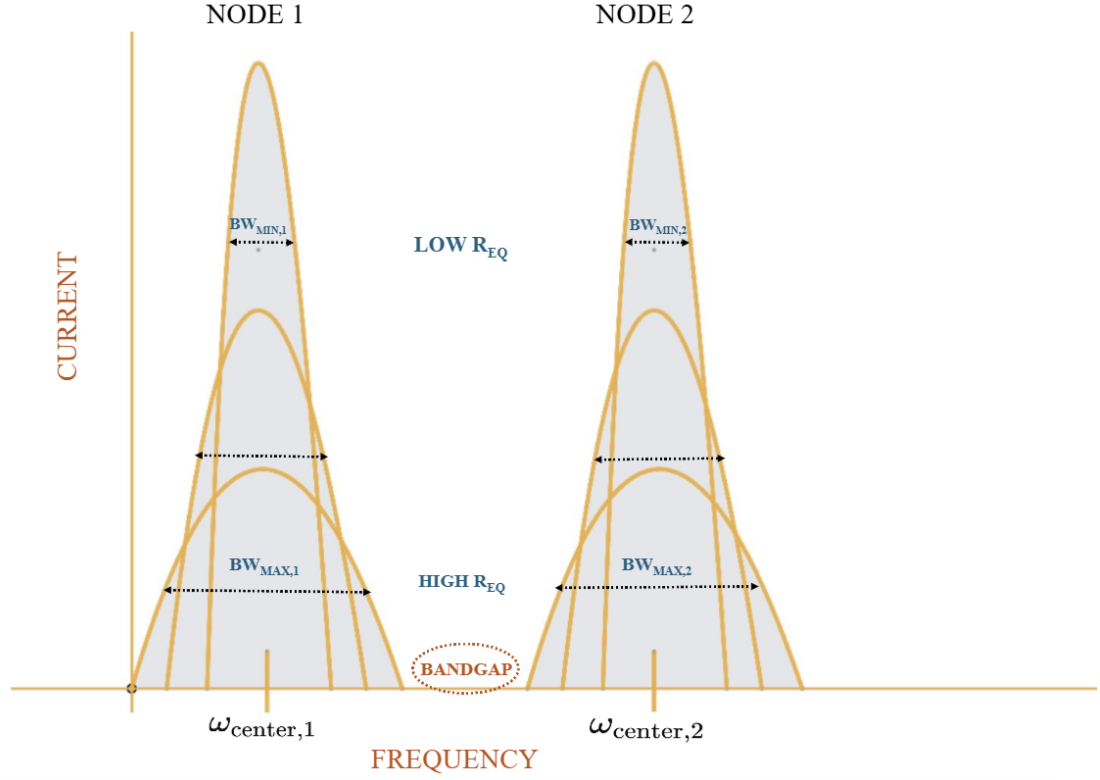


FIGURE 3.2. RLC

3.2. System Limitations

Sizing Limitations for capacitors and inductors are shown in Table 3.3 below:

Component	Value Range	Series	Size (mm)
Capacitor (min)	0.1 pF	GRM	0.6 X 0.3
Capacitor (max)	22 μ F	GRM	1.0 x 0.5
Inductor (min)	0.1 nH	LQP03	0.6 X 0.3
Inductor (max)	470 nH	LQP03	0.6 X 0.3

TABLE 3.3. Practical Capacitor and Inductor Limits

Inductor sizing and capacitor sizing constraints are the main limiting factors for this solution. As frequencies scale, finding exact values available on the market will be difficult and combining components boosts footprint sizing. The inductor sizing must be below 1mm x 1mm in order to keep the overall sensor footprint small. Staying with the 470 nH and 72 Hz minimum BW, while still keeping the capacitor below 22 uF, limits the resonant frequency range between 50 kHz and 1M.

CHAPTER 4

Active Varactor-Based Bandpass Architecture - FM

This chapter presents an active tactile sensing architecture based on multiple-feedback bandpass filters and variable capacitance techniques. In the proposed system, tactile pressure applied to a single node modifies the resonant frequency of that node through capacitance variation produced by a piezoresistive-controlled varactor network. The piezoresistive material changes the voltage applied to the varactor, causing the capacitance to vary from approximately 10 pF to 80 pF.

A transfer function describing resonant frequency as a function of applied pressure is first derived and extended to represent large sensing arrays where resonant behavior depends on both applied force and node location. Using this transfer function, a nonlinear system of equations is developed to constrain desirable resonator characteristics including bandwidth, quality factor, operable frequency range, gain, and channel spacing.

A Python-based optimization framework is then used to solve for component values satisfying these constraints for individual sensing nodes, where the optimized solution from each node is recursively used as the initial condition for subsequent nodes to scale the system across large resonant arrays.

4.1. General Solution

4.1.1. Transfer Function. Using KCL, we can transform the general circuit for a MFB Bandpass model into a transfer function, and comparing that transfer function to the general second order equation for a bandpass filter we can extract the center frequency of the filter in terms of its components.

We are using a Delliyannis-friend filter from Figure 4.1 for this setup, however the transfer function's general form and KCL from the Analog Devices Datasheet [12] were crosschecked:

MULTIPLE FEEDBACK BAND-PASS DESIGN EQUATIONS

$$\frac{-H \omega_0 s}{s^2 + \alpha \omega_0 s + \omega_0^2} \quad (1)$$

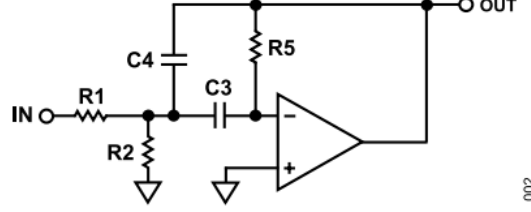


Figure 2.

$$\frac{V_O}{V_{IN}} = \frac{-s \frac{1}{R_1 C_4}}{s^2 + s \frac{(C_3 + C_4)}{C_3 C_4 R_5} + \frac{1}{R_5 C_3 C_4} \left(\frac{1}{R_1} + \frac{1}{R_2} \right)} \quad (2)$$

FIGURE 4.1. Analog Devices MT-218 Diagram from Datasheet

Since $\left(\frac{1}{R_1} + \frac{1}{R_2} \right) = \frac{1}{R_1 || R_2}$, we can simplify the equation from the figure above into the equation below

$$(1.1) \quad \frac{V_{out}}{V_{in}} = \frac{-s \frac{1}{R_1 C_4}}{s^2 + s \frac{C_3 + C_4}{C_3 C_4 R_5} + \frac{1}{(R_1 || R_2) C_3 C_4 R_5}}$$

setting $C_3 = C_4$ and simplifying:

$$(1.2) \quad \frac{V_{out}}{V_{in}} = \frac{-s \frac{1}{R_1 C}}{s^2 + s \frac{2}{C R_5} + \frac{1}{(R_1 || R_2) C^2 R_5}}$$

4.1.2. Finding Gain, Q, BW, and Resonant Frequency. Equation 2 below is the general form for a second order bandpass filter. We will use this format to find equations for Gain, Q, Resonator BW and Resonant Frequency in terms of R and C. In this chapter BW will always represent a singular resonator BW for one node, and resonant frequency and center frequency are used interchangeably.

$$(2) \quad H(s) = \frac{k \frac{\omega_{center}}{Q} s}{s^2 + \frac{\omega_{center}}{Q} s + \omega_{center}^2}$$

Subbing in an equation for bandwidth, $BW = \frac{\omega_{center}}{Q}$

$$(2.1) \quad H(s) = \frac{k * BW * s}{s^2 + BW * s + \omega_{center}^2}$$

Comparing equation 1.2, the KCL for our filter, with equation 2.1, a second order bandpass filter we can find maximum passband gain (K), Quality factor (Q), Resonator Bandwidth (BW) and Resonator Center Frequency (ω_{center}):

$$\frac{V_{out}}{V_{in}} = H(s)$$

$$\frac{-s \frac{1}{R_1 C}}{s^2 + s \frac{2}{C R_5} + \frac{1}{(R_1 || R_2) C^2 R_5}} = \frac{k * BW * s}{s^2 + BW * s + \omega_{center}^2}$$

Bandwidth and Resonant Frequency of the node are changing with the capacitance of the varactor, (at the same rate) which is dependent on pressure (P):

$$(3) \quad BW = \frac{2}{C(P) R_5}$$

$$(4) \quad \omega_{center} = \sqrt{\frac{R_1 + R_2}{R_1 R_2 C(P)^2 R_5}}$$

Resonator Gain and Q factor are not dependent on capacitance when C3=C4, meaning that they will not change as pressure is applied to the sensor:

$$(5) \quad |k| = \frac{R_5}{2 R_1}$$

$$(6) \quad \frac{\omega_{center}}{\omega_{BW}} = Q = \frac{1}{2} \sqrt{\frac{R_5 (R_1 + R_2)}{R_1 R_2}}$$

4.1.3. Modeling The Varactor. Now that we have equations for resonant frequency of bandpass filter, we must be able to shift its resonant frequency by small amounts as a reaction to pressure applied to the node. To do this we use variable capacitors.

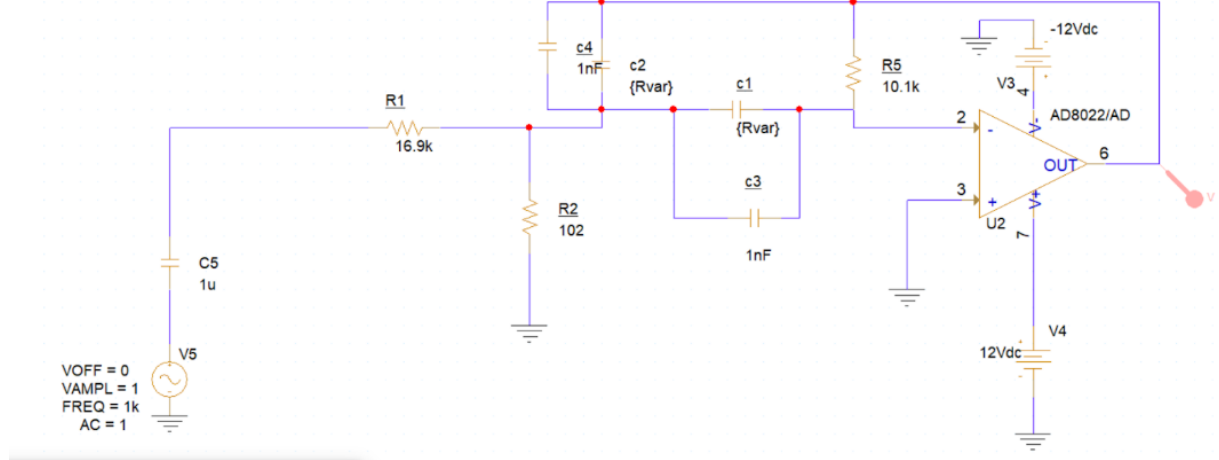


FIGURE 4.2. Circuit with Initial Values

Both $C3 = C4 = C$ are attached in parallel to separate varactors, allowing us to model this behavior with equation 7 below, where C' is the value of the capacitor and C_{var} is the variable value of the capacitor:

$$(7) \quad C(P) = C' + C_{var}(P)$$

Boundary Conditions:

$$C(P_{max}) = C' + C_{var,min}$$

$$C(P_{min}) = C' + C_{var,max}$$

$$\Delta C_{var,max} = C(P_{max}) - C(P_{min}) \approx 60pF.$$

Due to the addition of a variable capacitor, Node 1 will have a different resonant frequency depending on how much pressure is applied to that node. Looking at equation 4 we can see the minimum resonant frequency will be at $C(P_{min})$ and the maximum resonant frequency will be at $C(P_{max})$.

For Node 1, the range from the minimum possible resonant frequency to maximum possible resonant frequency will be called Channel 1. The terms node and channel will be used interchangeably.

Node 2 will operate in Channel 2, which will purposely have its minimum and maximum possible resonant frequencies outside of Channel 1.

4.1.4. Settling Time. Settling time will determine how fast it takes the resonator to reset to its undisturbed (no touch) frequency after pressure is done being applied. Ideally, if we pressed the same amount on every node in the system, they would all have the same settling time.

Comparing the denominators of 1.2 and equation 2, considering $Q = \frac{1}{2\zeta}$, we can find the damping coefficient:

$$(4.1) \quad s^2 + 2\zeta \omega_{center}s + \omega_{center}^2 = s^2 + s \frac{2}{CR_5} + \frac{1}{(R_1 || R_2)C^2 R_5}$$

$$(8) \quad \zeta = \sqrt{\frac{R_1 R_2}{R_5(R_1 + R_2)}}$$

For a second-order system, settling time T_s for 2% settling time can be approximated as the following, assuming that $\zeta \ll 1$

$$(9) \quad T_s = \frac{4}{\zeta \omega_{center}}, \zeta < 1$$

We know $\zeta = 1/2Q$

$$(9.1) \quad T_s = \frac{4}{\frac{\omega_{center}}{2Q}}, \zeta < 1$$

Bandwidth of the resonator, $BW = \omega_{center}/Q$

$$(9.2) \quad T_s = \frac{8}{BW}, \zeta < 1$$

In order for the settling time to remain consistent from Channel 1 to Channel 2 to Channel 3, we will have to make the ratio of $\frac{\omega_{center}}{Q}$ (aka Resonator Bandwidth) constant. As we add more nodes, the starting center frequency of each channel will be increased, so we will have to increase the Q to keep this ratio the same from Channel 1 to Channel 2 to etc.

4.1.5. Bandwidth Inside of the Channel. Consistant settling time means that within a given Channel (lets say Channel 1), Q and gain (k) will remain the same regardless of how much pressure is applied as they are solely dependent on the set resistor's values in equations 5 and 6.

So within an individual channel the bandwidth of a signal will increase with frequency from $BW_{min,1}$ to $BW_{max,1}$. Notice in Figure 4.3, the bandwidth slowly increases within Channel 1, but then begins back at its base value at the beginning of Channel 2

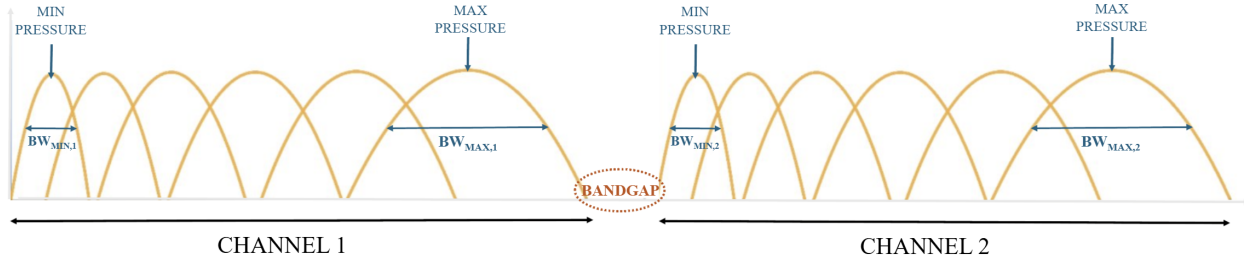


FIGURE 4.3. Bandwidth Increase Within Nodes

Therefore we will say for the system, a resonator corresponding to no touch will have the same bandwidth for every single channel, shown in the equations below.

The minimum bandwidth will be the same in every single channel

$$(10) \quad BW_{min,1} = BW_{min,2} = BW_{min,3} = \dots$$

Also, the maximum bandwidth will be the same in every single channel

$$(10.1) \quad BW_{max,1} = BW_{max,2} = BW_{max,3} = \dots$$

However, the maximum and minimum BWs will not be the same, as they change from the beginning of each channel to the end of each channel.

$$(10.2) \quad BW_{min} \neq BW_{max}$$

This will end up making it so that the Q factor of the system increases with each channel towards a max Channel N. $Q_{\max} = \frac{f_{start,n}}{BW_{min}}$

4.1.6. Consistent Channel Width. The offset in the beginning of the channel ($BW/2$) will not equal the $BW/2$ of the end of the channel, but they will be the same for every single channel:

$$(11) \quad f_{i+1} - f_i \geq \frac{1}{2} (BW_i + BW_{i+1})$$

$$(11.1) \quad f_2 - f_1 \geq \frac{1}{2} (BW_{max} + BW_{min})$$

Since $\omega_{center,max} - \omega_{center,min}$ is the same for every channel, and we know that the maximum BW and minimum BW is also the same for every channel, equation 11.1 proves the channel width will remain consistent for the entire system. Bandgap sizing, as shown in Figure 4.3, can be customized. Making the left side of equation 1 equal to the right size will result in no bandgap, and as the left side grows larger than the right, the bandgap grows.

A limiting factor could be that if we hold the bandwidth to be the same for every Channel, then the Q factor would increase as the starting frequency of each channel increases. The op amp we are currently using can only withstand a certain range of Q factors, meaning at higher nodes we would have to change op amp.

4.2. Nonlinear Solver

Using the transfer function, a system of equations with 4 equations and 4 unknowns is developed in order to constrain desirable characteristics such as BW_{min} , Gain (k), ω_{start} , and ω_{range} for one node.

An optimization function in Python is used to solve this system of equations and return component values R1, R2, R5, C that meet the given constraints for our first node. BW_{min} , Gain (k) and ω_{range} remain the same for the entire system but f_{min} changes from channel to channel

The following equations will relate our inputs on the left to the outputs on the right, assume that we already know the value of $C_{var,min}$ and $C_{var,max}$

1. Bandwidth of the resonator with no touch will be the same for all channels

$$(12) \quad BW_{min} = \frac{2}{(C' + C_{var,max})R_5}$$

2. Maximum gain will not be dependent on capacitance, determined by C3=C4 applied to the TF

$$(13) \quad k = \frac{R_5}{2R_1}$$

3. The range of frequencies will be the same for all channels

$$(14) \quad w_{range} = \left(\frac{1}{C_{var,min} + C'} - \frac{1}{C_{var,max} + C'} \right) \sqrt{\frac{R_1 + R_2}{R_1 R_2 R_5}}$$

or equivalently at max touch:

$$(14.1) \quad \omega_{center,max} = \sqrt{\frac{R_1 + R_2}{R_1 R_2 R_5 (C' + C_{var,min})^2}}$$

4. The resonator center frequency when no pressure is applied will define what frequency each channel starts at

$$(15) \quad \omega_{center,min} = \sqrt{\frac{R_1 + R_2}{R_1 R_2 R_5 (C' + C_{var,max})^2}}$$

Putting these equations into a non-linear solver will give us a way to calculate R1, R2, R5 and C

4.2.1. Solver Code. The solver will use the system of equations along with a `scipy.optimize` least squares optimization to solve for the best R and C values for one node.

Since the equations are used for a single node, additional nodes are generated iteratively. The solver uses the solution from Node 1 as the initial guess for Node 2, and updates the frequency range so that $\omega_{min,2} = \omega_{max,1} + increment$. The value of increment is an adjustable spacing between the nodes.

To start, the increment is set to $2 \times BW_{min}$ with no guardband, meaning that in theory the nodes should all be touching at the -3db point.

4.2.2. Finding the Right Weighting . To validate the solver's results, we substituted the optimized component values into the existing physical equations for Frequency, Bandwidth (BW), and Gain. These values were then compared directly with the corresponding solver outputs. Any differences between the two were traced to unequal weighting of the residuals across different equations. To address this, we manually adjusted the weighting factors until the analytical results closely matched the solver's outputs, as seen next to the equations for f1 and f2 in Figure 4.4.

```
def F(x):

    R1, R2, R5, Cprime = x
    # Scaling to make the solvers steps similar (ie change by one unit nF vs kOHM)
    R1 = abs(R1*1e3)
    R2 = abs(R2*1e3)
    R5 = abs(R5*1e3)
    Cprime = Cprime*1e-9

    f1 = (wbw_min - 2 / ((Cprime + Cvar_max) * R5))*1e9
    f2 = (K - R5 / (2 * R1))*1e12
    f3 = (wc_max_SQ - (R1 + R2)/(R1*R2*R5*(Cprime + Cvar_min)**2))
    f4 = (wc_min_SQ - (R1 + R2)/(R1*R2*R5*(Cprime + Cvar_max)**2))
    return [f1, f2, f3, f4]

# Perform Optimization
sol = least_squares(F, x0, method = 'lm', ftol=1e-8, xtol=1e-5, gtol=1e-8)
```

FIGURE 4.4. Manually Adjusted Weightings on f1 and f2

The result of the code in Figure 4.4 is shown below in the Node 2 output of Figure 4.5. The expected and actual values are even showing that the weighting is working

```

===== NODE 2 =====
Initial x0 used: [512.081541, 19.674992, 303.152273, 0.13]
Initial f_min=17500.00 Hz, f_max=22500.00 Hz

=== New Range Component Values ===
R1 = 3.414e+05  $\Omega$ 
R2 = 4.175e+03  $\Omega$ 
R5 = 2.021e+05  $\Omega$ 
C' = 2.350e-10 F

=== Frequency, Bandwidth & Gain Comparison ===

```

Type	f_min [Hz]	f_max [Hz]	BW_min [Hz]	K Gain
Expected (Inputs)	17500.00	22500.00	5000.00	0.2960
Actual (From R/C)	17500.00	22500.00	5000.00	0.2960
Error (%)	0.0000	0.0000	0.0000	0.0000

FIGURE 4.5. Node 2 with weighting

If the weighting is removed, the output for the same node will look like Figure 4.6, where the error is made clear by the mismatch in BW and Gain. When troubleshooting the code, weighting is the most important thing to check first, especially when changing initial conditions or inputs.

```

===== NODE 2 =====
Initial x0 used: [512.081541, 19.674992, 303.152273, 0.13]
Initial f_min=17500.00 Hz, f_max=22500.00 Hz

=== New Range Component Values ===
R1 = 1.994e+06  $\Omega$ 
R2 = 6.802e+03  $\Omega$ 
R5 = 1.230e+05  $\Omega$ 
C' = 2.350e-10 F

=== Frequency, Bandwidth & Gain Comparison ===

```

Type	f_min [Hz]	f_max [Hz]	BW_min [Hz]	K Gain
Expected (Inputs)	17500.00	22500.00	5000.00	0.2960
Actual (From R/C)	17500.00	22500.00	8217.27	0.0308
Error (%)	0.0000	0.0000	64.3454	-89.5835

FIGURE 4.6. Node 2 without weighting

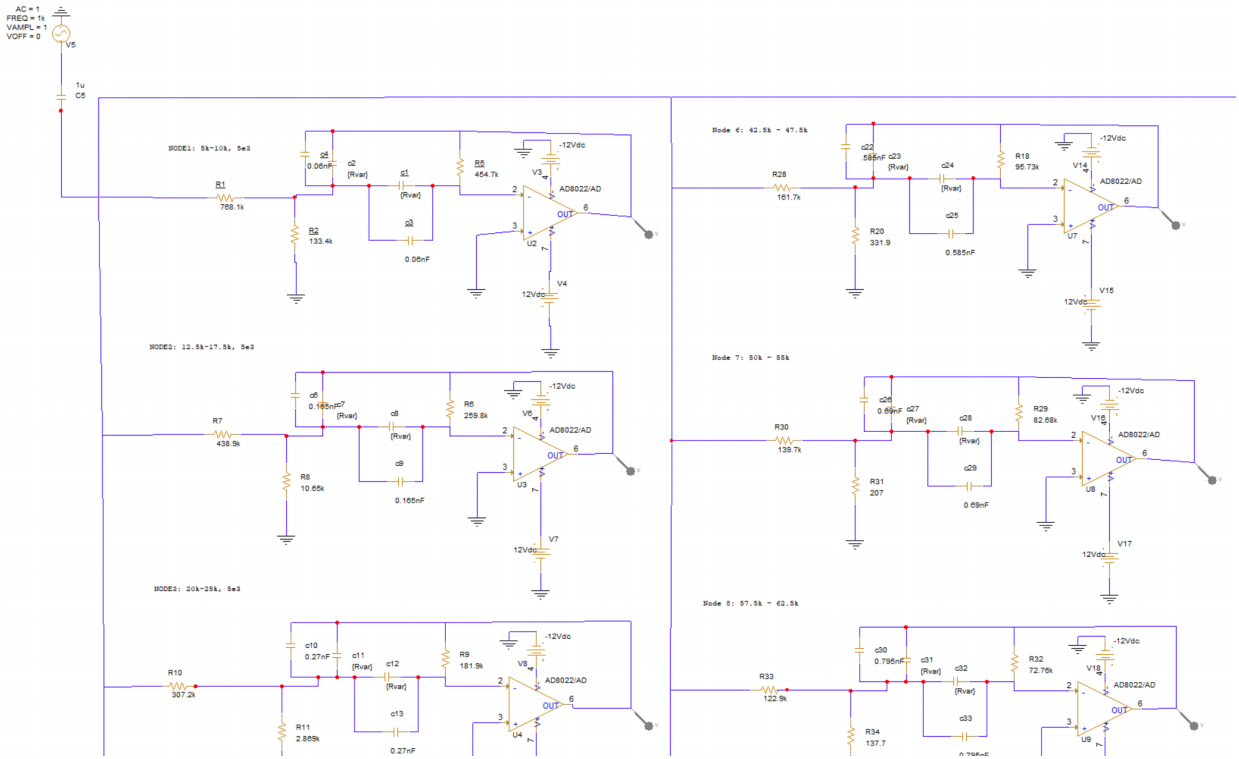
To ensure the solver operates on similarly scaled inputs, we normalized the parameter space at the top of 4.4. Since capacitances are in the pF–nF range, resistances span $\Omega - k\Omega$, and frequencies are in the kHz range. To improve numerical stability across these differing scales, we introduced weighting factors for the parameters R1, R2, R5, and C_{prime} .

Additionally, for the initial conditions, the solution for Node 2 uses the results from Node 1 as its starting point.

Finally to minimize complexity for the solver we took the square root out of the equation by squaring each side, which is why you will see f3 and f4 have elements that are squared.

4.3. Simulated Circuit

The initial circuit diagram in Cadence OrCAD X is shown in Figure 4.7. The starting values for Node 1 were based on an existing circuit and the values for the subsequent nodes were created using our iterative solver using the method in Section 4.2.



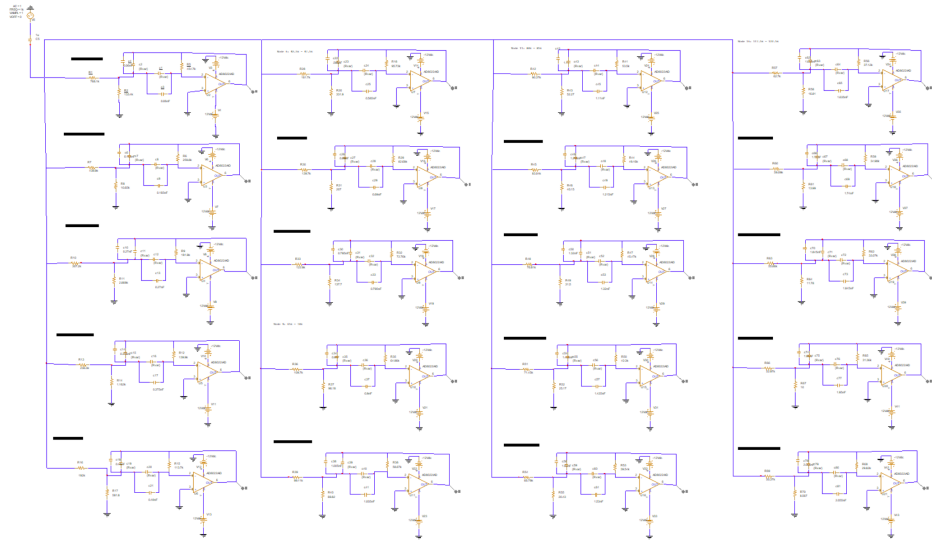


FIGURE 4.8. 20 Nodes

4.3.1. Finding a BW and Frequency Range that Fits Resistor Constraints

- . There are 3 adjustable inputs, $f_{c,min}$ for Node 1, $f_{c,range}$ aka how much the frequency will change from max to min touch, and BW of the resonator.

Original circuit was operating at 144k-155k Hz, and 30 kHz BW, shown below:

...

```
===== NODE 1 =====
Initial x0 used: [219.5, 0.2258, 129.9, 0.41]
Initial f_min=144000.00 Hz, f_max=155000.00 Hz
```

```
=== New Range Component Values ===
```

```
R1 = 1.817e+04 Ω
```

```
R2 = 1.175e+02 Ω
```

```
R5 = 1.076e+04 Ω
```

```
C' = 9.064e-10 F
```

```
=== Frequency, Bandwidth & Gain Comparison ===
```

Type	f_min [Hz]	f_max [Hz]	BW_min [Hz]	BW_max [Hz]	K Gain
Expected (Inputs)	144000.00	155000.00	30000.00	---	0.2960
Actual (From R/C)	144000.00	155000.00	30000.00	32291.67	0.2960
Error (%)	0.0000	-0.0000	0.0000		0.0000

FIGURE 4.9. Node 1, from 144-155k

After simulation it was found that using the orginial circuit values the array will only produce 5 nodes before the R2 sizing becomes too small as each iteration occurs. Due to scaling, the resistor must not come near trace resistance, he lower limit being around 10 ohms. Figure 4.10 shows the highest node in the system, and how R2 decreased from 117 ohms in Node 1 to 12 ohms in Node 5. This resistor sizing will be a major limitation for the system.

```

===== NODE 5 =====
Initial x0 used: [10.13111, 0.018965, 5.997617, 1.689091]
Initial f_min=308000.00 Hz, f_max=319000.00 Hz

=== New Range Component Values ===
R1 = 8.829e+03  $\Omega$ 
R2 = 1.241e+01  $\Omega$ 
R5 = 5.227e+03  $\Omega$ 
C' = 1.950e-09 F

=== Frequency, Bandwidth & Gain Comparison ===
Type                f_min [Hz]  f_max [Hz]  BW_min [Hz]  BW_max [Hz]  K Gain
-----
Expected (Inputs)    308000.00   319000.00   30000.00     ---          0.2960
Actual (From R/C)    308000.00   319000.00   30000.00     31071.43     0.2960
-----
Error (%)            0.0000      0.0000      0.0000                      0.0000
-----

```

FIGURE 4.10. Node 5, Limited by R2

We want to find a combination of inputs that will allow us to produce the maximum amount of nodes for the system before running into resistor sizing limitations. Tests 1-3 will change either in $f_{c,min}$, $f_{c,range}$, or BW to find the combination that allows for the most nodes.

Test 1:

Comparison of systems with Node 1 $f_{c,min}$ starting at 10 kHz versus $f_{c,min} = 100$ kHz while maintaining the same frequency fluctuation and resonator bandwidth.

Range & BW	Node 1		Node 20		IC
	Resistance (Ω)	Capacitance (F)	Resistance (Ω)	Capacitance (F)	
10k-20k, 3e3	1.4×10^4	6×10^{-11}	2.7	1.5×10^{-9} , 238k	[219.5, 0.2258, 129.9, 0.41]
10k-20k, 5e3	2.9×10^4	6×10^{-11}	3.6	1.7×10^{-9}	[219.5, 0.2258, 129.9, 0.41]
10k-20k, 10e3	6.6×10^4	6×10^{-11}	4.28	2×10^{-9}	[219.5, 0.2258, 129.9, 0.41]
10k-20k, 15e3	—	—	4.1	2.1×10^{-9}	—
100k-110k, 3e3	30	6.9×10^{-10}	1	2.1×10^{-9} , 328k	—
100k-110k, 5e3	51	6.9×10^{-10}	1.4	2.3×10^{-9}	—
100k-110k, 10e3	103	6.9×10^{-10}	1.9	2.68×10^{-9} , 395k	—

TABLE 4.1. Test 1: Keeping 10k Frequency Range and Varying Bandwidth

Operating within the 10–20 kHz range produced significantly larger Node 20 resistance values compared to operating within the 100–110 kHz range. This indicates that lower center-frequency ranges allow additional nodes to be generated before the resistor values approach the practical lower resistance limit.

Test 2a:

Varying $f_{c,\text{range}}$ while keeping $f_{c,\text{min}}$ constant and maintaining a resonator bandwidth of 5 kHz.

This test investigates how the width of the frequency shift caused by pressure affects component scaling while operating at low center frequencies.

Range & BW	Node 1		Node 20		IC
	Resistance (Ω)	Capacitance (F)	Resistance (Ω)	Capacitance (F)	
10–15, 5e3	1.9×10^4	1.3×10^{-10}	7.7	2.12×10^{-9}	
10k–20, 5e3 (repeat)	2.9×10^4	6×10^{-11}	3.6	1.7×10^{-9}	[219.5, 0.2258, 129.9, 0.41]
10k–30, 5e3	3.9×10^4	2.5×10^{-11}	1.3	1.5×10^{-9} , 457k	—
10k–40, 5e3	4.4×10^4	1×10^{-11}	6×10^{-1}	1.4×10^{-9}	—

TABLE 4.2. Keeping bandwidth fixed at 5k while varying frequency range

Test 2b:

Varying $f_{c,\text{range}}$ while keeping $f_{c,\text{min}}$ constant and maintaining a resonator bandwidth of 10 kHz.

Range & BW	Node 1		Node 20		IC
	Resistance (Ω)	Capacitance (F)	Resistance (Ω)	Capacitance (F)	
10–15, 10e3	4.4×10^4	1.32×10^{-10}	6.9	2.79×10^{-9}	
10k–20, 10e3 (repeat)	6.67×10^4	6×10^{-11}	4.28	2.055×10^{-9}	[219.5, 0.2258, 129.9, 0.41]
10k–30, 10e3	8.9×10^4	2.5×10^{-11}	1.9	1.6×10^{-9} , 457k	—

TABLE 4.3. Keeping bandwidth fixed at 10k while varying frequency range

Results from Tests 2a and 2b show that reducing $f_{c,\text{range}}$ helps prevent the iterative reduction of R_2 values, regardless of the selected bandwidth.

Smaller frequency ranges therefore improve scalability by allowing additional nodes to be generated before resistance values approach impractical limits.

Test 3:

Varying resonator bandwidth while keeping both $f_{c,\text{range}}$ and $f_{c,\text{min}}$ constant.

Range & BW	R2 at Node 20 (Ω)
10k–15k, 5e3	7.7
10k–15k, 10e3	6.93
10k–20k, 5e3	3.6
10k–20k, 10e3	4.28

TABLE 4.4. Comparison of Node 20 resistance values for varying frequency ranges and bandwidths

These results are consistent with the expected channel-spacing behavior. As discussed before, in order to prevent overlap between adjacent nodes, the following condition must be satisfied:

$$f_{i+1} - f_i \geq \frac{1}{2} (BW_i + BW_{i+1})$$

$$f_{c,min2} - f_{c,max1} \geq \frac{1}{2} (BW_{min,2} + BW_{max,1})$$

We already know based on equation 10 that across all channels the minimum bandwidth of the resonator (bandwidth of resonators at minimum touch) is the same

$$BW_{min,1} = BW_{min,2} = BW_{min,3} \dots$$

$$BW_{max,1} = BW_{max,2} = BW_{max,3} \dots$$

$$BW_{min} \neq BW_{max}$$

Therefore a large BW means that the the starting frequency of Node 2 is farther from Node 1, and the size that one node takes up is much larger.

As a result, each node occupies a larger portion of the available spectrum, reducing the total number of nodes that can fit within a fixed system frequency range, as shown by lower values of R2 in Table 4.4, the limiting factor for the system .

4.3.2. Simulation Results. This is the simulation of 20 nodes at both min and max touch in Cadence OrCAD X. For example the first 2 peaks are both Node 1, the first being minimum touch at node 1 and second peak being max touch at node 1. The 3db point should be at approximately 212mV. Looking closely, you can see that the end of each node overlaps very tightly with the beginning of the next node.

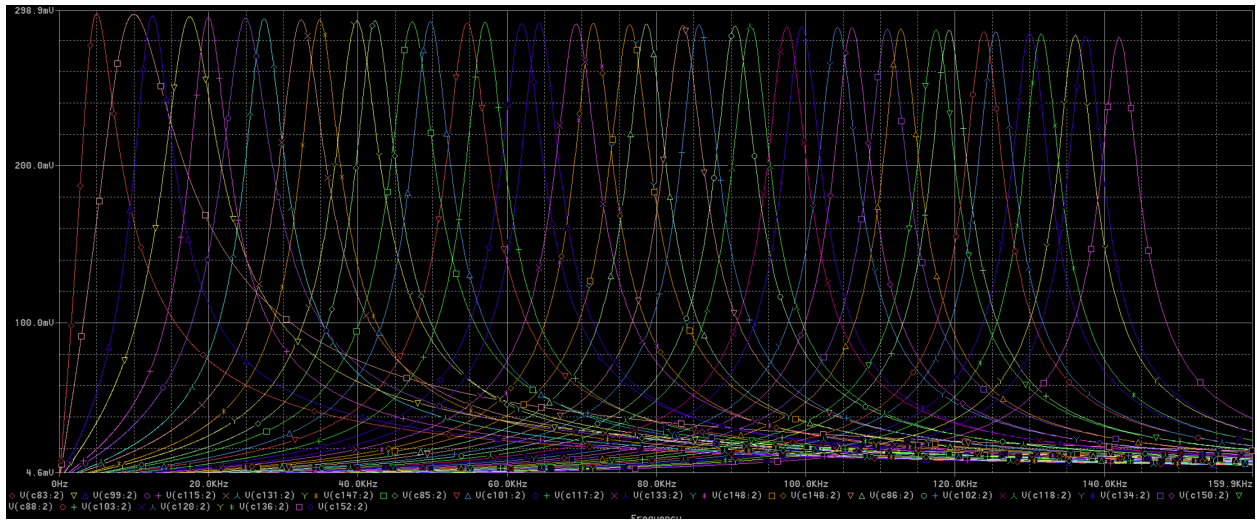


FIGURE 4.11. 20 nodes, Min and Max touch

One solution to this would to only use the center frequencies for every other node, creating little to no overlap between nodes but limiting the system to 10 nodes instead of 20, like Figure 4.12 below.

Alternatively, simply adding more space between the 20 nodes in the solver would not solve this issue, as elongating the total frequency range only makes the R2 of the final node smaller, and it is already close to its limit of 10ohms.

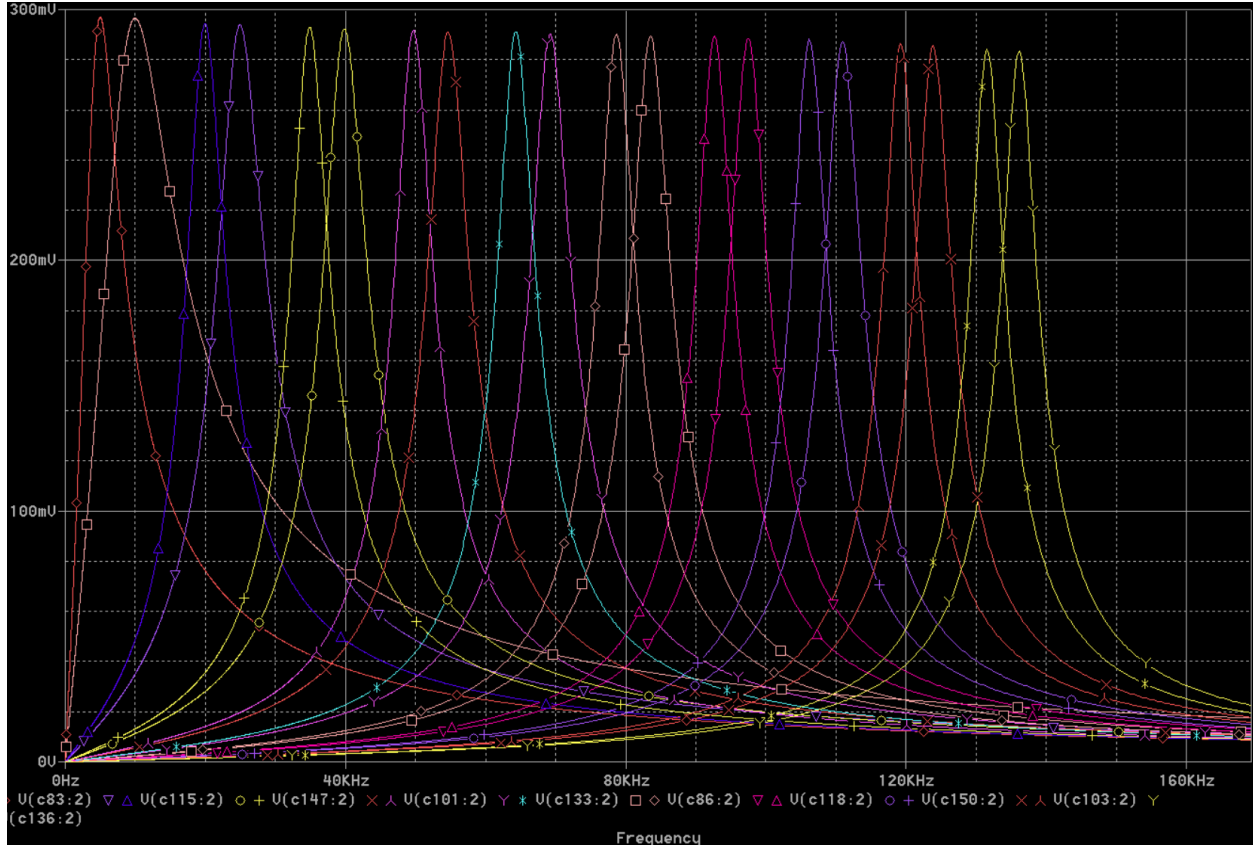


FIGURE 4.12. 10 nodes full size

In the photo above it is clear that the bandwidth at 212mV for the max touch is larger at the first few nodes, shown especially by the width of the second peak. This may cause the settling time to be slightly different for those nodes. The BW is staying at 5k however the max BW is around 10k for the first node and settles closer to 6k at Node 3, similarly to the trend of the max BW in Figure 4.14. This variation is due to setting the min BW to be the same equations 12-15 but leaving max BW unbounded.

Next, instead of using only 10/20 nodes we will use the code to create a 1 x BW separation between the max center frequency of Node 1 and the min center frequency of Node 2. This method allows us to create 15 nodes. This separation makes it so that at the 3db bandwidth there is little to no overlap and the frequencies are packed together as much as possible. The values for these 15 nodes are as follows:

The output of these values is shown below, cropping at 3db for clarity on overlap:

Node	$f_{\min}$ (Hz)	$f_{\max}$ (Hz)	R_1 (Ω)	R_2 (Ω)	R_5 (Ω)	C' (F)
1	5000	10000	7.681×10^5	1.334×10^5	4.547×10^5	6.000×10^{-11}
2	15000	20000	3.841×10^5	6.421×10^3	2.274×10^5	2.000×10^{-10}
3	25000	30000	2.560×10^5	1.525×10^3	1.516×10^5	3.400×10^{-10}
4	35000	40000	1.920×10^5	5.818×10^2	1.137×10^5	4.800×10^{-10}
5	45000	50000	1.536×10^5	2.812×10^2	9.095×10^4	6.200×10^{-10}
6	55000	60000	1.280×10^5	1.568×10^2	7.579×10^4	7.600×10^{-10}
7	65000	70000	1.097×10^5	9.618×10^1	6.496×10^4	9.000×10^{-10}
8	75000	80000	9.602×10^4	6.320×10^1	5.684×10^4	1.040×10^{-9}
9	85000	90000	8.535×10^4	4.373×10^1	5.053×10^4	1.180×10^{-9}
10	95000	100000	7.681×10^4	3.150×10^1	4.547×10^4	1.320×10^{-9}
11	105000	110000	6.983×10^4	2.344×10^1	4.134×10^4	1.460×10^{-9}
12	115000	120000	6.401×10^4	1.791×10^1	3.789×10^4	1.600×10^{-9}
13	125000	130000	5.909×10^4	1.399×10^1	3.498×10^4	1.740×10^{-9}
14	135000	140000	5.487×10^4	1.114×10^1	3.248×10^4	1.880×10^{-9}
15	145000	150000	5.121×10^4	9.013×10^0	3.032×10^4	2.020×10^{-9}

TABLE 4.5. Bandpass Filter Component Values for Nodes 1–15

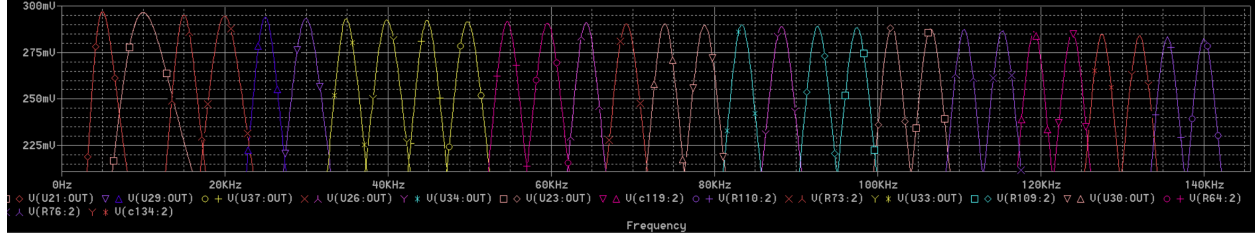


FIGURE 4.13. 15 nodes, 3db cropping

For this model, Node 1 is heavily overlapping into node 2, as expected based on our equations 12-15. The max bandwidth for each node should be around the same size so that each node takes up the same amount of spectrum. The issue with Node 1 is that its max BW is too large, making the frequency range that Node 1 operates in overlap with the frequency range of Node 2. As seen in both Figure 4.13 above and 4.14 below, the max BW stabilizes down to 6k after the first node, which allows for minimal overlap.

```

===== NODE 1 =====
Initial x0 used: [219.5, 0.2258, 129.9, 0.41]
Initial f_min=5000.00 Hz, f_max=10000.00 Hz

=== New Range Component Values ===
R1 = 7.681e+05  $\Omega$ 
R2 = 1.334e+05  $\Omega$ 
R5 = 4.547e+05  $\Omega$ 
C' = 6.000e-11 F

=== Frequency, Bandwidth & Gain Comparison ===
Type          f_min [Hz]  f_max [Hz]  BW_min [Hz]  BW_max [Hz]  K Gain
-----
Expected (Inputs)      5000.00    10000.00    5000.00      ---          0.2960
Actual (From R/C)      5000.00    10000.00    5000.00    10000.00     0.2960
-----
Error (%)             -0.0000     0.0000     -0.0000          0.0000
-----

===== NODE 2 =====
Initial x0 used: [768.122312, 133.429697, 454.728409, 0.06]
Initial f_min=15000.00 Hz, f_max=20000.00 Hz

=== New Range Component Values ===
R1 = 3.841e+05  $\Omega$ 
R2 = 6.421e+03  $\Omega$ 
R5 = 2.274e+05  $\Omega$ 
C' = 2.000e-10 F

=== Frequency, Bandwidth & Gain Comparison ===
Type          f_min [Hz]  f_max [Hz]  BW_min [Hz]  BW_max [Hz]  K Gain
-----
Expected (Inputs)     15000.00    20000.00    5000.00      ---          0.2960
Actual (From R/C)     15000.00    20000.00    5000.00    6666.67     0.2960
-----
Error (%)              0.0000     0.0000     0.0000          0.0000
-----

===== NODE 3 =====
Initial x0 used: [384.061156, 6.421267, 227.364204, 0.2]
Initial f_min=25000.00 Hz, f_max=30000.00 Hz

=== New Range Component Values ===
R1 = 2.560e+05  $\Omega$ 
R2 = 1.525e+03  $\Omega$ 
R5 = 1.516e+05  $\Omega$ 
C' = 3.400e-10 F

=== Frequency, Bandwidth & Gain Comparison ===
Type          f_min [Hz]  f_max [Hz]  BW_min [Hz]  BW_max [Hz]  K Gain
-----
Expected (Inputs)     25000.00    30000.00    5000.00      ---          0.2960
Actual (From R/C)     25000.00    30000.00    5000.00    6000.00     0.2960
-----
Error (%)              0.0000     0.0000     0.0000          0.0000
-----

```

FIGURE 4.14. jump from 10k max bw to 6k

To avoid any overlap and create a cleaner system, Node 1 was erased from the count entirely and the new system start at 15k Hz instead. The final node in Table 4.5 had a R2 value of 9 ohms so it was excluded as well. That leaves the system with 13 nodes ranging from 15k to 140k (12k to 142.5k) Hz, shown in Figure 4.15. All nodes below are evenly spaced so that there is little to no overlap.

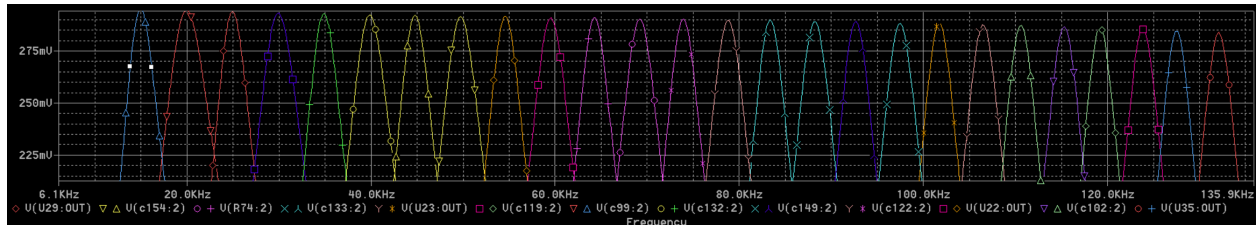


FIGURE 4.15. starting and ending frequencies for 13 nodes, 3db cutoff

Adding a guard band to our increment will help distinguish the nodes even further, 4.16 was created. The best node-to-separation ratio we were able to reach was 9 nodes with a 3k separation between each, making it clear the bounds of the node.

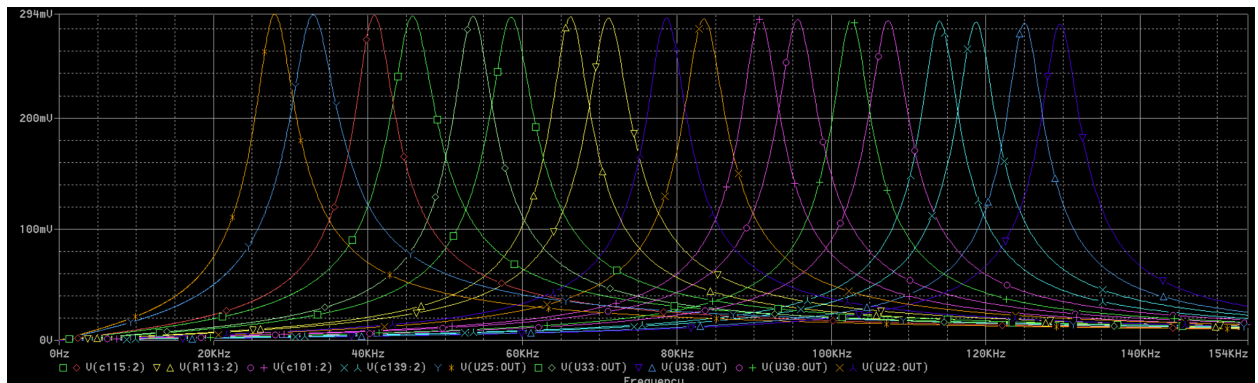


FIGURE 4.16. 9 Nodes, 3k Spacing

Figure 4.17 below shows the result of C_{var} being swept from 10 pF to 80 pF within each of the 9 nodes

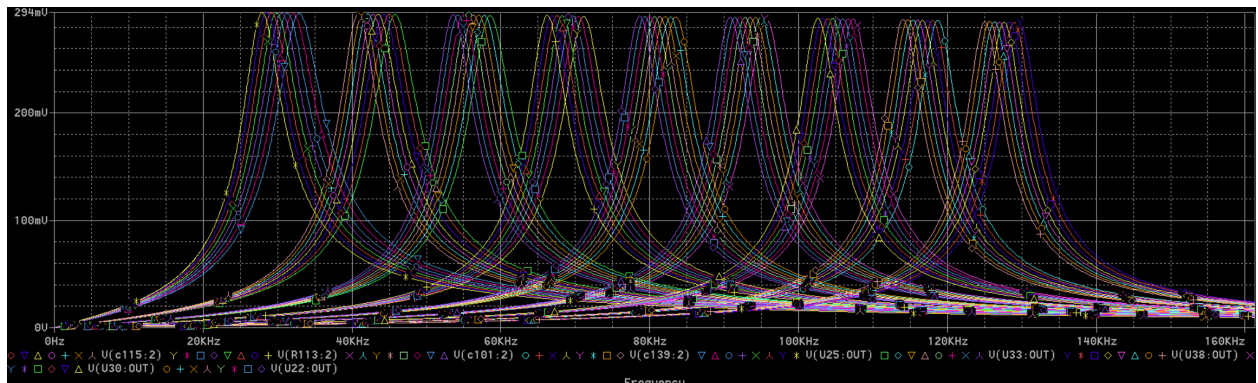


FIGURE 4.17. 9 Nodes All

To gain a closer look at the bandgap sizing, Figure 4.18 is the 9 nodes zoomed into their -3db point



FIGURE 4.18. 9 Nodes Zoomed

This 10 node system analysis considers a tolerance of approximately 5% of the center frequency. Across the operating range of the system, this corresponds to approximately 750 Hz at 15 kHz and 7 kHz at 140 kHz. A practical compromise between these extremes is approximately 3 kHz, suggesting that an appropriate guard band for the system is on the order of 1-3 kHz.

4.4. Limitations

This overlap occurred because the varactor capacitance changes from 10 pF to 80 pF, very small values of C' caused the maximum bandwidth, BW_{max} , to vary significantly from BW_{min} , particularly at Node 1, as shown in Table 4.6. This bandwidth expansion produced overlap in both the 15-node and 13-node systems, since node spacing was determined using BW_{min} rather than the larger and more variable BW_{max} , some nodes still exhibited slight overlap where the resonator bandwidth exceeded the allocated guardband. Future work could instead define channel spacing using BW_{max} to further reduce overlap between adjacent nodes.

Fixed C'	Total C at f_{min}	Total C at f_{max}	BW Variance
5 pF	85 pF	15 pF	5.6× Increase
50 pF	130 pF	60 pF	2.1× Increase
150 pF	230 pF	160 pF	1.4× Increase

TABLE 4.6. Capacitance Variation vs Bandwidth Stability

Overall, simulation results showed that lower operating frequencies, smaller channel ranges, and added guard bands reduce spectral overlap and improve scalability. The maximum number of nodes achieved without significant overlap was 13 nodes, while a cleaner 9-node system with 3 kHz guard bands provided greater tolerance to variation and error.

CHAPTER 5

FPGA-Based Multiplexed Architecture - AM

This chapter investigates two different FPGA-based multiplexing architectures designed to increase sensing density in amplitude-modulated tactile sensing systems. The system supports large tactile arrays by repeating identical sensing segments.

Multiple carrier-generation strategies, including sinusoidal and square-wave excitation methods, are investigated, creating a scalable architecture using heterodyning techniques, summing architectures, and phase-optimization methods.

The system consists of multiple grids of identical sensing nodes, where signal processing techniques are used to differentiate between segments. Let N represent the number of nodes within a segment and S represent the total number of segments in the system

5.1. Solution 1: Sin Waves

5.1.1. Input Generation

. The input to the system consists of N low-frequency sinusoidal carrier signals and S high-frequency carrier signals combined onto a shared wire. Low-frequency carriers identify nodes within a segment, while high-frequency carriers identify the segment itself. For example, in a system with 50 nodes per segment and 10 segments (500 nodes total), the first 50 sinusoids correspond to node resonant frequencies (2 kHz, 4 kHz, 6 kHz, 8 kHz, ..., 100kHz) with 100k being the 50th node. The next 10 high frequency sinusoids correspond to segment frequencies (200 kHz, 300 kHz, 400 kHz, ... 1.1 MHz).

Amplitude modification will take place at the node level, where applied pressure on the sensor changes the amplitude of the resonant frequency associated with that node.

5.1.2. Filtering Segments vs. Nodes

. The combined high frequency and low frequency input is connected to all segments. Each segment and node receives the same composite signal through this line, and selection occurs through frequency-specific filtering. Within each segment, the low-frequency components are distributed across all nodes, where each node contains a filter tuned to its assigned resonant frequency. This allows each node to isolate and respond only to its corresponding low-frequency tone

5.1.3. High Frequency Signals and Heterodyning

. For the high frequency signals, each segment includes a LC band pass filter turned to its assigned segment frequency (e.g., 200 kHz for segment 1 and 300 kHz for segment 2). The resulting segment frequency acts as a local oscillator used to heterodyne the low-frequency node signals within that segment.

This heterodyning process frequency-shifts the node responses by the segment frequency, allowing identical node frequencies to be differentiated across different segments in the frequency domain. As a result, each node is uniquely identified by the combination of its segment frequency and node frequency. For example, Segment 1 Node 2 corresponds to a 4 kHz node tone shifted by a 200 kHz segment tone (204 kHz), while Segment 2 Node 2 corresponds to the same 4 kHz node tone shifted by a 300 kHz segment tone (304 kHz).

5.1.4. Limitations

. The main limitation of this approach is footprint sizing. Currently, each sinusoidal component is generated by an individual oscillator IC, requiring $N+S$ independent oscillator ICs, which would lead to too much complexity with wiring and a severely increased footprint per segment.

5.2. Solution 2: Square Waves

5.2.1. Input generation

. This method uses a signal generator with N total wires to supply a square wave to each node. We started by using an IC generates 3 different square waves, however found that an FPGA would be of a similar size and generate 49 square waves, so we switched to the FPGA. A set of 49 wires then feeds into a segment with an array of 49 nodes

System 2 will work through amplitude modulation of N total nodes within an array of size $\sqrt{N}$ x $\sqrt{N}$. The piezoresistive material is used to scale the amplitude of each individual node using a voltage divider.

5.2.2. Node Signals and Segment Summing

. The excitation signals for all N nodes are generated by an FPGA and distributed as N wires going into each segment, where each wire corresponds to a single node in the array.

For example, in a segment containing 49 nodes, Node 1 would have a center frequency of 100 kHz and Node 2 would have 102 kHz, continuing in 2 kHz increments up to Node 49 at 198 kHz. This first array of nodes will be called Segment 1.

The signals from all nodes within Segment 1 will be added up using a operational amplifier-based summer, producing a singular signal as the output of that segment. This summed output is then multiplied by a segment-specific modulation signal and passed through a bandpass filter spanning from 100 kHz to 198 kHz.

5.2.3. Segment Multiplication

. All segments use identical hardware and receive the same set of node excitation the same frequencies by the FPGA. However, each segment will have a different multiplier, so that each segment's outputs are distinguishable. Segment 1 will be multiplied by 100k while segment 2 will be multiplied by 200k square wave and segment 3 will be multiplied by 300k square wave and so on.

We will have M total segments, each containing N nodes, one summing amplifier, 1 multiplier and one bandpass filter.

5.2.4. Trace Density and Node Density

. Being able to have the fundamental equations for trace density and node density will aid footprint optimization in the future.

The total nodes divided by the total area of all the segments combined gives us the Node Density (ND) for the system:

$$ND = \frac{\text{Total Nodes}}{\text{Total Area}} = \frac{N * M}{M * A_{seg}}$$

Using the Node Density, we find Trace Density ($\frac{T}{A}$) by creating the equation below

$$\frac{T}{A} = \frac{\text{Total Traces}}{\text{Total Area}} = \frac{\text{Total Nodes}}{\text{Total Area}} * \frac{\text{Total Traces}}{\text{Total Nodes}}$$

Since each segment will have N input traces from the FPGA and M output traces coming out of the summing amplifier (Total Traces M + N):

$$\frac{T}{A} = ND * \frac{N + M}{N * M}$$

5.3. Delaying Impulse Trains Code

If the FPGA fires all impulse trains simultaneously, their combined voltage after the summing amplifier will create a massive spike that exceeds the limit of the ADC. To prevent this clipping, we randomize the starting phase (time offset) of each impulse train. Spreading these pulses out minimizes the peak output amplitude —F(t)— over the period (0 ≤ t ≤ T), keeping the combined signal safely within the ADC's measurable range.

We will solve this problem through optimization, by finding a set of phi that minimize the maximum amplitude of the square waves added together. For the equation below, F(t) is the final combined signal voltage, K is the total number of nodes, A is the peak to peak amplitude, f_0 is the node 1 starting frequency which steps up in equal increments of Δf

$$F(t) = \frac{A}{2} \sum_{k=0}^{K-1} \text{sgn}(\sin(2\pi f_k t + \phi_k))$$

$$f_k = f_0 + k \cdot \Delta f$$

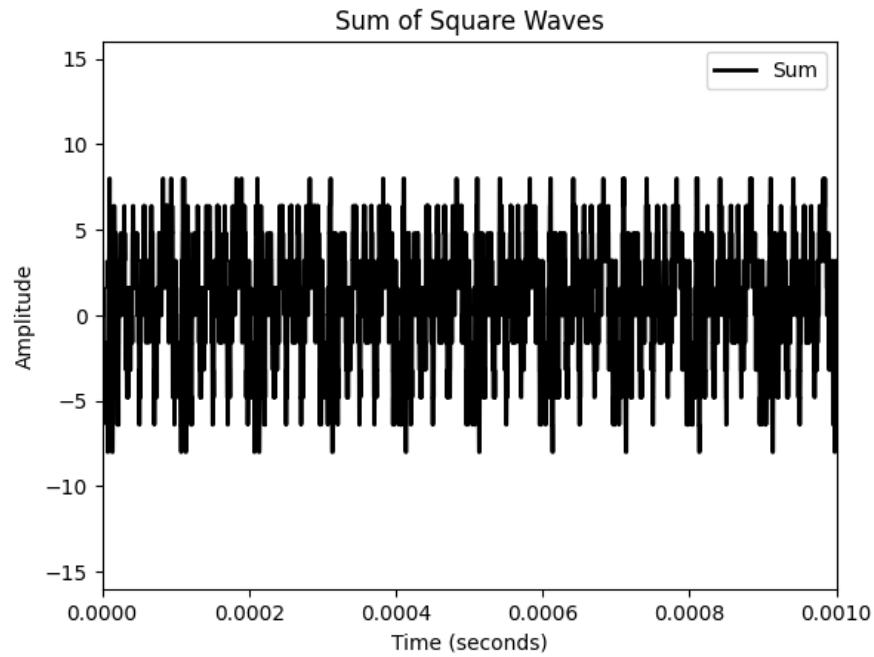
Using the initial conditions:

$f_0 = 100$	starting frequency (Hz)
$\Delta f = 10000$	distance before starting frequency of next segment (Hz)
$A = 1.6$	amplitude of square wave
$K = 20$	number of square waves

Along with the scipy.optimize function:

minimize(cost, phi, args=(f0, delF, A, K), method='CG')

The script solved for a set of phi that should be applied to each node in order to keep the minimal voltage spike to 7.99 V, shown in the results below.



Optimized Phi Array:

```
[0.32472583 3.33859961 3.39691065 4.00509019 4.5621664 6.13145945
 3.24401076 2.02919537 4.99630222 1.70168922 2.75813878 0.49295598
 0.15928342 6.04849838 5.25261801 4.37293491 2.56952713 1.08884033
 0.98292293 1.5723225 ]
```

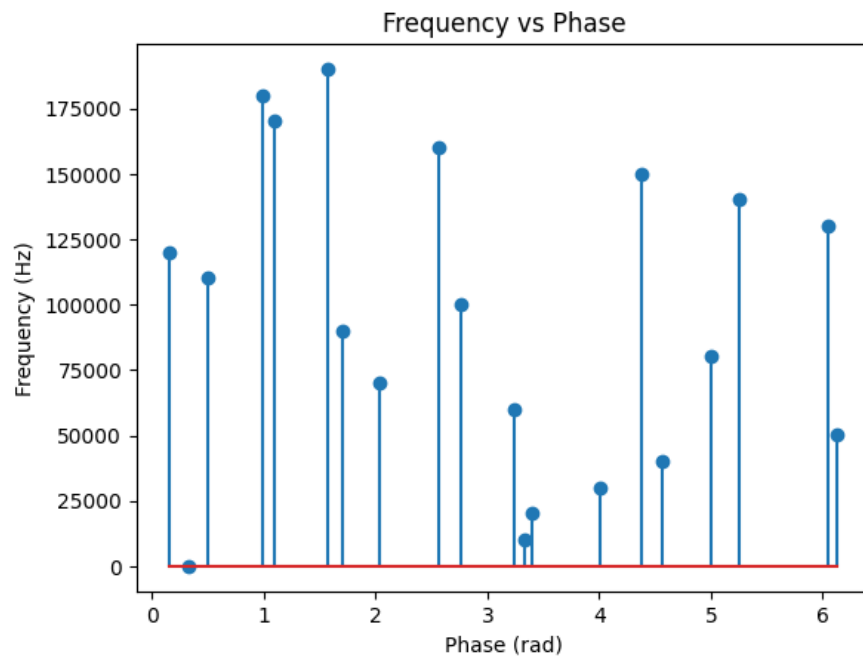


FIGURE 5.1. Phi Optimization Outputs

5.4. Limitations

This system only produces 49 nodes per segment. Additionally, despite having a smaller footprint than the sinusoidal version, the multiplier chip is a similar size to the FPGA, meaning that as we scale the system towards having more segments the multiplier chips will begin to be an issue as they are required in every single segment. The system does however produce more nodes than the other resononant systems.

CHAPTER 6

Conclusion

This thesis investigated scalable tactile sensing architectures utilizing frequency-domain multiplexing techniques to reduce wiring complexity while maximizing achievable sensing density. Three sensing approaches were explored, including a passive RLC resonator architecture using amplitude modulation, an active varactor-based multiple-feedback bandpass filter architecture using frequency modulation, and an FPGA-based multiplexed amplitude modulation system.

The passive RLC architecture demonstrated that resonant tactile sensing can be achieved using only passive components and minimal wiring. However, practical scalability was limited by commercially available inductor and capacitor sizes, achievable bandwidth constraints, and resonant frequency limitations. Analysis showed that minimizing resonator bandwidth is critical for maximizing node density within a fixed spectral range.

The active varactor-based architecture introduced a frequency-modulated sensing approach capable of encoding tactile pressure through resonant frequency shifts. A nonlinear optimization framework was developed to solve for component values while constraining resonant frequency, bandwidth, gain, and channel overlap. Simulation results demonstrated that bandwidth allocation, guard-band spacing, and quality factor growth strongly influence achievable node density. The system was capable of generating densely packed resonant channels while maintaining distinguishable frequency responses between nodes.

Finally, FPGA-based multiplexing techniques were investigated as a method for further increasing sensing density while reducing resonant hardware complexity. Both sinusoidal and square-wave carrier approaches were explored, along with signal summing, heterodyning, and carrier phase optimization methods for reducing ADC saturation. Analytical relationships governing node density

and trace density were additionally developed.

Overall, this work demonstrated that scalable tactile sensing systems are fundamentally constrained by spectral bandwidth allocation, resonator overlap, component limitations, and wiring complexity. Future work includes improving resonator optimization techniques, integrating printable sensing materials, developing custom integrated circuits for resonant channel generation, and exploring adaptive spectral allocation methods for large-scale electronic skin systems used in robotics and human-machine interaction applications.

Architecture	Modulation	Wiring	Main Limitation
Passive RLC	AM	2-wire	Inductor Footprint
Active MFB	FM	2-wire	Q growth / R2 size
FPGA	AM	N-wire segmented	Multipliers / ADC

TABLE 6.1. Comparison of Proposed Tactile Sensing Architectures

APPENDIX A

Code Screenshots

Module 1: $F(t) = \sum_{k=1}^K f_k(t)$

Defining and Plotting Sum of Square Waves

```
import numpy as np
from scipy import signal
import matplotlib.pyplot as plt

def sum_square_waves(phi, f0, delf, A, K, t_end=0.001, samples=200000, plot=False): ## was 0.002 before

    # time vector
    t = np.linspace(0, t_end, samples, endpoint=False)          # time going from 0-1 in 500 increments (500 samples/sec)
                                                                # for begin parameters fmax = 100 + 19(10k) = 190k Hz
                                                                # try 200k/0.002 =100M sampling to avoid alaising

    # initialize sum
    F = np.zeros_like(t)                                         # F = [0,0,0,0,0,...,0] in the same shape as t

    # generate k number of square waves
    for k in range(K):                                           # using lowercase k to increment through
        f_k = f0 + k*delf                                         # increment f based on delf and k
        wave = (A/2) * signal.square(2*np.pi*f_k*t + phi[k])    # amplitude infront of square signal with frequency 2piF
        F += wave                                                 # adding it all together

    if plot:                                                     # include choice to add plot
        plt.plot(t, F, color='black', linewidth=2, label="Sum")
        plt.title("Sum of Square Waves")
        plt.xlabel("Time (seconds)")
        plt.ylabel("Amplitude")
        plt.xlim(0, 0.001)                                       # changed from 0.00005 or 0.01. Messing with this allows for us to sometimes get 9.6 with smaller nums like 0.001, but
        plt.ylim(-(A/2)*K, (A/2)*K)                             # guessing the longest ampint of time is the best despite it only allowing for 11.2 because its the most realistic
        plt.legend()
        plt.show()

    return F
```

FIGURE A.1. PHI OPTIMIZATION CODE PT.1

Module 2: Maximim value of $F(t) = \sum_{k=1}^K f_k(t)$

Defining the cost function using sum_square_waves

```
3) 08 import numpy as np
    from scipy import signal
    import matplotlib.pyplot as plt
    from scipy.optimize import least_squares
    from scipy.optimize import minimize

    # define cost function, max magnitude of the sum
    def cost(phi, f0, delF, A, K):
        F = sum_square_waves(phi, f0, delF, A, K, plot=False) #exclude plot
        return np.max(np.abs(F))
```

Module 3: Optimization

```
4) # Initial Conditions for system
f0=100      # starting frequency (Hz)
delF=10000  # distance before starting frequency of next segment (Hz)
A=1.6      # amplitude of square wave
K=20       # number of square waves

np.random.seed(42) # everytime the optimization is ran, the same phis will be used

initial_phi = 0 # initial phi offset is 0
optimized_max = 999 # Start high so the loop runs

while optimized_max > 8: # if the cost is above 10V (adc capacity) rerun

    initial_phi = np.random.uniform(0, 2*np.pi, K) #np.linspace(0, 2*np.pi, K) #np.zeros(K) # Reset starting point

    # Run minimization optimization
    result = minimize(cost, initial_phi, args=(f0, delF, A, K), method='CG') #CG executes slower but finds minimum in few iterations

    # Check the "Score"
    optimized_phi = result.x
    optimized_max = cost(optimized_phi, f0, delF, A, K)
    print(f"Current Max: {optimized_max}")

# This only runs once the loop breaks (Max < 10)
sum_square_waves(optimized_phi, f0, delF, A, K, plot=True)

print("Optimized Phi Array:")
print(optimized_phi)

frequencies = f0 + np.arange(K) * delF
plt.figure()
plt.stem(optimized_phi % (2 * np.pi), frequencies)
plt.title("Frequency vs Phase")
plt.ylabel("Frequency (Hz)")
plt.xlabel("Phase (rad)")
plt.show()
```

FIGURE A.2. PHI OPTIMIZATION CODE PT.2

Component Selection:

Using a transfer function from the dally-friend bandpass filter we can design a desired frequency and bandwidth by changing the filter's resistor and capacitor values

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal
```

NONLINEAR EQUATION SOLVER iteration loop

```
import numpy as np
from scipy.optimize import least_squares
from math import pi

# USER INPUTS
f_min_start = 144e3
f_max_start = 154e3
bw_min = 30e3
K = 0.296
Cvar_min = 10e-12
Cvar_max = 80e-12
increment = 2 * bw_min/2 + 3e3 # BW_res_min ~ BW_min/2 + BW_max/2, but we cant calculate BW max so using 2x min with a 3e3 guard band
M = 20

x0 = [219.5, .2258, 129.9, 0.41] #[99.76, .0178, 59.06, .998] #new initial node with 5e3 bw #[16.9, 0.102, 10.1, 1] # initial Node 1
fmin = f_min_start
fmax = f_max_start
frange = fmax - fmin
node = 1
current_R2 = 999 # Placeholder for start of loop

def bandpass(f_min, f_max, bw_min, K, Cvar_min, Cvar_max, x0, node_id):
    # Convert to radians
    wc_min_SQ = (f_min**2*pi)**2
    wc_max_SQ = (f_max**2*pi)**2
    wbw_min = 2*pi*bw_min

    def F(x):
        R1, R2, R5, Cprime = x
        # Scaling to make the solvers steps similar (ie change by one unit nF vs kOHM)
        R1 = abs(R1*1e3)
        R2 = abs(R2*1e3)
        R5 = abs(R5*1e3)
        Cprime = Cprime*1e-9

        f1 = (wbw_min - 2 / ((Cprime + Cvar_max) * R5))*1e9
        f2 = (K - R5 / (2 * R1))*1e12
        f3 = (wc_max_SQ - (R1 + R2)/(R1*R2*R5*(Cprime + Cvar_min)**2))
        f4 = (wc_min_SQ - (R1 + R2)/(R1*R2*R5*(Cprime + Cvar_max)**2))
        return [f1, f2, f3, f4]

    # Perform Optimization
    sol = least_squares(F, x0, method = 'lm', ftol=1e-8, xtol=1e-5, gtol=1e-8)

    # After solving converts answers back to ohms and farads
    R1, R2, R5, Cprime = abs(sol.x[0])*1e3, abs(sol.x[1])*1e3, abs(sol.x[2])*1e3, abs(sol.x[3])*1e-9
```

FIGURE A.3. ACTIVE FILTER COMPONENT SELECTION PT.1

```

# Calucute Frequencies from resulting R and C value to check that the solver is correct
f_calc_min = np.sqrt((R1 + R2)/(R1*R2*R5*(Cprime + Cvar_max)**2)) / (2*pi)
f_calc_max = np.sqrt((R1 + R2)/(R1*R2*R5*(Cprime + Cvar_min)**2)) / (2*pi)
bw_calc = 2/((Cprime + Cvar_max)*R5)/(2*pi)
bw_max_calc = 2/((Cprime + Cvar_min)*R5)/(2*pi)
k_calc = R5 / (2*R1)

# residuals and cost
cost = np.sqrt(np.mean(np.array(F(sol.x))**2))

# Output block
print(f"\n===== NODE {node_id} =====")
print(f"Initial x0 used: {[float(round(v, 6)) for v in x0]}")
print(f"Initial f_min={f_min:.2f} Hz, f_max={f_max:.2f} Hz\n")

print("=== New Range Component Values ===")
print(f"R1 = {R1:.3e} Ω")
print(f"R2 = {R2:.3e} Ω")
print(f"R5 = {R5:.3e} Ω")
print(f"C' = {Cprime:.3e} F")

print("\n=== Frequency, Bandwidth & Gain Comparison ===")
# Adjusted widths to accommodate the extra column (12, 12, 14, 14, 10)
print(f"{'Type':<25} {'f_min [Hz]':>12} {'f_max [Hz]':>12} {'BW_min [Hz]':>14} {'BW_max [Hz]':>12} {'K Gain':>10}")
print("-" * 92) # Lengthened line to match new width

# Expected (inputs) - placeholder for BW_max
print(f"{'Expected (Inputs)':<25}"
      f"{'f_min:12.2f'}{'f_max:12.2f'}{'bw_min:14.2f'}{'---':>14}{'K:10.4f'}")

# Actual (computed)
print(f"{'Actual (From R/C)':<25}"
      f"{'f_calc_min:12.2f'}{'f_calc_max:12.2f'}{'bw_calc:14.2f'}{'bw_max_calc:14.2f'}{'k_calc:10.4f'}")

print("-" * 92)

# Compute errors (%)
err_fmin = 100*(f_calc_min - f_min)/f_min
err_fmax = 100*(f_calc_max - f_max)/f_max
err_bw = 100*(bw_calc - bw_min)/bw_min
err_k = 100*(k_calc - K)/K

# Error row - Added an empty space for the BW_max column to keep alignment
print(f"{'Error (%)':<25}"
      f"{'err_fmin:12.4f'}{'err_fmax:12.4f'}{'err_bw:14.4f'}{'':>14}{'err_k:10.4f'}")
print("-" * 92)

# Use new scaled x0 for next node
new_x0 = [R1/1e3, R2/1e3, R5/1e3, Cprime/1e-9] # take away the units for the solvevr
return new_x0, R2 #R2 added to make iterations based on its limits

while current_R2 >= 11: # at 10 ohms we are too close to trace resistance

    x0, current_R2 = bandpass(fmin, fmax, bw_min, K, Cvar_min, Cvar_max, x0, node)

    # Update frequencies for the next potential node
    fmin = fmax + increment
    fmax = fmin + frange
    node += 1

print(f"Loop finished at Node {node-1}. Final R2: {current_R2:.3f} Ω")

```

FIGURE A.4. ACTIVE FILTER COMPONENT SELECTION PT.2

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

Bibliography

- [1] Ariel Slepian, Junjun Chen, and N.v Thakor. Scalable tactile sensing skins: Wiring and data management. *Proceedings of the IEEE*, PP:1–19, 01 2026.
- [2] Long et al. One-wire reconfigurable and damage-tolerant sensor matrix inspired by the auditory tonotopy. *Sci Adv.*, 9(48), 2023.
- [3] Abdul Hakeem Anwer, Nishat Khan, Mohd Zahid Ansari, Sang-Soo Baek, Hoon Yi, Soeun Kim, Seung Man Noh, and Changyoon Jeong. Recent advances in touch sensors for flexible wearable devices. *Sensors*, 22(12), 2022.
- [4] Igor Vehec and Lubomir Livovsky. Flexible resistive sensor based on velostat. *2020 43rd International Spring Seminar on Electronics Technology (ISSE)*, pages 1–6, 2020.
- [5] Zhang et al. Li, Yang. Ultrafast readout, crosstalk suppression iontronic array enabled by frequency-coding architecture. *Nature Electronics*, 8, 2023.
- [6] Davide Deflorio, Massimiliano Di Luca, and Alan M. Wing. Skin and mechanoreceptor contribution to tactile input for perception: A review of simulation models. *Frontiers in Human Neuroscience*, 16, 2022.
- [7] Rahsoft Radio Frequency Certificate. Transfer function, bandwidth, and quality factor in rlc circuits. <https://rahsoft.com/2021/08/06/transfer-function-bandwidth-and-quality-factor-in-rlc-circuits/>, 2021. Accessed: 2026-05-21.
- [8] Ariel Slepian, Sriramana Sankar, and Nitish Thakor. Texture discrimination using a neuromimetic asynchronous flexible tactile sensor array with spatial frequency encoding. *IEEE*, 2021.
- [9] Weidong Yang, Matthew Hon, Haicheng Yao, and Benjamin Tee. An atlas for large-area electronic skins. *Elements in Flexible and Large-Area Electronics*, 2020.
- [10] Zhangping Ji, Hui Zhu, Huicong Liu, Nan Liu, Tao Chen, Zhan Yang, and Lining Sun. The design and characterization of a flexible tactile sensing array for robot skin. *Sensors*, 16(12), 2016.
- [11] Zhenxing Wang and Xuan Dou. A hybrid-frequency sampling tactile sensing system based on a flexible piezoresistive sensor array: Design and dynamic loading validation. *Sensors*, 26(5), 2026.
- [12] Analog Devices. Multiple feedback band-pass design example, Feb. 2012. [Accessed: May 21, 2026].
- [13] Stefan C. B. Mannsfeld, Benjamin C.-K. Tee, Randall M. Stoltenberg, Christopher V. H.-H. Chen, Soumendra Barman, Beinn V. O. Muir, Anatoliy N. Sokolov, Colin Reese, and Zhenan Bao. Highly sensitive flexible pressure sensors with microstructured rubber dielectric layers. *Nature Materials*, 9(10):859–864, Oct. 2010.

- [14] Baochun Xu, Da Chen, Yu Wang, Ruili Tang, Lina Yang, Hui Feng, Yijian Liu, Zhuopeng Wang, Fei Wang, and Tong Zhang. Wireless and flexible tactile sensing array based on an adjustable resonator with machine-learning perception. *Advanced Electronic Materials*, 9, Apr. 2023.
- [15] Z. Zhu, K. Chen, W. Lin, H. Yan, L. Liang, G. Wang, X. Song, C. Zhang, and Z. Wang. Recent advances in spike-based neural coding for tactile perception. *Microsystems & Nanoengineering*, 11(1):212, November 2025.
- [16] Taeyeong Kim, Jaehun Kim, Insang You, Joosung Oh, Sung-Phil Kim, and Unyong Jeong. Dynamic tactility by position-encoded spike spectrum. *Science Robotics*, 7(63), 2022.