SYNTHESIS OF TRANSLINEAR ANALOG SIGNAL PROCESSING SYSTEMS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Eric John McDonald

August 2004

SYNTHESIS OF TRANSLINEAR ANALOG SIGNAL PROCESSING SYSTEMS

Eric John McDonald, Ph.D.

Cornell University 2004

Even in the predominantly digital world of today, analog circuits maintain a significant and necessary role in the way electronic signals are generated and processed. A straightforward method for synthesizing analog circuits would greatly improve the way that analog circuits are currently designed. In this dissertation, I build upon a synthesis methodology for translinear circuits originally introduced by Bradley Minch that uses multiple-input translinear elements (MITEs) as its fundamental building block. Introducing a graphical representation for the way that MITEs are connected, the designer can get a feel for how the equations relate to the physical circuit structure and allows for a visual method for reducing the number of transistors in the final circuit. Having refined some of the synthesis steps, I illustrate the methodology with many examples of static and dynamic MITE networks. For static MITE networks, I present a squaring reciprocal circuit and two versions of a vector magnitude circuit. A first-order log-domain filter and an RMS-to-DC converter are synthesized showing two first-order systems, both linear and non-linear. Higher order systems are illustrated with the synthesis of a second-order log-domain filter and a quadrature oscillator. The resulting circuits from several of these examples are combined to form a phase-locked loop (PLL). I present simulated and experimental results from many of these exam-

ples. Additionally, I present information related to the process of programming the floating-gate charge for the MITEs through the use of Fowler-Nordheim tunneling and hot-electron injection. I also include code for a Perl program that determines the optimum connections to minimize the total number of MITEs for a given circuit.

Biographical Sketch

Having grown up in Pittsburgh, PA, Eric John McDonald remained there to receive his Bachelors of Science degree in Electrical Engineering from the University of Pittsburgh in May, 1998. Continuing his education at Cornell University, he received his Masters of Science degree in Electrical Engineering in May, 2002. He is moving back to Pittsburgh to be closer to friends and family and to find a job working in the engineering industry.

I would like to dedicate this dissertation to my parents, John and Sally McDonald, my brother and his wife, Michael and Heather McDonald, and to my Creator.

Acknowledgements

I would like to express my thanks to my advisor, Bradley Minch. While not only advising me along my path towards completion, he has spent countless hours teaching me both in the classroom and in one on one conversations. Additionally, his previous work provided the foundation upon which I based my dissertation.

I would like to thank Paul Hasler for his assistance in learning the art of programming the mysterious floating-gate transistor.

I would like to thank Rajit Manohar, Mark Heinrich, and Alyssa Apsel for serving on my advising committee.

I would like to thank Livia Gilstrap and Sarah Spence for their advice and guidance on various aspects of completing a graduate degree at Cornell University.

I would like to thank my parents, John and Sally McDonald, who would have supported me down whatever road in life I had chosen. Their example has been a guiding force in my life. I would also like to thank my brother and his wife, Michael and Heather McDonald, who have taught me things that can not be found in any dissertation. Lastly, I would like to express my thanks to the Lord. Without the family, friends, and countless blessings He has provided in my life, I could not have made it this far.

The following work was supported under the NSF Career award CCR-9984625.

Table of Contents

1 Analog Circuit Design **1**
- 1.1 Analog Signal Processing . . . 2
- 1.2 Analog Versus Digital . . . 3
- 1.3 Why Translinear? . . . 6
- 1.4 Multiple-Input Translinear Elements . . . 7
- 1.5 MITE Fundamentals . . . 8
- 1.6 Circuit Synthesis Overview . . . 10

2 Static MITE Networks **11**
- 2.1 Squaring Reciprocal Circuit . . . 12
  - 2.1.1 System Decomposition . . . 12
  - 2.1.2 Translinear Loops . . . 14
  - 2.1.3 Biasing . . . 14
  - 2.1.4 Diode Connections . . . 15
- 2.2 Vector Magnitude . . . 15
  - 2.2.1 System Decomposition . . . 16
  - 2.2.2 Translinear Loops . . . 16
  - 2.2.3 Consolidation . . . 17
  - 2.2.4 Biasing . . . 19
  - 2.2.5 Diode Connections . . . 19
- 2.3 Vector Magnitude with Offsets . . . 20
  - 2.3.1 System Decomposition . . . 20
  - 2.3.2 Translinear Loops . . . 22
  - 2.3.3 Consolidation . . . 23
  - 2.3.4 Biasing . . . 25
  - 2.3.5 Diode Connections . . . 29

3 Linear and Non-Linear First-Order Dynamic MITE Networks **32**
- 3.1 Dynamic MITE Networks . . . 32
- 3.2 First-Order Low-Pass Filter . . . 33
  - 3.2.1 System Decomposition . . . 33
  - 3.2.2 The Inverting Output Structure . . . 33
  - 3.2.3 System Decomposition Continued . . . 36
  - 3.2.4 Translinear Loops . . . 37

3.2.5 Biasing . . . 37
3.2.6 Diode Connections . . . 39
3.3 RMS-to-DC Converter . . . 39
3.3.1 System Decomposition . . . 40
3.3.2 Translinear Loops . . . 41
3.3.3 Consolidation . . . 42
3.3.4 Biasing . . . 45
3.3.5 Diode Connections . . . 45

4 Linear and Non-Linear Second-Order Dynamic MITE Networks 47
4.1 Second-Order Low-Pass Filter . . . 47
4.1.1 System Decomposition . . . 47
4.1.2 Translinear Loops . . . 49
4.1.3 Consolidation . . . 49
4.1.4 Biasing . . . 52
4.1.5 Diode Connections . . . 52
4.2 Quadrature Oscillator . . . 54
4.2.1 System Decomposition . . . 54
4.2.2 Dynamic Constraints . . . 55
4.2.3 Translinear Loops . . . 57
4.2.4 Consolidation . . . 59
4.2.5 Biasing . . . 61
4.2.6 Diode Connections . . . 61

5 Phase-Locked Loop 69
5.1 System Decomposition . . . 69
5.2 Multiplier . . . 70
5.3 Low-Pass Filter . . . 71
5.4 Inter-Network Connections . . . 74

6 Results and Conclusions 79
6.1 Vector Magnitude Results . . . 80
6.2 First-Order Low-Pass Filter Results . . . 80
6.3 RMS-to-DC Converter Results . . . 82
6.4 Second-Order Low-Pass Filter Results . . . 85
6.5 Quadrature Oscillator Results . . . 85
6.6 Phase-Locked Loop Results . . . 91
6.7 Results Summary . . . 91
6.7.1 Mismatch . . . 94
6.7.2 Measurement Errors and Noise . . . 95
6.7.3 Feed Through and Higher Order Effects . . . 96
6.8 Conclusions . . . 97
6.9 Contributions . . . 98

A Programming Floating-Gate Charge **100**
A.1 Programming Overview 100
A.2 Fowler-Norheim Tunneling 101
A.3 Hot-Electron Injection 102
A.4 Programming Method 102
A.5 Derivation 103
A.6 Data Collection 106
A.7 Programming Infrastructure 108

B Circuit Practicalities **111**
B.1 Power Supply 111
B.2 Current Levels 112
B.3 Frequency Limits and Higher Order Effects 112
B.4 General Design and Layout Techniques 112

C Perl Code for Automated Consolidation **114**

Bibliography **121**

List of Figures

1.1 Two possible MITE implementations. (a) An simple k-input MITE realized by a floating-gate PMOS transistor operated in weak inversion. (b) A more practical implementation including a cascode transistor (with bias voltage, V_{cp}) to reduce the Early effect and the gate-drain parasitic. All results are obtained from circuits using this cascoded implementation. 7

2.1 Synthesis of a squaring reciprocal circuit that computes the function $I_z = I_x^2/I_y$. (a) MITE connections according to the inverse of the relationship of powers between I_y and I_x^2. (b) Additional MITE connections according to the inverse of the relationship of powers between I_x^2 and I_z. (c) Biasing the MITEs with the input currents for I_y and I_x. (d) Completing the network by making local diode connections around the I_y and I_x MITEs to generate control gate voltages and force a signal flow to the output MITE passing I_z. . . 13
2.2 Circuit construction for a vector-magnitude circuit. 18
2.3 Initial MITE connections for the radius calculation. 24
2.4 Initial consolidations for the radius calculation. 26
2.5 Final consolidated MITE network for the radius calculation. 27
2.6 Final consolidated MITE network for the radius calculation. 28
2.7 Biasing of the radius calculation network. 30
2.8 Diode connections to complete the radius calculation network. . . . 31

3.1 Inverting output structure used to introduce a dI/dt. 34
3.2 Circuit construction for a first-order low-pass filter. 38
3.3 Initial MITE connections and consolidation for an RMS-to-DC converter. 43
3.4 Biasing and diode connections to complete the RMS-to-DC converter circuit. 44

4.1 Initial MITE connections and consolidation during circuit construction for the second-order low-pass filter. 50
4.2 Biasing and diode connections for the completion of a second-order low pass filter. 53
4.3 Initial connections for the "a" side of the dynamic constraint network. 58

4.4 Initial connections for the "b" side of the dynamic constraint network. 60
4.5 Consolidations for the "a" side of the dynamic constraint network. 62
4.6 Consolidations for the "b" side of the dynamic constraint network. 63
4.7 Rearranged consolidated network for both the "a" and "b" sides. . 64
4.8 Rearranged consolidated network for both the "a" and "b" sides where the voltages, V_a and V_b, are shared from the radius calculation network. 65
4.9 Diode connections for the dynamic constraints. Voltages V_a and V_b represent log-compressed currents and can be used to remove the two input MITEs in the radius calculation network. 67
4.10 Final changes to complete the entire oscillator circuit linking the radius and dynamic sides. 68

5.1 Phase-locked loop block diagram. 69
5.2 Phase-locked loop block diagram. 70
5.3 Construction of the multiplier circuit. (a) Initial connections. (b) Biasing for the circuit. (c) Completed circuit. 72
5.4 Modified low-pass filter from Section 3.2 to include a gain of k. . . 74
5.5 Connecting the output of the oscillator to the second input of the multiplier. 75
5.6 Connecting the output of the oscillator to the second input of the multiplier. 76
5.7 Additional circuit to generate the γ-scaled version of I_τ. 77
5.8 Connecting the output of the filter, I_y, to the oscillator to generate both I_τ and the scaled version, $I_{\gamma\tau}$. 78

6.1 Measured data from the vector magnitude circuit. Measured data is shown with circles and the ideal curves are shown with solid lines. 81
6.2 Frequency response for a first order low-pass filter. 82
6.3 Results from the RMS-to-DC converter circuit with a sinusoidal input signal. 83
6.4 Results from the RMS-to-DC converter circuit with a sawtooth input signal. 84
6.5 Frequency response for a second-order low-pass filter with various quality factors (Q=0.25, 0.5, 1, 2) and an approximate corner frequency of 4kHz. 86
6.6 Frequency response for a second-order low-pass filter with various quality factors (Q=0.25, 0.5, 1, 2) and an approximate corner frequency of 8kHz. 86
6.7 Frequency response for a second-order low-pass filter with various quality factors (Q=0.25, 0.5, 1, 2) and an approximate corner frequency of 11kHz. 87
6.8 Scope capture of the two oscillator outputs at 8.93kHz. 88
6.9 Scope capture of the two oscillator outputs at 40.3kHz. 88

6.10 Scope capture of the two oscillator outputs at 81.7kHz. 89
6.11 Plot of the relationship between the oscillation frequency and I_τ. . 89
6.12 Plot of the two oscillator outputs, I_a versus I_b, showing an approximate phase difference of 87.4 degrees. A perfect circle would be the equivalent of a 90 degrees phase difference. 90
6.13 PLL simulation results. (a) Frequency-controlling current showing the locking behavior. (b) Traces of the input and output signals. . 92
6.14 Experimental PLL results showing the output of the loop filter (which controls the oscillator frequency and is dominated by 60Hz interference) and the output of the oscillator. 93
6.15 Path by which a high frequency input signal could bypass the filter capacitor and show up at the output if the driving source for V_{ref} does not have a low enough output impedance to keep V_{ref} fixed to an effective DC potential. 97

A.1 Plots of injection data used to extra modeling parameters. 107
A.2 Programming infrastructure that allows for a global "erase" through the shared tunneling line, V_{tun}, and individual hot-electron injection through the use of the MITE select signal, Sel_i, control gate bus, CG_{bus}, and drain bus, D_{bus}. (a) MITE cell including two cascode transistors and two transmission gates. (b) Effective MITE cell during the programming phase when both cascode voltages are turned off. 110

Chapter 1

Analog Circuit Design

The recent technology trends for computers and electronics have been focused on pushing digital circuits toward faster clock speeds and smaller channel lengths while still using the analog circuits of yesterday. The lengthy design time required to go from system specification to circuit design has partially contributed to this setback in analog advancements. Even though the analog part of a mixed-signal circuit is generally quite small in comparison to its digital counterpart, it is essential for interfacing with the analog signals of the real world.

With the boom in wireless communications, low power supply and minimal power consumption have become extremely important. Low-power, compact analog circuits could be used to replace their bulky power-hungry digital counterparts if the design time could be reduced. Without a defined method for approaching analog circuit design, each design must be approached starting from scratch or alternately, modifications can be made to an existing circuit if the systems' functionalities match. A circuit synthesis methodology, originally introduced by Minch [20, 21, 24], allows for a straightforward path from a high-level system specification to transistor-level circuit design.

This methodology describes the construction of a class of circuits known as *static and dynamic translinear circuits* [12, 23, 30, 40]. These circuits are able to realize a wide range of systems whose behavior is described by polynomial constraints or algebraic differential equations. The following work is not intended to be a complete tutorial on the entirety of the methodology but rather to expand upon the already published work in this area. Specifically, I focus on some of the more ambiguous aspects of this methodology using many circuit examples to highlight the result of certain design decisions. Refining some of the original synthesis steps, I hope to make the methodology easier to understand and use. Additionally, I present a detailed discussion of issues related to actually realizing circuits in silicon using this methodology. For a more detailed discussion, see [11, 30] for translinear circuits in general and [17] for the basis of this methodology.

1.1 Analog Signal Processing

Whether we are willing to admit it or not, the world is not going digital anytime soon. It is probably true that 99% of the products on our shelves are fundamentally digital. Digital circuits are powerful. Digital signal processing is everywhere and its abilities seem limitless. At some point, the question changes from "Can a digital circuit do this?" to "Should a digital circuit do this?" There is no way of bypassing the fact that the world is analog. The best we can do is take an analog sensor (photo-sensitive transistors, microphones, stress sensors, etc.), run it through an analog anti-aliasing filter, and then feed it to an analog-to-digital converter. In the other direction, the minimum path would include the digital output passing through a digital-to-analog converter, a reconstruction filter, and then going into

whatever output device is required (loudspeaker, monitor, etc.). In either case, the signal either starts as being analog or ends as being analog.

If we have to deal with an analog signal anyway, it might be beneficial to also process that signal in an analog fashion either before we make it digital or instead of processing it digitally at all. In the case of wireless electronics, low-power operation is ultimately the most important feature. A wireless phone that could automatically convert your voice into text and email it to a friend would be worthless if the battery only lasted 15 minutes. Every system designer must be aware of the total power consumption required by the system. If we can perform the same digital operations using a similar analog counterpart and reduce the power consumption, then it is surely worthwhile to explore the possibilities of analog signal processing.

1.2 Analog Versus Digital

In signal processing, as with most things, there is a tradeoff between power consumption and precision. For digital signal processing, increasing precision means adding more bits to the numbers which increases power consumption and complexity but does so in a linear way. For analog signal processing, increasing the precision sometimes means simply increasing the power by increasing the current levels. However, since power is a quadratic function of current level, doubling the current level to get an extra bit's worth of precision quadruples the power consumption. The real advantage of analog systems is that they use the physics of the actual devices to perform the calculations. The advantage of this is that it is possible to perform complex calculations with a relatively few number of tran-

sistors compared to the number of devices a similar digital system would require. This leads to small areas and lower power than the digital counterpart. However, the drawback is that the modeling of the physics of the devices is never exact, thereby limiting the precision of these calculations. Due to the inherent nature of digital circuits, each stage includes a full signal restoration and the only accumulated noise is a result of numerical rounding. Alternately, analog systems have to compete with temperature variations, mismatch, and offsets as well as the inaccuracies of the modeling. All of these sources of error for analog circuits accumulate throughout the entire system.

Power consumption, area (size), precision (noise), and signal frequency are the main characteristics to be examined when considering how to process a signal. Where high precision arithmetic is required, digital signal processing is most likely the better choice. If medium or low precision is all that is needed, then it is possible that analog signal processing may prove to be the more efficient option. Sarpeshkar suggests that analog signal processing is often better in power and area for applications requiring under 10 bits of precision ($\sim$60dB SNR) [38]. However, some more subtle factors play a role in determining which style is best suited for a certain application such as available tools, designer skills, and required time to market. Because there are seemingly endless ways of implementing various systems, deciding whether analog or digital signal processing is most appropriate is not obvious. If it can be accepted that there are some cases when analog signal processing is useful, then it follows that it is worthwhile to research ways of creating such analog systems in a straightforward and efficient manner.

There are many CAD tools on the market today that assist in creating digital circuits from high level circuit descriptions including programming languages like

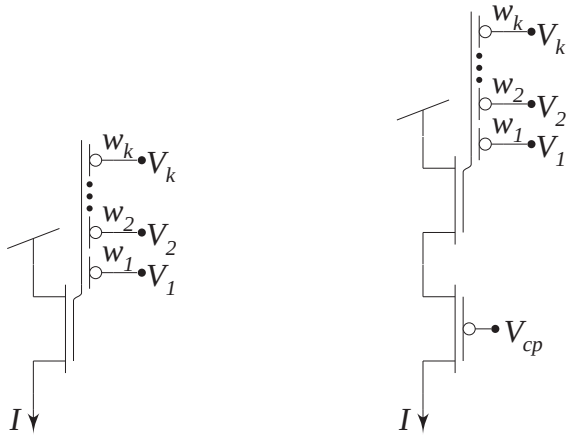
VHDL and reconfigurable devices such as FPGA's that allow for fast prototyping. While current research is being done on various ways to synthesize analog circuits [1,3,10,15,19,21,24,26,29,30,32,37,39,44,46,47], none have been proven to be able to be used for an automated method of circuit synthesis that can produce circuits that perform a wide range of functions. Often times, these techniques will only provide proper sizing and biasing for a fixed circuit topology and are not applicable to a wide range of applications. Alternately, several methods based on a variety of "analog cells" have be developed including a method based on the Bernoulli cell for log-domain filters [3] and one that uses a "tau-cell" to implement arbitrary differential equations [51]. An overview of some automated design techniques that use a cell-based method can be found in [1] and some more recent work in [16,43,47]. In spite of the ongoing work, the currently published synthesis methods are either very limited in what functions they can perform or are too complicated and unclear to be used by the average reader.

Without the aid of a straightforward synthesis methodology, if one wants to design an entire analog system from scratch, he will be investing a great deal of time and energy. The synthesis methodology described in this dissertation is intended to be concise enough to be understood by the majority of readers, allowing them to reduce the amount of time required for analog design and create a solid foundation upon which CAD tools can be designed to further lessen the required work. (Appendix C includes the code for a very rough Perl program that was written to perform part of the synthesis methodology described in this dissertation.)

1.3 Why Translinear?

In 1975, Barrie Gilbert coined the term *translinear* by noting that the *trans*-conductance for a bipolar junction transistor varies *linear*ly with the current. This term also applies to the behavior of a MOSFET when operated in weak inversion or subthreshold. An emerging class of circuits, referred to as *translinear circuits* [12, 23, 30, 40], has been shown to provide a solid foundation for building circuits that can compute a large variety of functions. A subset of this class of circuits, known as *log-domain filters*, has also proven useful for performing various kinds of filtering operations.

Expanding upon this class of circuits, Minch developed another subset of translinear circuits using circuit elements labeled as *multiple-input translinear elements* (MITEs) [17]. MITEs can be implemented in a variety of fashions and lend themselves well to a double-poly process. It is also possible to implement MITEs in any single-poly process as detailed in [25]. When combined to form complex systems, these *MITE networks* are capable of performing numerous functions including any systems defined by algebraic differential equations and polynomial constraints. Minch further went on to develop a structured synthesis methodology for constructing MITE networks. The following pages expand upon this body of work in hopes that it will advance the understanding of how MITE networks are created, provide evidence as to their validity, and inspire further research into their development.



(a) (b)

Figure 1.1: Two possible MITE implementations. (a) An simple k-input MITE realized by a floating-gate PMOS transistor operated in weak inversion. (b) A more practical implementation including a cascode transistor (with bias voltage, V_{cp}) to reduce the Early effect and the gate-drain parasitic. All results are obtained from circuits using this cascoded implementation.

1.4 Multiple-Input Translinear Elements

By limiting circuit construction to identical building blocks, MITEs, a straightforward synthesis methodology for analog circuits has become possible. Figure 1.1 shows two of many implementations of a k-input MITE. For an ideal MITE, the output current, I, is given by

$$I = I_{\mathrm{s}} e^{\kappa(w_1 V_1 + \cdots + w_k V_k)/U_{\mathrm{T}}}, \tag{1.1}$$

where I_{s} is a pre-exponential scaling current, κ accounts for the back-gate effect, V_k is the kth input voltage, w_k is a dimensionless positive weight that scales V_k, and U_{T} is the thermal voltage, kT/q. MITEs can be realized using a variety of transistor configurations [17, 21]. However, to simplify schematics, for the entirety of

this dissertation, I implement MITEs using the non-cascoded floating-gate PMOS transistor, shown in Fig. 1.1(a). For all simulated and experimental results, MITEs are implemented with the cascoded implementation, shown in Fig. 1.1(b). From MITEs, we can build more complex translinear circuits, called static MITE networks [21,22,24] (vector magnitude circuits, squaring-reciprocal circuits, etc.) and dynamic MITE networks (log-domain filters, oscillators, RMS-to-DC converters, etc.) [18,20,21].

1.5 MITE Fundamentals

As mentioned in Section 1.4, we implement MITEs using floating-gate transistors. By connecting several capacitors to the floating-gate of a transistor, we gain the ability to have multiple controlling voltages. The effective floating-gate voltage can be calculated as the weighted sum of the control gate voltages. The weight of each control gate voltage is given by the ratio of that control gate capacitance to the total capacitance at the floating-gate, i.e.,

$$w_i = \frac{C_i}{\sum_{i=1}^{k} C_i + C_{\text{parasitic}}} = \frac{C_i}{C_{\text{total}}} \tag{1.2}$$

Since we ultimately want to connect multiple MITEs together, we would like these control gate weights to be equal across all MITEs. With good layout techniques and adequately sized transistors and capacitors to make mismatch negligible (area of capacitors $\geq 100\lambda^2$, W/L of transistors $\geq 20/4$), we can assume that the parasitic capacitance will be approximately equal for all MITEs. Using unit-sized control gate capacitors with an equal number per MITE will both swamp out variations in parasitics and create uniform weights. With these requirements, Eq. 1.2 simplifies

to

$$w = \frac{C_{\text{cg}}}{kC_{\text{cg}} + C_{\text{parasitic}}} = \frac{C_{\text{cg}}}{C_{\text{total}}}\,, \tag{1.3}$$

where C_{cg} is the capacitance for a unit-sized control gate thereby creating uniform weights for every control gate.

As shown in Fig. 1.1(a), a single floating-gate transistor could be used for MITE implementation. However, the gate-drain overlap capacitance causes this implementation to have an unacceptable performance. With floating-gate transistors, the drain voltage can be thought of as an additional controlling voltage where the gate-drain overlap capacitance determines its weight. We can remove almost all dependency on the drain voltage by using a cascode transistor, as shown in Fig. 1.1(b). This configuration also has the positive effect of drastically reducing the Early effect.

With the drain voltage's influence on the floating-gate voltage effectively removed, the floating-gate voltage can be calculated as the weighted sum of the control gate voltages plus the charge trapped on the floating-gate itself.

$$V_{\text{fg}} = \sum_{i=1}^{k} w_i V_i + \frac{Q}{C_{\text{total}}} \tag{1.4}$$

(Since the drain voltage variance should be small and the gate-drain overlap capacitance is nearly constant over the operating range, the drain's influence on the floating-gate voltage can be approximated as being constant and therefore, can be thought of as being lumped in with the trapped charge, Q.)

We can derive the equation for the MITE drain current in Eq. 1.1 by substituting the expression for V_{fg} into the relationship between drain current and gate voltage for a subthreshold MOS transistor,

$$I_{\text{d}} = I_{\text{o}} e^{(\kappa V_{\text{g}}/U_{\text{T}})}\,. \tag{1.5}$$

This substitution results in

$$I_{\mathrm{d}} = I_{\mathrm{o}} e^{\kappa\left(\sum_{i=1}^{k} w_i V_i + \frac{Q}{C_{\mathrm{total}}}\right)/U_{\mathrm{T}}}, \tag{1.6}$$

which can be rearranged to find the MITE current expression from Eq. 1.1 by grouping the floating-gate charge, Q, into the pre-exponential scaling factor, I_{s},

$$I_{\mathrm{d}} = \underbrace{I_{\mathrm{o}} e^{\kappa Q/C_{\mathrm{total}}U_{\mathrm{T}}}}_{I_{\mathrm{s}}} e^{\kappa\left(\sum_{i=1}^{k} w_i V_i\right)/U_{\mathrm{T}}}. \tag{1.7}$$

Note that the trapped charge on each control gate, Q, is not uniform across all MITEs initially and must be adjusted so that each MITE has the same value of I_s. Methods of *programming* the floating-gate charge are addressed in Appendix A.6.

1.6 Circuit Synthesis Overview

The following chapters detail the specifics of circuit synthesis for various types of systems and progress in increasing complexity. The synthesis methodology is summarized by the following overview. First, high-level system descriptions are broken down into equations of polynomial constraints and first-order differential equations. The dimensionless variables are replaced by ratios of currents. Any time derivatives are replaced with a product of currents according to an output structure primitive. These equations of currents are arranged into translinear loop (TL) equations and Kirchhoff Current Law (KCL) equations. The TL equations are used to generate connections between MITEs. Very often, several MITEs are determined to be redundant and can be removed through a process called *consolidation* [17]. Once the MITEs are biased with current sources and the constraints in any KCL equations, they are locally diode connected to force a signal flow and generate the proper control gate voltages, completing the circuit.

Chapter 2

Static MITE Networks

Due to the exponential relationship between the drain current and the control gate voltages, MITE networks are ideal for many system implementations. This exponential relationship coupled with the weighted summation at the floating-gate allows for the easy calculation of products of currents raised to various powers. Summations are computed by simply summing currents through Kirchhoff's Current Law (KCL).

The term *static MITE networks* refers to MITE networks whose high-level description does not include a dependency on time. In other words, the output is dependent upon the inputs to the network only and does not retain any kind of "state". Examples of static MITE networks are the squaring reciprocal circuit described by

$$I_{\text{out}} = \frac{I_x^2}{I_y} \tag{2.1}$$

and the vector magnitude circuit described by

$$I_{\text{out}} = \sqrt{I_x^2 + I_y^2}\,. \tag{2.2}$$

Sections 2.1, 2.2, and 2.3 outline the steps necessary to synthesize several example

static MITE networks.

2.1 Squaring Reciprocal Circuit

Our first example network will compute the function,

$$z = \frac{x^2}{y}\,, \tag{2.3}$$

where z is the output given by the square of x divided by y. The first step is to decompose the high-level description into a collection of translinear loop (TL) and Kirchhoff Current Law (KCL) equations.

2.1.1 System Decomposition

We replace the dimensionless variables, x, y, and z, by making substitutions of ratios of currents. We do so by defining a constant *unit current*, I_1, that represents the number 1. Making three definitions,

$$x = \frac{I_x}{I_1}\,, \quad y = \frac{I_y}{I_1}\,, \quad \text{and} \quad z = \frac{I_z}{I_1}\,, \tag{2.4}$$

we can replace the original system description with

$$\frac{I_z}{I_1} = \left(\frac{I_x}{I_1}\right)^2 \frac{I_1}{I_y}\,. \tag{2.5}$$

Multiplying through by I_1, we can simplify Eq. 2.5 to

$$I_z = \frac{I_x^2}{I_y}\,. \tag{2.6}$$

It is worthwhile to note that very often the unit currents will cancel out (as in this example). However, this cancellation does not always occur and therefore this step is strongly recommended for each decomposition. Next, we rearrange Eq. 2.6 to

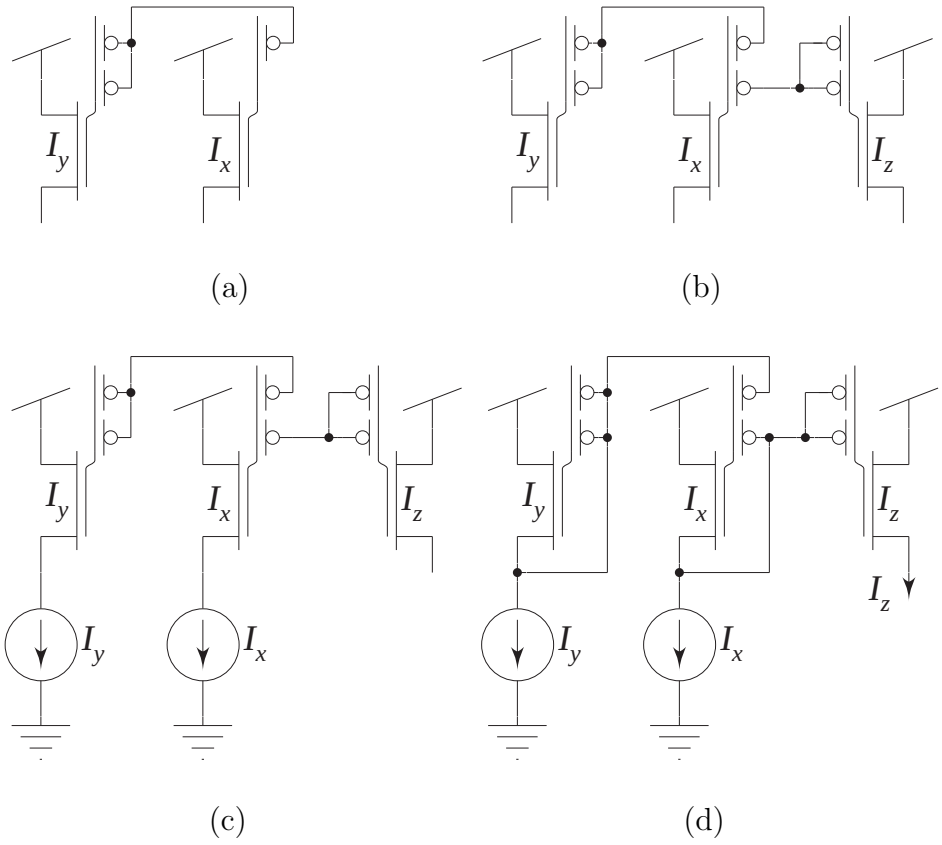


Figure 2.1: Synthesis of a squaring reciprocal circuit that computes the function $I_z = I_x^2/I_y$. (a) MITE connections according to the inverse of the relationship of powers between I_y and I_x^2. (b) Additional MITE connections according to the inverse of the relationship of powers between I_x^2 and I_z. (c) Biasing the MITEs with the input currents for I_y and I_x. (d) Completing the network by making local diode connections around the I_y and I_x MITEs to generate control gate voltages and force a signal flow to the output MITE passing I_z.

remove any quotients, finding a single translinear loop equation (no KCL equations in this example),

$$I_z I_y = I_x^2 \,. \tag{2.7}$$

2.1.2 Translinear Loops

MITE connections are made in a similar fashion to the clockwise/counter-clockwise method of traditional translinear circuit synthesis. For MITEs, connections are made from odd currents (left-hand side) to even currents (right-hand side). The only choices available for this simple circuit are connections from I_z to I_x and from I_y to I_x. Considering the connection from I_y to I_x, we connect the control gates of two MITEs according to the *inverse* of the ratio of their powers. In this case, we connect two control gates from an I_y MITE to one control gate of an I_x MITE, as shown in Fig. 2.1(a). A connection from one control gate of the I_x MITE is then made to two control gates of the I_z MITE according to the relationship between I_x^2 and I_z. This last connection is shown in Fig. 2.1(b).

2.1.3 Biasing

Once all connections have been made, we need to bias the MITEs. Biasing can be completed by either adding current sources for inputs, making connections according to any KCL equations, or adding NMOS current mirrors. The convention of labelling the expected MITE current at the transistor has been adopted in order to eliminate confusion during the biasing stage. This example requires only two biasing current sources for the two inputs, I_x and I_y. The biased circuit is shown in Fig. 2.1(c).

2.1.4 Diode Connections

Looking at the circuit in Fig. 2.1(c), it is obvious that nothing is driving the capacitors connecting the MITEs. In order to force these control gate voltages to the appropriate potentials such that each MITE passes the expected current, we make local connections from the drains to the control gates. These kind of connections are referred to as *diode connections* since they give the MITE a behavior similar to that of a diode. (An NMOS transistor with the gate and drain tied together becomes very similar to a diode.) These local feed-back connections ensure that the MITEs will pass the biasing currents. Since the output MITEs are not biased, diode connections are not made around them. There is only one possible diode connection scheme for this circuit, diode connecting around the two input MITEs (I_x and I_y). The completed circuit is shown in Fig. 2.1(d).

2.2 Vector Magnitude

Suppose that we need a circuit to compute the magnitude of a two-dimensional vector, $[x, y]$, where we take x and y to be strictly positive. The magnitude can be computed as the square root of the sum of the squares,

$$r = \sqrt{x^2 + y^2} \quad . \tag{2.8}$$

One possible solution would be to use two squaring circuits whose output currents are summed at a KCL node. These summed currents can then be used as the input to a square-rooting circuit. While this straightforward method will work, we can address this problem as a complete system resulting in a more efficient design.

2.2.1 System Decomposition

We begin by representing the input and output signals by current ratios,

$$r = \frac{I_r}{I_1}\,, \qquad x = \frac{I_x}{I_1}\,, \qquad \text{and} \qquad y = \frac{I_y}{I_1} \quad . \tag{2.9}$$

By substituting these representations into Eq. 2.8, we find that

$$\frac{I_r}{I_1} = \sqrt{\left(\frac{I_x}{I_1}\right)^2 + \left(\frac{I_y}{I_1}\right)^2}\,, \tag{2.10}$$

which can easily be arranged to obtain

$$\left(\frac{I_r}{I_1}\right)^2 = \left(\frac{I_x}{I_1}\right)^2 + \left(\frac{I_y}{I_1}\right)^2 \,. \tag{2.11}$$

Multiplying through by I_1^2 removes all dependency on I_1 resulting in

$$I_r^2 = I_x^2 + I_y^2\,. \tag{2.12}$$

Dividing through by I_r in order to get a representation of the output current to the first power,

$$I_r = \underbrace{\frac{I_x^2}{I_r}}_{I_{r1}} + \underbrace{\frac{I_y^2}{I_r}}_{I_{r2}}\,, \tag{2.13}$$

allows us to find the following KCL equation and two TLP equations:

$$\begin{aligned} \text{KCL}: &\quad I_r = I_{r1} + I_{r2} \\ \text{TL}: &\quad I_{r1} I_r = I_x^2 \qquad I_{r2} I_r = I_y^2\,. \end{aligned} \tag{2.14}$$

2.2.2 Translinear Loops

Every circuit construction begins with the TL equations. In this case, we examine the relationships of the powers of the currents in Eq. 2.14. Noting that these two TL equations are of the same form as the equation for the squaring reciprocal circuit of Section 2.1, we can make the same control gate connections (repeated here for

clarity). Because I_x and I_y are raised to the second power, their connections to the other MITEs must be in a relationship of one to two. Specifically, the ratio of connections between any two alternating currents (currents on opposite sides of the equation) will be the opposite of the ratio of their powers. To list the connections more succinctly, the TL equations can be rearranged into an alternating pattern that more clearly represents the MITE connections.

Currents	Power ratios	Connection ratios
$I_r^1 \leftrightarrow I_x^2 \leftrightarrow I_{r1}^1$	1:2:1	2:1:2
$I_r^1 \leftrightarrow I_y^2 \leftrightarrow I_{r2}^1$	1:2:1	2:1:2

Due to the circuit's symmetry, we chose to draw the MITEs in a symmetric fashion by placing the I_r MITEs on the outside, as shown in Fig. 2.2(a).

2.2.3 Consolidation

Once the MITEs have been drawn with the proper connections, it is sometimes possible to examine the circuit to remove redundant components. In this example, the I_r MITEs on the ends in Fig. 2.2(a) are identical. Since both of their control gates are tied together and they are both passing the same current, I_r, then the voltages on the control gates must be equal. Therefore, we can remove the MITE on the right end and use only the one on the left, as shown in Fig. 2.2(b). This can be seen more clearly by examining the current-voltage relationship of a MITE (with two control gates tied together),

$$I_{\mathrm{d}} = I_{\mathrm{s}} e^{\kappa(2wV_{cg})/U_{\mathrm{T}}} \,. \tag{2.15}$$

Because I_d and V_{cg} are the only varying terms, if the I_d's are equal, then the V_{cg}'s must also be equal. A more visual method for consolidation is presented in Section 2.3.

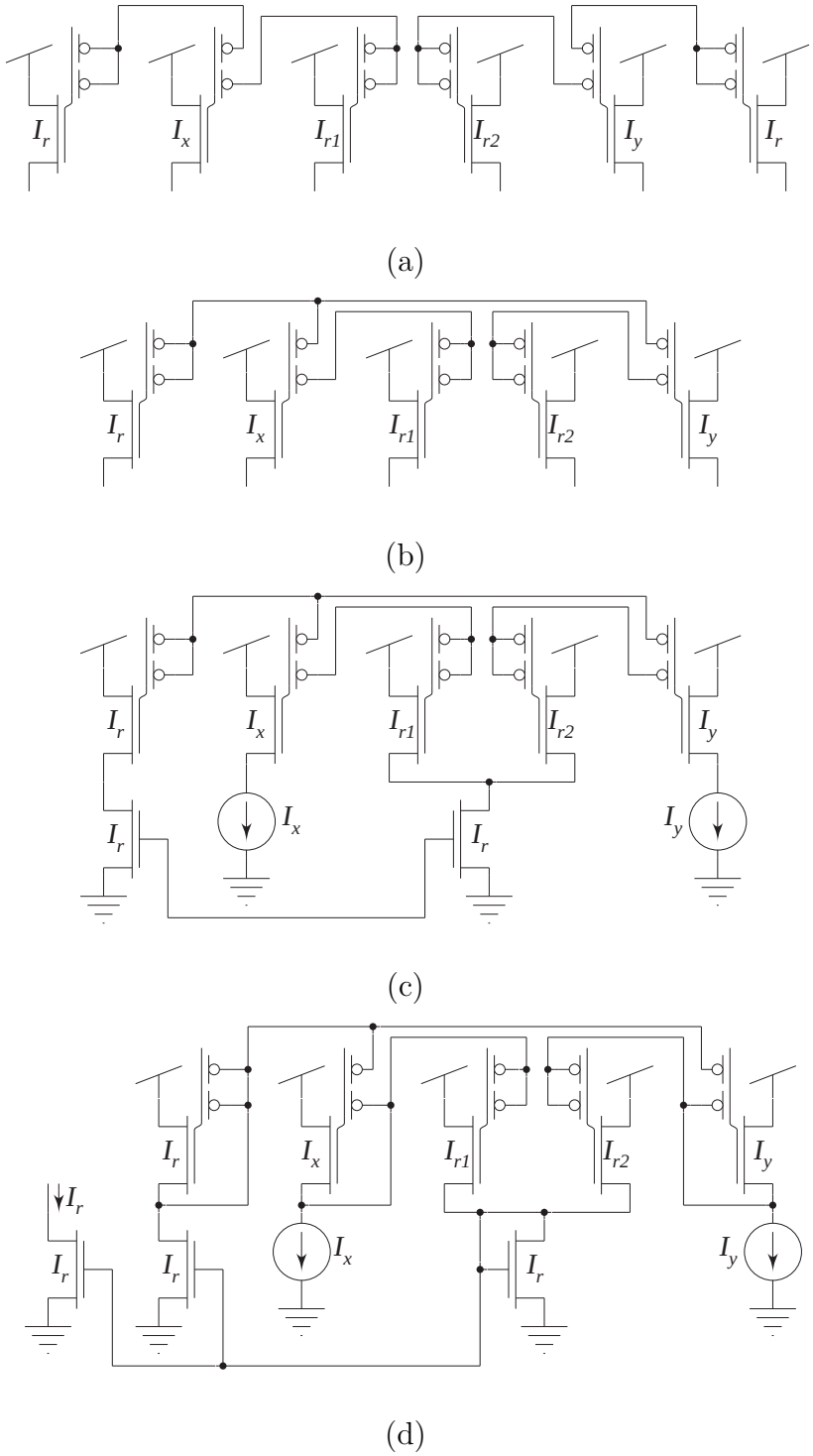


Figure 2.2: Circuit construction for a vector-magnitude circuit.

2.2.4 Biasing

To bias the circuit in Fig. 2.2(b), we begin by adding current sources to the drains of the I_x and I_y MITEs since these are the inputs. The I_{r1} and I_{r2} MITEs are biased through the use of the KCL equation in Eq. 2.14 by tying their drains together and connecting those to an NMOS transistor that is passing I_r. Since I_r is the output of this circuit, we must use an NMOS current mirror to sink I_r for the KCL constraint. The output MITE (passing I_r) is similarly biased with an NMOS transistor sinking I_r creating the other half of the current mirror. At this point, it does not matter which direction the current mirror is going. The direction of the current mirror is determined when the diode connections are made. Figure 2.2(c) shows the appropriate biasing additions. Simple NMOS transistors are shown in all schematics to keep them compact. For all simulated and experimental results, all NMOS transistors are cascoded to reduce gain error due to the Early effect.

2.2.5 Diode Connections

Diode connections must be made to force the gates (for NMOS transistors) and control gates (for MITEs) to the proper voltages. Starting with the input MITEs (I_x and I_y), we diode connect from the drains to the first control gate for each. The I_r MITE is then diode connected leaving only the NMOS transistor below the I_{r1} and I_{r2} MITEs (the KCL node) available for diode connection. The completed circuit is shown in Fig. 2.2(d). Note that another NMOS transistor is shown to provide a mirrored copy of I_r as an output. It would also be possible to use a MITE to mirror I_r as an output should a current source be required instead of a current sink. Experimental results for this circuit can be found in Section 6.1.

It is possible to choose a different diode connection scheme. The behavior of all

valid schemes will still be the same to the first order. However, second order effects will cause varying performance (particularly at higher frequencies). Analyzing higher order effects for translinear circuits is an ambitious task and beyond the scope of this dissertation (even more so for MITE networks whose signal flow is primarily through capacitively coupled nodes). Limited work has been done in the analysis of higher order effects in log-domain filters by Leung [14] and Frey [10].

2.3 Vector Magnitude with Offsets

Thus far, I have proceeded with the unmentioned assumption that all currents are positive (as is required for MITE networks). Reconsidering the vector magnitude function of Section 2.2,

$$r = \sqrt{x^2 + y^2}\,, \tag{2.16}$$

we observe that r will always be positive due to the squaring functions on x and y. However, x and y could take on negative values. In order to ensure strictly positive currents, we can introduce an offset to both x and y,

$$a = x + f \quad \text{and} \quad b = y + f\,. \tag{2.17}$$

Squaring both sides of Eq. 2.16 and inserting these new expressions for x and y, we find

$$\begin{aligned} r^2 &= x^2 + y^2 = (a - f)^2 + (b - f)^2 \\ &= a^2 + b^2 + 2f^2 - 2fa - 2fb \end{aligned} \tag{2.18}$$

2.3.1 System Decomposition

Examining Eq. 2.18, we see that the factor, $2f$, appears in three of the five terms on the right-hand side. Recognizing that by lumping the 2 and the f together when

the current ratios are introduced, the three terms containing the $2f$ factor will only contain two terms instead of three (i.e. $I_{2f}I_a$ instead of $I_2I_fI_a$). Sometimes it can prove beneficial to leave dimensionless numbers in the equations until later in the decomposition process, as demonstrated in the oscillator circuit of Section 4.2. Determining whether making such a grouping simplifies the resulting circuit is often difficult to see in advance and is usually determined only after trying several different decompositions.

Defining I_{2f} as $2I_f$, we introduce current ratios and solve for I_r, obtaining

$$\left(\frac{I_r}{I_1}\right)^2 = \left(\frac{I_a}{I_1}\right)^2 + \left(\frac{I_b}{I_1}\right)^2 + \frac{I_{2f}}{I_1}\frac{I_f}{I_1} - \frac{I_{2f}}{I_1}\frac{I_a}{I_1} - \frac{I_{2f}}{I_1}\frac{I_b}{I_1}\,. \tag{2.19}$$

Finding that every I_1 cancels out, the result simplifies to

$$I_r^2 = I_a^2 + I_b^2 + I_{2f}I_f - I_{2f}I_a - I_{2f}I_b\,. \tag{2.20}$$

Dividing both sides by I_r, we obtain

$$I_r = \underbrace{\frac{I_a^2}{I_r}}_{I_{r1}} + \underbrace{\frac{I_b^2}{I_r}}_{I_{r2}} + \underbrace{\frac{I_{2f}I_f}{I_r}}_{I_{r3}} - \underbrace{\frac{I_{2f}I_a}{I_r}}_{I_{r4}} - \underbrace{\frac{I_{2f}I_b}{I_r}}_{I_{r5}}\,. \tag{2.21}$$

By introducting five intermediate currents, we reduce this constraint to five TL equations and one KCL equation:

$$\begin{aligned} \text{KCL}: &\quad I_r = I_{r1} + I_{r2} + I_{r3} - I_{r4} - I_{r5} \\ \text{TL}: &\quad I_{r1}I_r = I_aI_a \quad I_{r2}I_r = I_bI_b \quad I_{r3}I_r = I_{2f}I_f \\ &\quad I_{r4}I_r = I_{2f}I_a \quad I_{r5}I_r = I_{2f}I_b\,. \end{aligned} \tag{2.22}$$

For complex networks, I have found that limiting MITEs to two control gates and only one to one connections simplifies the synthesis process allowing for easier consolidation. This also has the added benefit of removing several degrees of freedom making the automation of this synthesis methodology easier to implement. (Appendix C includes a Perl program that takes advantage of the two control gate limit

and finds the best connection scheme in order to maximize consolidation resulting in the minimum number of required MITEs.) Limiting MITEs to two control gates, we represent any currents raised to a power other than one as a repeated product, as shown in the first two TL equations in Eq. 2.22. This restriction also allows a rewording of the original Translinear Loop Principal to apply to static MITE networks:

> Following the connections of control gates through a static MITE network limited to one to one connections and two control gates per MITE, the product of the currents for even MITEs is equal to the product of currents for odd MITEs when the starting and ending control gates are at the same potential.

2.3.2 Translinear Loops

Since we have limited MITEs to only two control gates each and one to one connections only, any currents raised to powers greater than one have been repeated (i.e., I_a^2 becomes $I_a I_a$). The first TL equation, $I_{r1} I_r = I_a I_a$, can be arranged in one to one connections as

$$I_{r1} \leftrightarrow I_a \leftrightarrow I_r \leftrightarrow I_a \,. \tag{2.23}$$

This configuration leaves the end MITEs (I_{r1} and I_a) without a connection to their second control gate. According to the balancing theorem [21], we can connect these unused control gates to a DC reference voltage, labeled V_{ref} in Fig. 2.3. Examining

the five TL equations,

$$\begin{aligned} I_{r1}I_r &= I_aI_a \\ I_{r2}I_r &= I_bI_b \\ I_{r3}I_r &= I_{2f}I_f \\ I_{r4}I_r &= I_{2f}I_a \\ I_{r4}I_r &= I_{2f}I_b\,, \end{aligned} \tag{2.24}$$

we see that the similarities will provide us with opportunities to consolidate. We arrange the above equations into the following odd-even pairings:

$$\begin{aligned} I_r &\leftrightarrow I_a \leftrightarrow I_{r1} \leftrightarrow I_a \\ I_r &\leftrightarrow I_b \leftrightarrow I_{r2} \leftrightarrow I_b \\ I_r &\leftrightarrow I_{2f} \leftrightarrow I_{r3} \leftrightarrow I_f \\ I_r &\leftrightarrow I_{2f} \leftrightarrow I_{r4} \leftrightarrow I_a \\ I_r &\leftrightarrow I_{2f} \leftrightarrow I_{r5} \leftrightarrow I_b\,. \end{aligned}$$

Figure 2.3 shows the layout of MITEs with these connections.

2.3.3 Consolidation

Looking at the MITE arrangement in Fig. 2.3, we see that the I_r MITEs on the left can all be shared and that the I_a MITEs along with the I_b MITEs on the right ends can be shared. When limited to the two control gate structure, opportunities to consolidate can be seen by observing the order of currents as they accumulate from the edges and proceed inward. Any time that two or more rows of currents contain the same ordering on either end, they can be shared, as indicated below for I_r, I_a, and I_b:

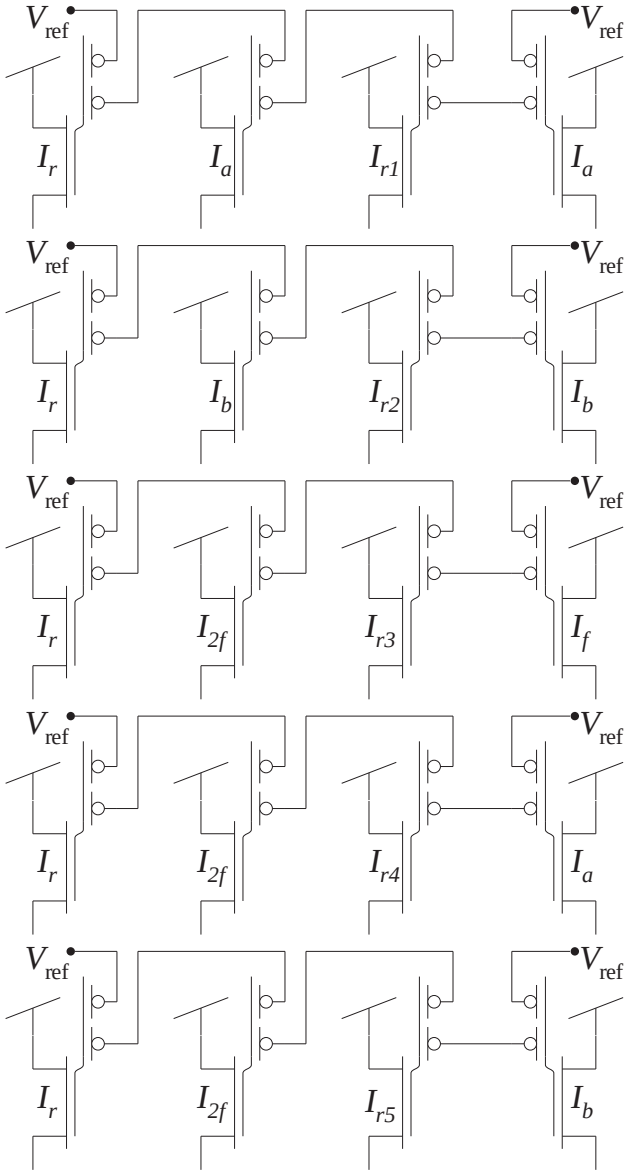


Figure 2.3: Initial MITE connections for the radius calculation.

$$\boldsymbol{I_r} \leftrightarrow I_a \leftrightarrow I_{r1} \leftrightarrow I_a \qquad I_r \leftrightarrow I_a \leftrightarrow I_{r1} \leftrightarrow \boldsymbol{I_a}$$
$$\boldsymbol{I_r} \leftrightarrow I_b \leftrightarrow I_{r2} \leftrightarrow I_b \qquad I_r \leftrightarrow I_b \leftrightarrow I_{r2} \leftrightarrow \boldsymbol{I_b}$$
$$\boldsymbol{I_r} \leftrightarrow I_{2f} \leftrightarrow I_{r3} \leftrightarrow I_f \qquad I_r \leftrightarrow I_{2f} \leftrightarrow I_{r3} \leftrightarrow I_f$$
$$\boldsymbol{I_r} \leftrightarrow I_{2f} \leftrightarrow I_{r4} \leftrightarrow I_a \qquad I_r \leftrightarrow I_{2f} \leftrightarrow I_{r4} \leftrightarrow \boldsymbol{I_a}$$
$$\boldsymbol{I_r} \leftrightarrow I_{2f} \leftrightarrow I_{r5} \leftrightarrow I_b \qquad I_r \leftrightarrow I_{2f} \leftrightarrow I_{r5} \leftrightarrow \boldsymbol{I_b}\,.$$

The removal of the redundant MITEs is shown in Fig. 2.4. Looking towards the insides from the left side, we see that several of the I_{2f}'s can be shared as highlighted below.

$$I_r \leftrightarrow I_a \leftrightarrow I_{r1} \leftrightarrow I_a$$
$$I_r \leftrightarrow I_b \leftrightarrow I_{r2} \leftrightarrow I_b$$
$$\boldsymbol{I_r} \leftrightarrow \boldsymbol{I_{2f}} \leftrightarrow I_{r3} \leftrightarrow I_f$$
$$\boldsymbol{I_r} \leftrightarrow \boldsymbol{I_{2f}} \leftrightarrow I_{r4} \leftrightarrow I_a$$
$$\boldsymbol{I_r} \leftrightarrow \boldsymbol{I_{2f}} \leftrightarrow I_{r5} \leftrightarrow I_b$$

The final MITE network has been reduced to 12 MITEs from the original 20. The currents highlighted below indicate the MITEs that remain after all consolidations have been completed. The reduced MITE network is shown in Fig. 2.5.

$$\boldsymbol{I_r} \leftrightarrow \boldsymbol{I_a} \leftrightarrow \boldsymbol{I_{r1}} \leftrightarrow \boldsymbol{I_a}$$
$$I_r \leftrightarrow \boldsymbol{I_b} \leftrightarrow \boldsymbol{I_{r2}} \leftrightarrow \boldsymbol{I_b}$$
$$I_r \leftrightarrow \boldsymbol{I_{2f}} \leftrightarrow \boldsymbol{I_{r3}} \leftrightarrow \boldsymbol{I_f}$$
$$I_r \leftrightarrow I_{2f} \leftrightarrow \boldsymbol{I_{r4}} \leftrightarrow I_a$$
$$I_r \leftrightarrow I_{2f} \leftrightarrow \boldsymbol{I_{r5}} \leftrightarrow I_b$$

2.3.4 Biasing

The MITEs from Fig. 2.5 have been rearranged into an array, shown in Fig. 2.6. Biasing for this circuit is very straightforward. Placing current sources for all of

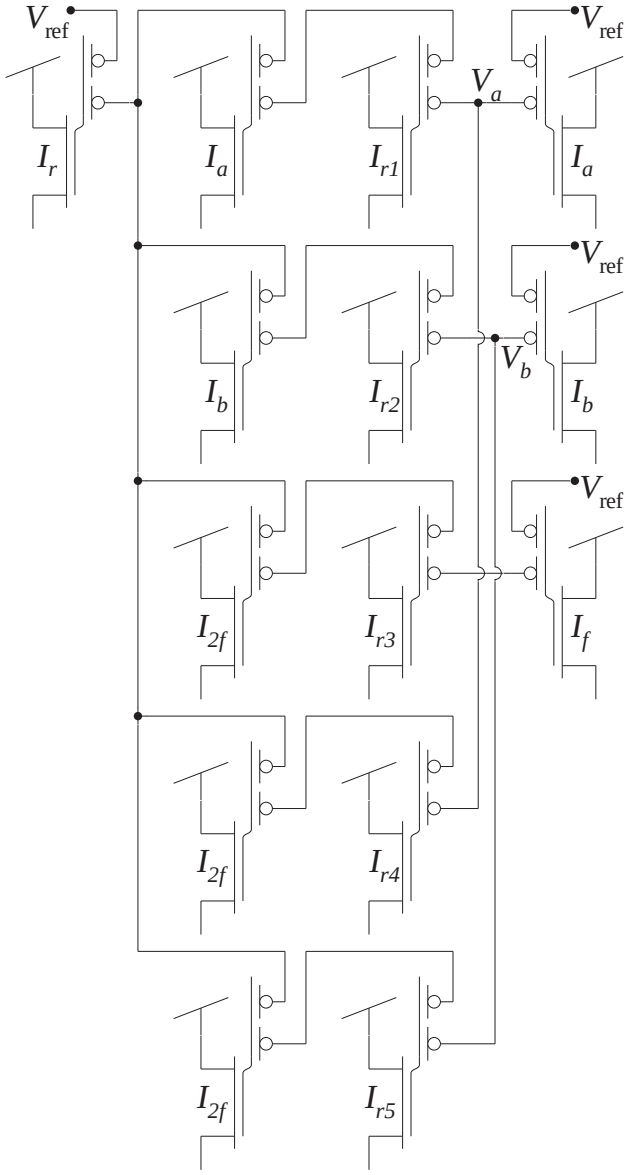


Figure 2.4: Initial consolidations for the radius calculation.

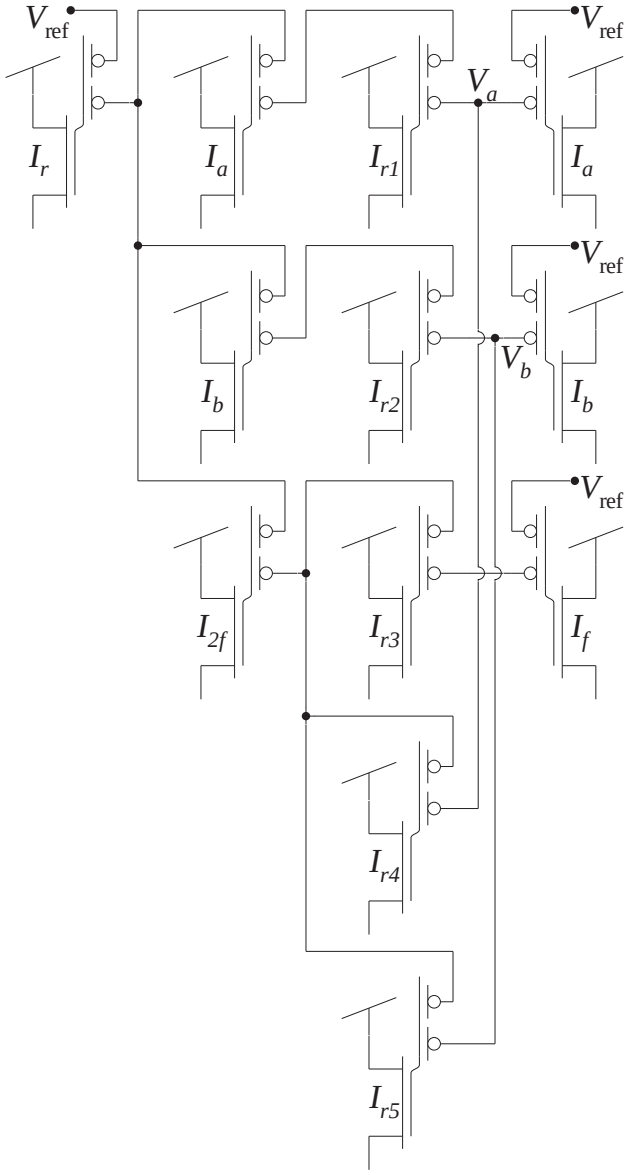


Figure 2.5: Final consolidated MITE network for the radius calculation.

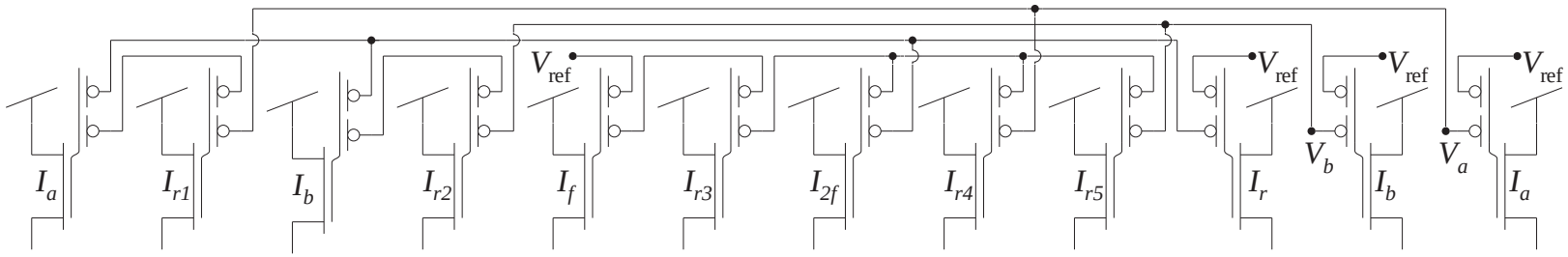


Figure 2.6: Final consolidated MITE network for the radius calculation.

the inputs ($I_a, I_b, I_f,$ and I_{2f}) leaves only the outputs of the translinear loops and the final output, I_r. Observing that the KCL equation equates the sum of I_{r1}, I_{r2}, and I_{r3} to the sum of I_{r4}, I_{r5}, and I_r, we connect the drains of the appropriate MITEs and send these two summed currents into a current mirror by adding two NMOS transistors. The biased circuit is shown in Fig. 2.7.

2.3.5 Diode Connections

Making diode connections around all the MITEs passing input currents leaves just the two KCL nodes. Choosing to diode connect around the NMOS transistor passing $I_{r4} + I_{r5} + I_r$ forces us to diode connect around the I_{r3} MITE. The final circuit is shown in Fig. 2.8.

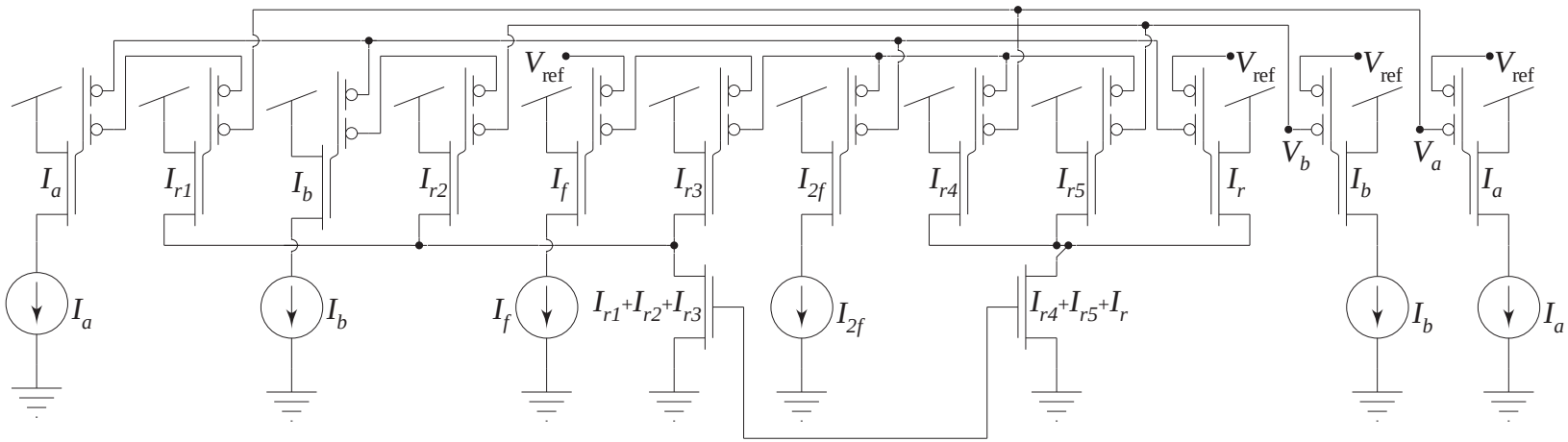


Figure 2.7: Biasing of the radius calculation network.

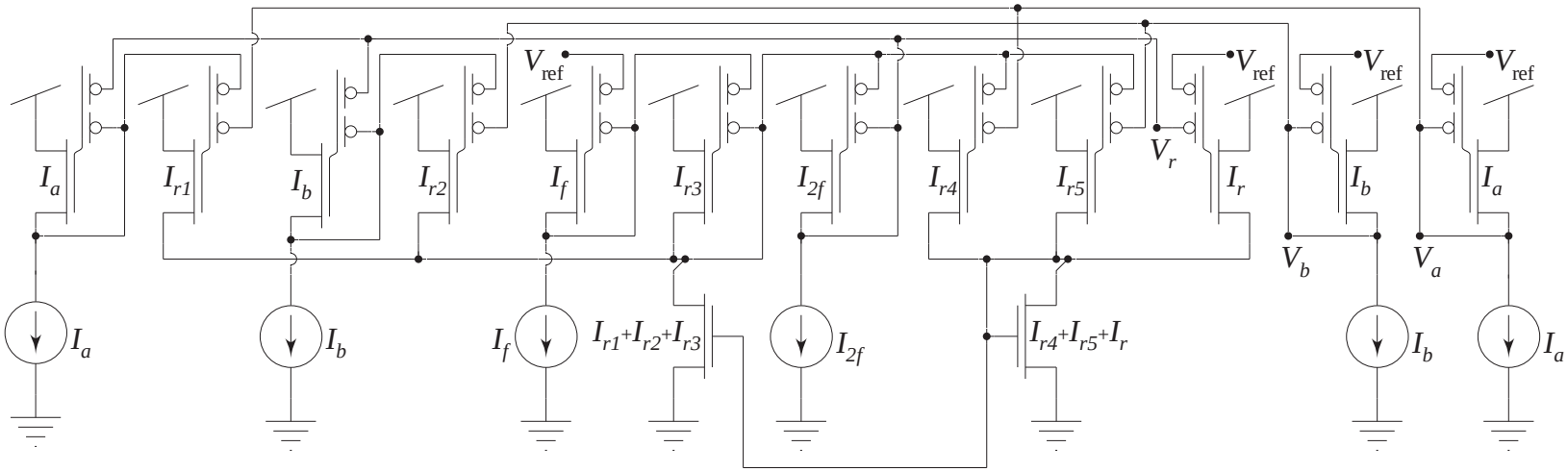


Figure 2.8: Diode connections to complete the radius calculation network.

Chapter 3

Linear and Non-Linear First-Order Dynamic MITE Networks

3.1 Dynamic MITE Networks

A more interesting type of MITE network is called the *dynamic MITE network*. These networks are identified by having a dependency on time and most easily recognized by a d/dt in the system description. Some circuits that fall into this classification are RMS-to-DC converters, log-domain filters, and oscillators. Two sample system descriptions are a first-order low-pass filter,

$$\tau \frac{dy}{dt} = x - y \,, \tag{3.1}$$

and an RMS-to-DC converter,

$$2\tau z \frac{dz}{dt} + z^2 = u^2 - 2uv + v^2 \,. \tag{3.2}$$

3.2 First-Order Low-Pass Filter

3.2.1 System Decomposition

A first-order low-pass filter of the form,

$$\tau \frac{dy}{dt} = x - y\,, \tag{3.3}$$

is a good example to demonstrate how to implement a dynamic MITE network. We can define ratios of currents to represent the variables, x and y, given by

$$x = \frac{I_x}{I_1} \quad \text{and} \quad y = \frac{I_y}{I_1}\,. \tag{3.4}$$

Introducing these representations, we find that

$$\tau \frac{d}{dt}\left(\frac{I_y}{I_1}\right) = \frac{I_x}{I_1} - \frac{I_y}{I_1}\,, \tag{3.5}$$

which we can simplify by multiplying through by I_1, to obtain

$$\tau \frac{dI_y}{dt} = I_x - I_y\,. \tag{3.6}$$

The form of the above equation requires us to introduce an output structure in order to generate the dI_y/dt. Either an inverting or a non-inverting output structure can be used. Experience with both kinds of output structures has indicated that the non-inverting version often requires additional transistors in order to mirror currents resulting in a more complicated circuit. Therefore, the inverting output structure, shown in Fig. 3.1 and detailed in Section 3.2.2, will be used for all dynamic MITE network examples.

3.2.2 The Inverting Output Structure

In order to analyze this structure (shown in Fig. 3.1) and ultimately find an expression for dI_{out}/dt, we represent the output current, I_{out}, in terms of the control

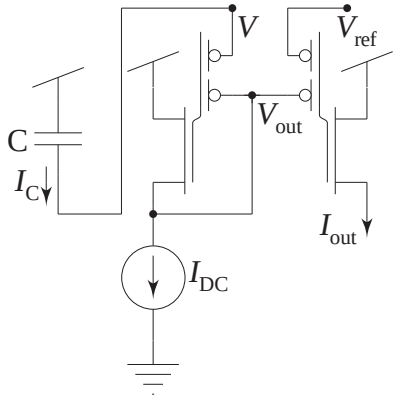


Figure 3.1: Inverting output structure used to introduce a dI/dt.

gate voltages, V_{ref} and V_{out},

$$I_{\text{out}} = I_{\text{s}} e^{\kappa(wV_{\text{ref}} + wV_{\text{out}})/U_{\text{T}}}\,, \tag{3.7}$$

where V_{ref} is a DC reference voltage. Similarly, the current flowing through the other MITE is represented by

$$I_{\text{DC}} = I_{\text{s}} e^{\kappa(wV + wV_{\text{out}})/U_{\text{T}}}\,. \tag{3.8}$$

We can remove V_{out} by dividing Eq. 3.7 by Eq. 3.8, obtaining

$$\frac{I_{\text{out}}}{I_{\text{DC}}} = e^{\kappa(wV_{\text{ref}} + wV_{\text{out}} - wV - wV_{\text{out}})/U_{\text{T}}}\,. \tag{3.9}$$

Solving for I_{out}, we find that

$$I_{\text{out}} = I_{\text{DC}} e^{\kappa(wV_{\text{ref}} - wV)/U_{\text{T}}}\,. \tag{3.10}$$

Assuming κ is constant (i.e., only I_{out} and V vary with time), the derivative of I_{out} with respect to time is found to be

$$\frac{dI_{\text{out}}}{dt} = \underbrace{I_{\text{DC}} e^{\kappa(wV_{\text{ref}} - wV)/U_{\text{T}}}}_{I_{\text{out}}} \left(-\frac{\kappa w}{U_{\text{T}}}\right) \frac{dV}{dt}\,, \tag{3.11}$$

which can be simplified to

$$\frac{dI_{\text{out}}}{dt} = I_{\text{out}} \left(-\frac{\kappa w}{U_{\text{T}}} \right) \frac{dV}{dt} \,. \tag{3.12}$$

Noticing that the capacitor current, I_{C}, can be defined as

$$I_{\text{C}} = C \frac{dV}{dt} \,, \tag{3.13}$$

we can use this expression to remove the dV/dt from Eq. 3.12 to find

$$\frac{dI_{\text{out}}}{dt} = I_{\text{out}} \left(-\frac{\kappa w}{U_{\text{T}}} \right) \frac{I_{\text{C}}}{C} \,. \tag{3.14}$$

Multiplying both sides by τ and rearranging gives us the desired term on the left-hand side and the grouped expression on the right-hand side has units of inverse current.

$$\tau \frac{dI_{\text{out}}}{dt} = -I_{\text{out}} I_{\text{C}} \left(\frac{\tau \kappa w}{U_{\text{T}} C} \right) \,. \tag{3.15}$$

We can define a current, I_τ, that can be used to tune the time constant of the circuit.

$$I_\tau \equiv \frac{U_{\text{T}} C}{\tau \kappa w} \tag{3.16}$$

The final result of the analysis of this output structure is an expression for the derivative of I_{out} in terms of the capacitor current and a current that controls the time constant.

$$\tau \frac{dI_{\text{out}}}{dt} = -\frac{I_{\text{out}} I_{\text{C}}}{I_\tau} \tag{3.17}$$

Using this expression, we can remove all time derivatives during the decomposition phase. The negative sign on the right hand side gives this structure its name, *inverting* output structure. The remaining chapters will assume that every output structure is of the inverting kind.

It is very important to remember that this result is only valid if the general form of the output structure is maintained. Specifically, the capacitor must connect to

the output MITE through a single MITE passing a DC current. The output MITE must also only have DC voltages connected to its "unused" control gates (shown as being connected to V_{ref} in Fig. 3.1). This configuration is easily maintained by restricting the output currents to be on either end of the connection graphs with DC currents as their inner neighbors. It is a good practice to double-check that the form of the output structure has been maintained after the circuit is completed.

3.2.3 System Decomposition Continued

Using the expression derived from the inverting output structure, we can continue to decompose the low-pass filter description in Eq. 3.6. Note that a separate output structure is required for every derivative. See Sections. 4.1 and 4.2 for examples of higher order systems requiring multiple output structures. Replacing $\tau dI_y/dt$ according to the relationship in Eq. 3.17 (I_{out} is replaced with I_y), we find that

$$-\frac{I_{\text{C}} I_y}{I_\tau} = I_x - I_y \,. \tag{3.18}$$

Because the capacitor current, I_{C}, is not an input current and not generated by a transistor, it cannot be a part of any TL equation. Therefore, all equations must be solved for any capacitor currents (if they are present) in order to ensure that they are only included in KCL equations. Solving for I_{C}, we obtain

$$I_{\text{C}} = I_\tau - \underbrace{\frac{I_\tau I_x}{I_y}}_{I_{\text{TL}}} \,, \tag{3.19}$$

which leaves us with the following KCL equation and TL equation:

$$\begin{aligned} \text{KCL}: &\quad I_{\text{C}} = I_\tau - I_{\text{TL}} \\ \text{TL}: &\quad I_{\text{TL}} I_y = I_\tau I_x \,. \end{aligned} \tag{3.20}$$

3.2.4 Translinear Loops

We can begin to construct this circuit by examining the TL equation shown in Eq. 3.20. Because all currents are of the first degree, we can connect them in an alternating pattern of one to one connections as shown in Fig. 3.2(a) and as detailed below:

$$I_x \quad \leftrightarrow \quad I_{\mathrm{TL}} \quad \leftrightarrow \quad I_\tau \quad \leftrightarrow \quad I_y \,.$$

Note that we have maintained the output structure ordering by placing the output current, I_y, on the right end with a DC current, I_τ, as its inner neighbor.

We begin with a MITE for I_x from the right-hand side and make a single connection to a MITE for I_{TL} from the left-hand side. Then using the other control gate of the I_{TL} MITE, we make a connection to a MITE for I_τ from the right-hand side. The connections are completed with a final connection from the remaining control gate of I_τ's MITE to a MITE for I_y. This order was chosen because I_x is the input and I_y is the output thus giving a signal flow from left to right. Note that the unused control gates on both ends have been given connections that are connected to a reference voltage (also ensuring that the output structure relationship remains valid).

3.2.5 Biasing

Each MITE needs to be biased according to the label shown on the floating-gate transistors. We accomplish this by connecting input current sources to the drains for the I_x and I_τ MITEs. The KCL equation (from Eq. 3.20) is then used to bias the I_{TL} MITE. Figure 3.2(b) shows the circuit with the proper biasing.

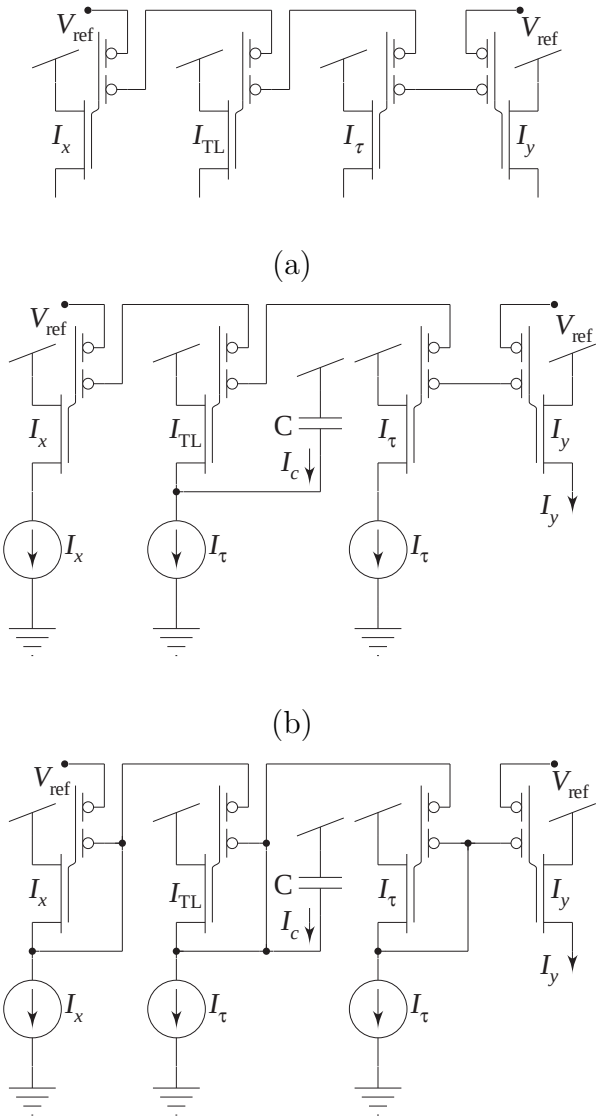


Figure 3.2: Circuit construction for a first-order low-pass filter.

3.2.6 Diode Connections

Since the signal flow of this circuit is very obviously left to right, we make the diode connections in the same direction starting with the input MITE passing I_x. The KCL node is then diode connected. The circuit is completed with the final diode connection of the I_τ MITE. Figure 3.2(c) shows the completed low-pass filter circuit and experimental data from this circuit can be found in Section 6.2.

3.3 RMS-to-DC Converter

Suppose that we need to implement an RMS-to-DC converter, which we can describe in the time domain with two static nonlinear constraints and a linear ordinary differential equation, given by

$$x = w^2, \quad \tau\frac{dy}{dt} + y = x, \quad \text{and} \quad z = \sqrt{y}, \tag{3.21}$$

where w is the input signal, whose RMS amplitude we want to compute, x is the square of the input signal, y is a low-pass filtered version of x, giving an approximation of the time average of the square of the input signal, and z is the output of the system, giving the square-root of the time average value of the square of the input signal. We shall assume that w can take on both negative and positive values. Since all variables are represented by currents and must be strictly positive, we will need to provide a DC offset, v, which will make positive the total input, $u = w + v$, to the circuit that computes x. Note that $x = w^2$ will always be a nonnegative quantity, so the low-pass filter only needs to be single-ended.

3.3.1 System Decomposition

One approach to designing such a circuit would be to synthesize separately a squaring circuit, a first-order low-pass filter, and a square-root circuit and cascade these together with current mirrors. Although this approach will work, we shall take a different tact in this example, resulting in a more efficient implementation. We begin by eliminating x and y from the description of the system given in Eq. 3.21. We have that

$$x = w^2, \quad y = z^2, \quad \text{and} \quad \frac{dy}{dt} = 2z\frac{dz}{dt}\,, \tag{3.22}$$

which we can substitute into the ordinary differential equation in Eq. 3.21, thereby obtaining a first-order algebraic differential equation, given by

$$2\tau z\frac{dz}{dt} + z^2 = w^2\,. \tag{3.23}$$

However, this equation is not directly implementable as a dynamic translinear circuit because w can be positive or negative. To remedy this situation, we substitute $u - v$ for w into this equation and expand the right-hand side to obtain a directly-implementable equation, given by

$$2\tau z\frac{dz}{dt} + z^2 = u^2 - 2uv + v^2\,. \tag{3.24}$$

Next, we represent u, v, and z as ratios of signal currents to a unit current, I_1, given respectively by

$$u = \frac{I_u}{I_1}, \quad v = \frac{I_v}{I_1}, \quad \text{and} \quad z = \frac{I_z}{I_1}\,. \tag{3.25}$$

We substitute these representations into Eq. 3.24 and after multiplying both sides of the equation by I_1^2, we obtain

$$I_z\left(2\tau\frac{dI_z}{dt}\right) + I_z^2 = I_u^2 - I_u\left(2I_v\right) + I_v^2\,. \tag{3.26}$$

In order to implement the time derivative in this equation, we use the inverting output structure to replace $2\tau dI_z/dt$ with $-I_z I_\mathrm{C}/I_\tau$ to get

$$I_z\left(\frac{-I_z I_\mathrm{C}}{I_\tau}\right) + I_z^2 = I_u^2 - I_u I_{2v} + I_v^2\,. \tag{3.27}$$

Note that we have absorbed the first 2 into the τ constant which becomes part of I_τ,

$$I_\tau = \frac{U_\mathrm{T} C}{2\tau\kappa w}\,, \tag{3.28}$$

and the second 2 into the offset current I_v,

$$I_{2v} = 2I_v\,. \tag{3.29}$$

Solving for I_C, we find

$$I_\mathrm{C} = I_\tau - \underbrace{\frac{I_\tau I_u^2}{I_z^2}}_{I_\mathrm{TL1}} + \underbrace{\frac{I_\tau I_u\,(2I_v)}{I_z^2}}_{I_\mathrm{TL2}} - \underbrace{\frac{I_\tau I_v^2}{I_z^2}}_{I_\mathrm{TL3}}\,. \tag{3.30}$$

From this equation, we obtain the following KCL equation and three TL equations:

$$\begin{aligned} \text{KCL}: &\quad I_\mathrm{C} = I_\tau - I_\mathrm{TL1} + I_\mathrm{TL2} - I_\mathrm{TL3} \\ \text{TL}: &\quad I_\mathrm{TL1} I_z^2 = I_\tau I_u^2 \quad I_\mathrm{TL2} I_z^2 = I_\tau I_u I_{2v} \quad I_\mathrm{TL3} I_z^2 = I_\tau I_v^2\,. \end{aligned} \tag{3.31}$$

3.3.2 Translinear Loops

For this example, I have chosen to not adhere to the two control gate restriction in order to give an example where more than two control gates are used. Before we begin, it is worthwhile to point out that each of the TL equations have the same relationship between I_z and I_τ (illustrated below in bold) which will provide an opportunity to simplify the network through consolidation. To take advantage of the similarity, we first make MITE connections according to the relationship between I_z and I_τ and then between I_τ and $I_{\mathrm{TL}i}$. This arrangement maintains the

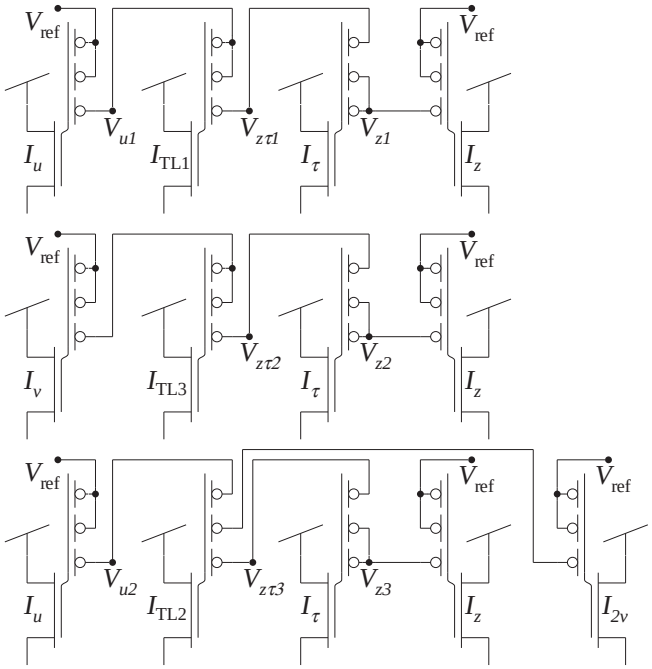
required output structure connections ($I_z \leftrightarrow I_\tau$) and is summarized below. For this example, 3-control gate MITEs are used allowing for a connection from the I_{TL2} MITE to both the I_u and I_{2v} MITEs, as shown on the last three lines below:

$$\begin{array}{lllll}
\boldsymbol{I_z^2} & \leftrightarrow & \boldsymbol{I_\tau} & \leftrightarrow I_{\mathrm{TL1}} & \leftrightarrow I_u^2 \\
\boldsymbol{I_z^2} & \leftrightarrow & \boldsymbol{I_\tau} & \leftrightarrow I_{\mathrm{TL3}} & \leftrightarrow I_v^2 \\
 & & & \swarrow & I_u \\
\boldsymbol{I_z^2} & \leftrightarrow & \boldsymbol{I_\tau} & \leftrightarrow I_{\mathrm{TL2}} & \\
 & & & \searrow & I_{2v}
\end{array}$$

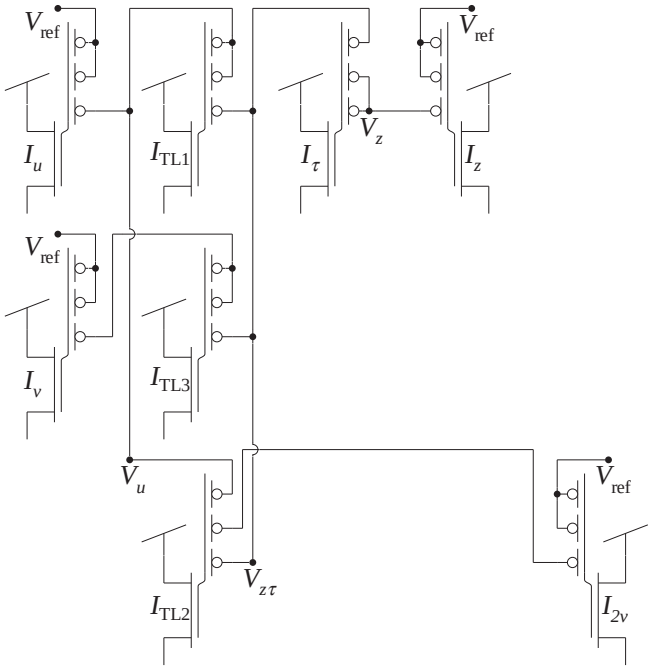
These connections are shown in Fig. 3.3(a) with all unused control gates connected to V_{ref}. Note that allowing more than two control gates has significantly increased the complexity of the inter-MITE connections and has also eliminated the "linear" one to one connections. (The I_{TL} MITE has two right neighbors instead of the usual single right neighbor.)

3.3.3 Consolidation

Connecting in the above order allows for the removal of several MITEs. Since there are three control gates per MITE and the connectivity is not in a straightforward left to right order, more care must be given to make sure that all sharing is valid. In this case, we can share a voltage when two MITEs are passing the same current and two of the three control gate potentials match. This observation implies that the third control gate on each MITE must be at the same potential, and therefore, can be shared. Looking at Fig. 3.3, we find that a single I_u MITE can be shared since V_{u1} and V_{u2} must be equal. We can also share a single I_z and a single I_τ MITE since nodes V_{z1}, V_{z2}, and V_{z3} must be equal which implies that nodes $V_{z\tau 1}$, $V_{z\tau 2}$, and $V_{z\tau 3}$ must also be equal. Figure 3.3(b) shows the consolidated network



(a)



(b)

Figure 3.3: Initial MITE connections and consolidation for an RMS-to-DC converter.

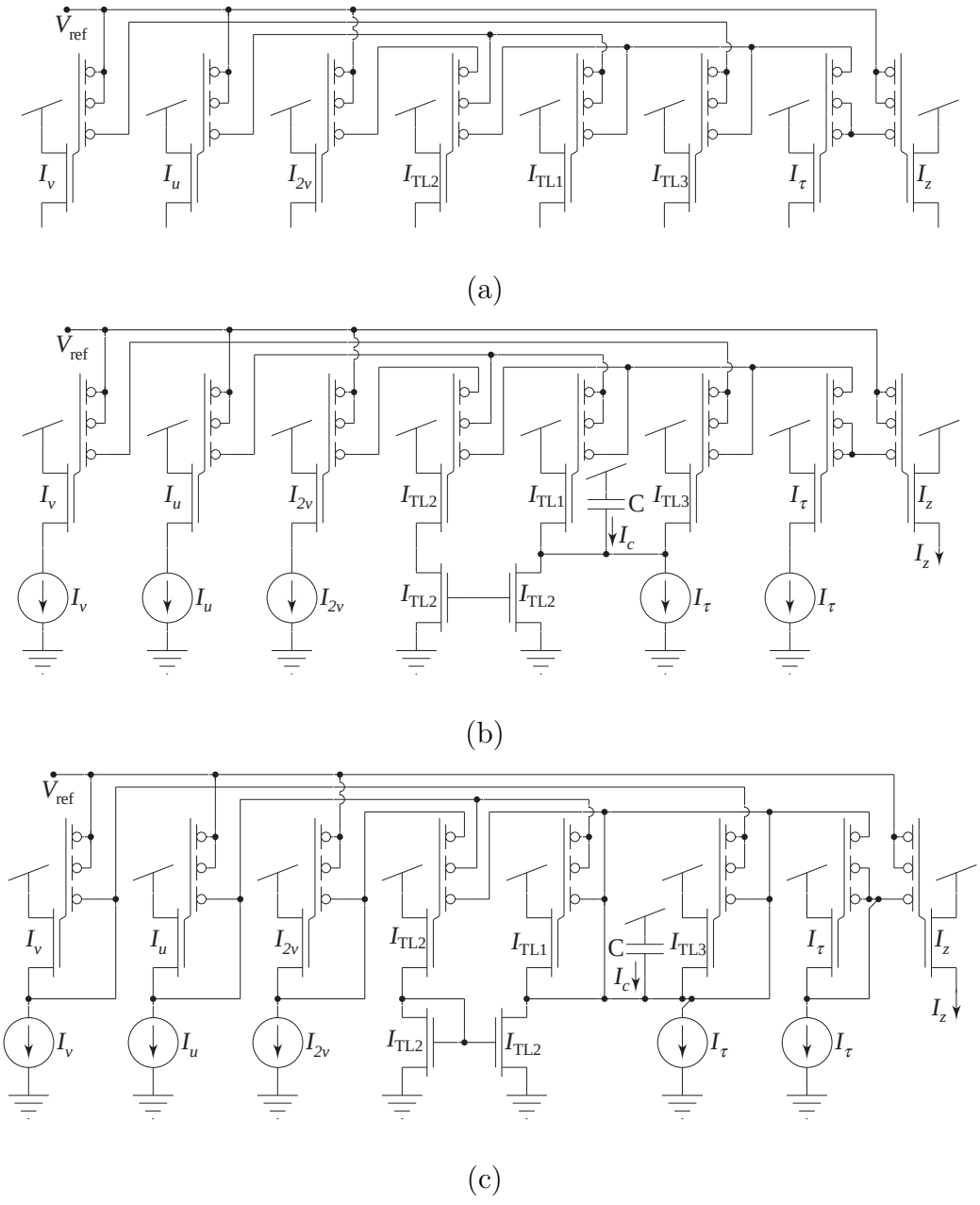


Figure 3.4: Biasing and diode connections to complete the RMS-to-DC converter circuit.

with the removal of those five redundant MITEs.

3.3.4 Biasing

The consolidated network is rearranged and shown in Fig. 3.4(a). Half of the MITEs can be biased with simple current sources (I_v, I_u, I_{2v}, and I_τ). Because a current sink passing I_{TL2} is required in the KCL equation, we bias the I_{TL2} MITE with an NMOS current mirror. We now use the other half of that mirror in the KCL equation,

$$I_{\mathrm{C}} = I_\tau - I_{\mathrm{TL1}} + I_{\mathrm{TL2}} - I_{\mathrm{TL3}}\,, \tag{3.32}$$

adding a current source for I_τ and a capacitor. These components are connected to the two drains of the I_{TL2} and I_{TL3} MITEs. Figure 3.4(b) shows all biasing connections.

3.3.5 Diode Connections

Finally, we diode connect the MITEs by starting with those connected to current sources on the left. Choosing to diode connect the left NMOS of the I_{TL2} mirror forces us to diode connect around either the I_{TL1} or I_{TL3} MITE. Since the only available control gate is shared by both, a double diode connection is made at the KCL node. The final diode connection is made at the I_τ MITE, which also creates the inverting output structure that we were required to maintain.

The final circuit is shown in Fig. 3.4(c) where all V_{ref} nodes have been connected. In practice, it is generally not wise to create any signal path that does not explicitly pass through the capacitor for a dynamic MITE network as was done when all V_{ref} nodes were connected. A further explanation of reasons to avoid these kind of

connections can be found in Chapter 6. Experimental data from this circuit can be found in Section 6.3.

Chapter 4

Linear and Non-Linear Second-Order Dynamic MITE Networks

In this chapter, we shall consider second-order systems whose dynamics are described by a second-order algebraic differential equation (ADE) or by a system of two coupled first-order ADEs. During the initial decomposition, any high order systems must be separated into a set of first-order ADEs before continuing on with the normal decomposition process.

4.1 Second-Order Low-Pass Filter

4.1.1 System Decomposition

We can implement a second-order low-pass filter, described by

$$\tau^2 \frac{d^2y}{dt^2} + \frac{\tau}{Q}\frac{dy}{dt} + y = x\,, \tag{4.1}$$

in much the same way as we did the first-order one by viewing it as a first-order filter embedded inside another. This way, we break down the second-order system,

$$\tau\frac{d}{dt}\underbrace{\left(\tau\frac{dy}{dt}+\frac{y}{Q}\right)}_{z}+y=x\,, \tag{4.2}$$

into two first-order systems,

$$\tau\frac{dz}{dt}=x-y \quad \text{and} \quad \tau\frac{dy}{dt}=z-\frac{y}{Q}\,. \tag{4.3}$$

We then represent the variables by current ratios to find

$$\tau\frac{d}{dt}\left(\frac{I_z}{I_1}\right)=\frac{I_x}{I_1}-\frac{I_y}{I_1} \quad \text{and} \quad \tau\frac{d}{dt}\left(\frac{I_y}{I_1}\right)=\frac{I_z}{I_1}-\frac{1}{Q}\frac{I_y}{I_1}\,. \tag{4.4}$$

In this example, we chose to leave Q as a dimensionless scaling factor because it can be combined with I_τ as will be shown in the next few steps. Multiplying through by I_1 simplifies the equations to

$$\underbrace{\tau\frac{dI_z}{dt}}_{-\frac{I_{\mathrm{Cz}}I_z}{I_\tau}}=I_x-I_y \quad \text{and} \quad \underbrace{\tau\frac{dI_y}{dt}}_{-\frac{I_{\mathrm{Cy}}I_y}{I_\tau}}=I_z-\frac{I_y}{Q}\,. \tag{4.5}$$

As shown above, we use the relationship for the output structure from Eq. 3.17 to remove the time derivatives. Note that multiple capacitor currents should be labeled differently, because the capacitor currents were introduced by way of two different output structures. Also, if the time constants are different, the I_τ currents should be labeled accordingly (not so in this example). By solving for the capacitor currents and defining the current, $I_{\tau/\mathrm{Q}}$ to be I_τ/Q, we find

$$I_{\mathrm{Cz}}=\underbrace{\frac{I_yI_\tau}{I_z}}_{I_{\mathrm{TL1}}}-\underbrace{\frac{I_xI_\tau}{I_z}}_{I_{\mathrm{TL2}}} \quad \text{and} \quad I_{\mathrm{Cy}}=I_{\tau/\mathrm{Q}}-\underbrace{\frac{I_zI_\tau}{I_y}}_{I_{\mathrm{TL3}}}\,. \tag{4.6}$$

Defining TL equations as shown with the underbraces, we are left with the following final decomposed system:

$$\begin{aligned} \text{KCL}: \quad & I_{Cz} = I_{TL1} - I_{TL2} \quad I_{Cy} = I_{\tau/\text{Q}} - I_{TL3} \\ \text{TL}: \quad & I_{\text{TL1}} I_z = I_y I_\tau \qquad I_{\text{TL2}} I_y = I_z I_\tau \quad I_{\text{TL3}} I_z = I_x I_\tau \,. \end{aligned} \tag{4.7}$$

4.1.2 Translinear Loops

Before beginning to connect MITEs, it is useful to arrange the KCL equations in an alternating pattern to try to determine if any opportunities to consolidate MITEs exist. This is similar to factoring out common terms for algebraic manipulations. Using all one to one connections and restricting MITEs to two control gates, the connections can be arranged in the alternating odd-even order shown below. We have maintained the form for both output structures by placing the two output currents, I_y and I_z, on the outsides with DC currents (I_τ) for their inner neighbors. The currents on the ends can easily be compared to look for common patterns going inwards. The following arrangement allows for the maximum amount of consolidation, as explained in Section 4.1.3. The initial connections for this arrangement is shown in Fig. 4.1(a).

$$\begin{array}{ccccccc} I_x & \leftrightarrow & I_{\text{TL3}} & \leftrightarrow & I_\tau & \leftrightarrow & I_z \\ I_y & \leftrightarrow & I_{\text{TL1}} & \leftrightarrow & I_\tau & \leftrightarrow & I_z \\ I_y & \leftrightarrow & I_\tau & \leftrightarrow & I_{\text{TL2}} & \leftrightarrow & I_z \end{array}$$

4.1.3 Consolidation

As indicated below, the I_y terms on the left end of the latter two rows imply that one of these MITEs can be removed. This consolidation arises from the fact that V_{y1} and V_{y2} in Fig. 4.1(a) must be equal. Similarly, the I_z terms on the right end

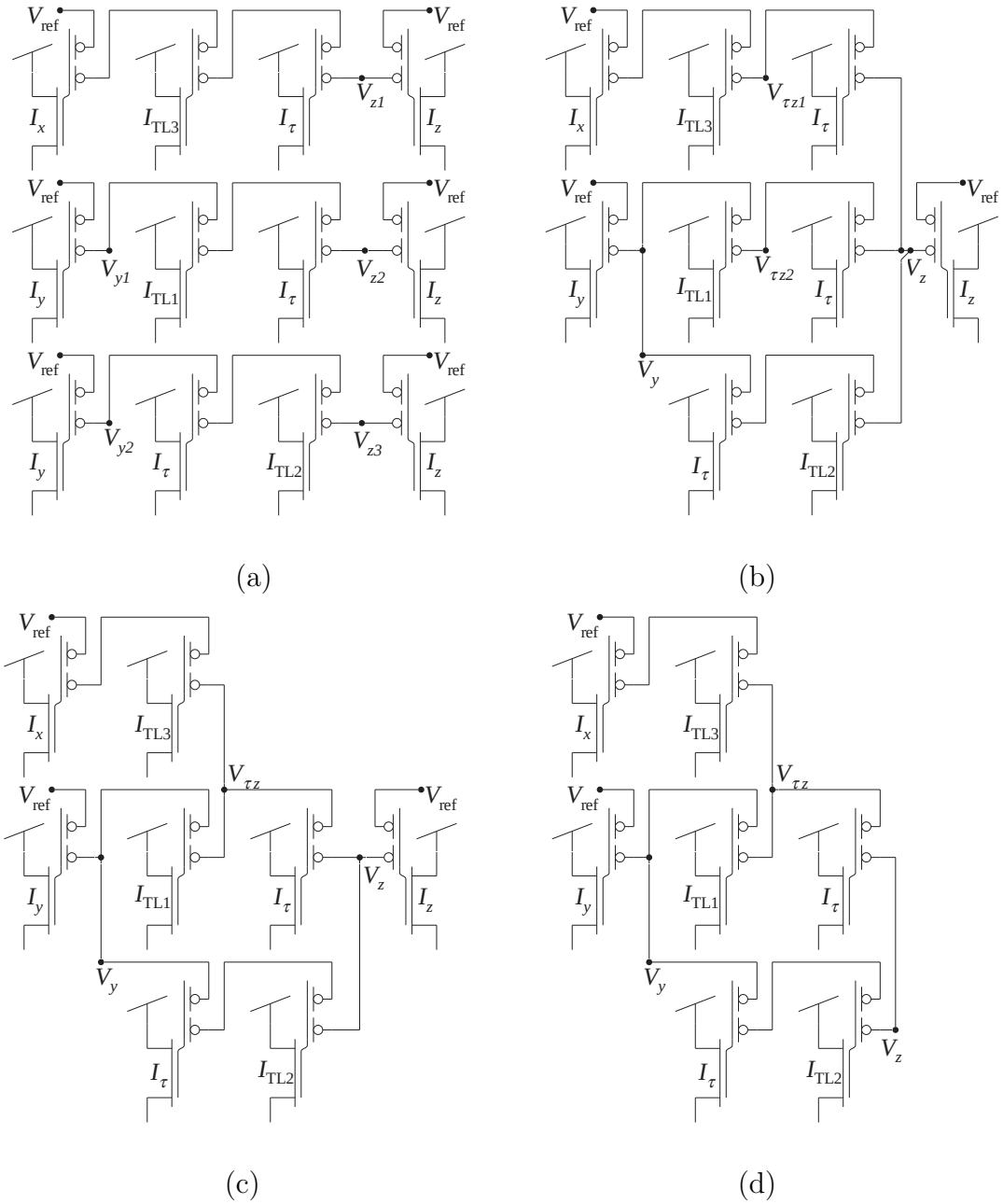


Figure 4.1: Initial MITE connections and consolidation during circuit construction for the second-order low-pass filter.

also allow one I_z MITE to be used for all three I_z MITEs. This step is possible because all three V_z voltages must also be equal.

$$\begin{array}{ccccccc} I_x & \leftrightarrow & I_{\mathrm{TL3}} & \leftrightarrow & I_\tau & \leftrightarrow & \boldsymbol{I_z} \\ \boldsymbol{I_y} & \leftrightarrow & I_{\mathrm{TL1}} & \leftrightarrow & I_\tau & \leftrightarrow & \boldsymbol{I_z} \\ \boldsymbol{I_y} & \leftrightarrow & I_\tau & \leftrightarrow & I_{\mathrm{TL2}} & \leftrightarrow & \boldsymbol{I_z} \end{array}$$

The highlighted terms below show another opportunity to consolidate. The top two rows share an $I_\tau I_z$ combination on their right ends. The top row's I_z has already been removed, but this does not change the fact that it still matches the middle one. Thus, we can remove the I_τ MITE as indicated in Fig. 4.1(c). Looking at the circuit in Fig. 4.1(b), it should be fairly obvious that the $V_{\tau z}$ voltages are equal which allows us to remove one of those two I_τ MITEs.

$$\begin{array}{ccccccc} I_x & \leftrightarrow & I_{\mathrm{TL3}} & \leftrightarrow & \boldsymbol{I_\tau} & \searrow & \\ I_y & \leftrightarrow & I_{\mathrm{TL1}} & \leftrightarrow & \boldsymbol{I_\tau} & \leftrightarrow & \boldsymbol{I_z} \\ & \searrow & I_\tau & \leftrightarrow & I_{\mathrm{TL2}} & \nearrow & \end{array}$$

To summarize, the original ordering is shown below on the left and the new consolidated network is shown on the right.

$$\begin{array}{ccccccccc} I_x & \leftrightarrow & I_{\mathrm{TL3}} & \leftrightarrow & I_\tau & \leftrightarrow & I_z & & \\ I_y & \leftrightarrow & I_{\mathrm{TL1}} & \leftrightarrow & I_\tau & \leftrightarrow & I_z & \Rightarrow & \\ I_y & \leftrightarrow & I_\tau & \leftrightarrow & I_{\mathrm{TL2}} & \leftrightarrow & I_z & & \end{array} \qquad \begin{array}{ccccccccc} I_x & \leftrightarrow & I_{\mathrm{TL3}} & \searrow & & & & & \\ I_y & \leftrightarrow & I_{\mathrm{TL1}} & \leftrightarrow & I_\tau & \leftrightarrow & I_z & & \\ & \searrow & I_\tau & \leftrightarrow & I_{\mathrm{TL2}} & \nearrow & & & \end{array}$$

Taking a closer look at the circuit in Fig. 4.1(c), we see that it is possible to remove another MITE. Thinking back to the original decomposition, we defined I_z (or z) as the intermediate variable used to break the second-order system down into two first-order equations. Because we are not interested in what I_z actually looks like, we can remove that MITE altogether. This does not remove the effect of having I_z in the circuit but merely leaves this signal in a log-compressed form

at the node labeled V_z. The remaining MITEs are shown in Fig. 4.1(d). If this MITE remained until the circuit was completed, it would become obvious that it is unnecessary since it will not be diode connected (outputs are never diode connected unless mirrored) and the generated current will not be mirrored around for use elsewhere in the circuit.

4.1.4 Biasing

Figure 4.2(a) shows the consolidated and reduced MITEs in the same configuration but rearranged into a one dimensional array for biasing. As with all inputs, current sources are added for biasing the I_x MITE and the two I_τ MITEs. The first KCL equation allows us to bias the I_{TL2} MITE with a capacitor and an NMOS transistor sinking I_{TL1}. This NMOS transistor implies that it will be either the input or output of a current mirror passing I_{TL1} so another NMOS transistor is used to bias the I_{TL1} MITE. With the I_y MITE remaining unbiased, we bias the final MITE with the second KCL equation by adding a capacitor and a current source passing $I_{\tau/Q}$.

4.1.5 Diode Connections

Starting at the left and forcing a left to right signal flow, we diode connect around the I_x MITE. Choosing to diode connect around the I_{TL2} MITE forces us to also diode connect the I_{TL1} NMOS transistor. Continuing on in a straightforward left to right order, we can finish all the diode connections and complete the circuit. Figure 4.2(c) shows the completed second-order low-pass filter and experimental data can be found in Section 6.4.

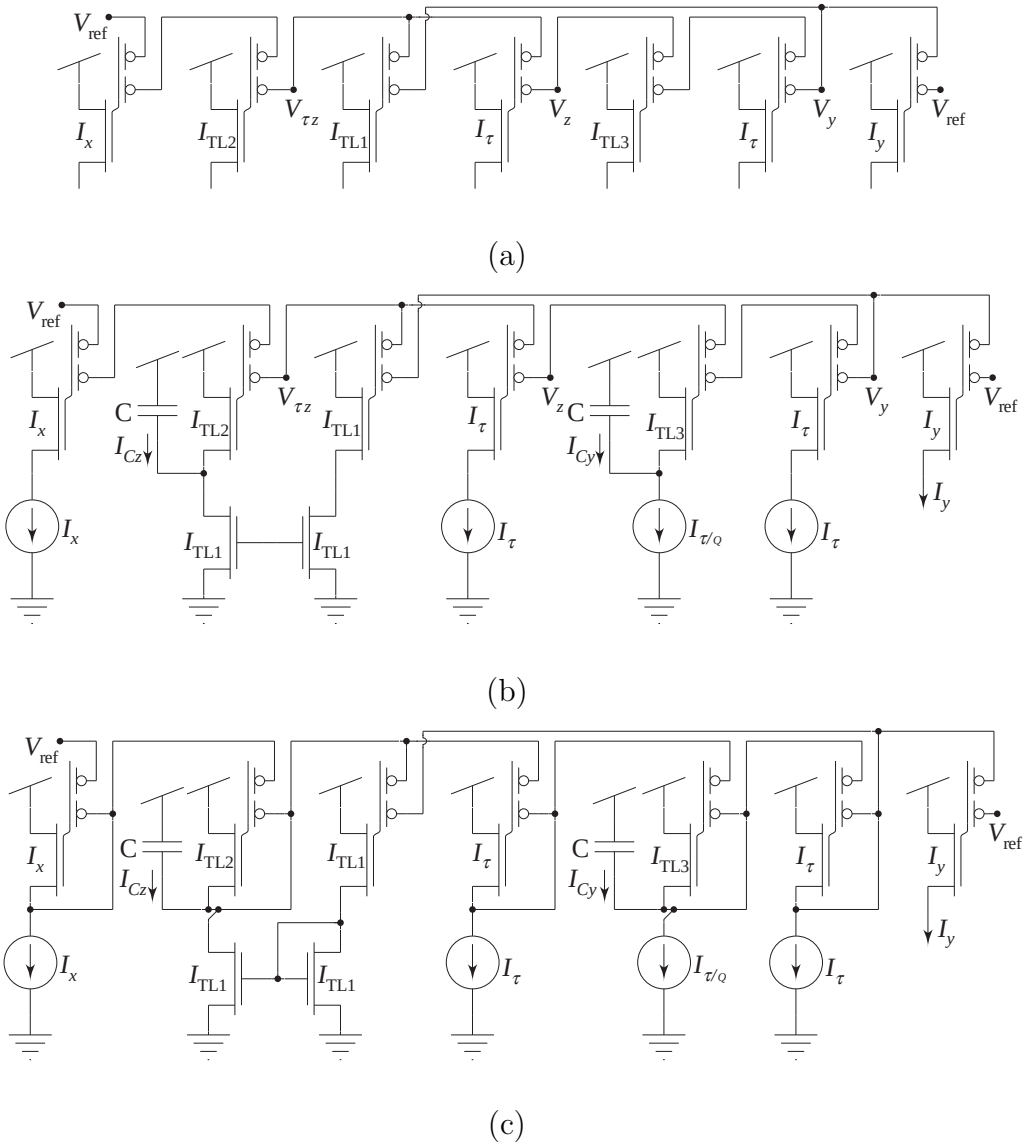


Figure 4.2: Biasing and diode connections for the completion of a second-order low pass filter.

4.2 Quadrature Oscillator

Another useful circuit that is significantly more complicated than the earlier examples is a quadrature oscillator. There are two output signals in this system which are both sinusoidal and 90 degrees out of phase. The frequency and amplitude of these signals are controlled by inputs. Controllable oscillators have many uses and the one described in this chapter will be used in the phase-locked loop example in Chapter 5.

4.2.1 System Decomposition

We begin by listing the constraints for a quadrature oscillator in polar coordinates (constant radius vector of the two outputs, r, and frequency, $d\theta/dt$),

$$\tau\frac{dr}{dt} = \gamma r(\rho - r) \quad \text{and} \quad \tau\frac{d\theta}{dt} = 1\,, \tag{4.8}$$

where ρ is the desired radius and γ determines the circuit's sensitivity to deviations in the desired radius. We can transform these constraints to the Cartesian system with the following mapping:

$$x = r\,\cos(\theta) \quad \text{and} \quad y = r\,\sin(\theta)\,. \tag{4.9}$$

Finding dx/dt gives

$$\frac{dx}{dt} = \cos(\theta)\,\frac{dr}{dt} - r\,\sin(\theta)\,\frac{d\theta}{dt}\,. \tag{4.10}$$

Using Eqs. 4.8 and 4.9 to eliminate θ from the right-hand side results in

$$\frac{dx}{dt} = \frac{\gamma}{\tau}x\,(\rho - r) - \frac{y}{\tau}\,. \tag{4.11}$$

Similarly, we can calulate dy/dt as

$$\frac{dy}{dt} = \frac{\gamma}{\tau}y\,(\rho - r) + \frac{x}{\tau}\,, \tag{4.12}$$

giving us the following system description:

$$r = \sqrt{x^2 + y^2} \tag{4.13}$$

$$\tau \frac{dx}{dt} = -y + \gamma(\rho - r) \tag{4.14}$$

$$\tau \frac{dy}{dt} = x + \gamma(\rho - r)\,. \tag{4.15}$$

It is possible to combine the radius calculation of Eq. 4.13 into Eqs. 4.14 and 4.15 but it seems to make more sense to have a separate network calculate the radius. We have already constructed a vector magnitude circuit (with offsets applied to x and y) in Section 2.3 that will be used to calculate the radius.

4.2.2 Dynamic Constraints

Decomposing the dynamic constraints on x, we add offsets and introduce current ratios for the variables in Eq. 4.14,

$$\tau \frac{dx}{dt} = -y + \gamma x\,(\rho - r)\ . \tag{4.16}$$

We add offsets to x and y (in the same way as in the vector magnitude circuit of Section 2.3), which are given by

$$a = x + f \qquad b = y + f\,, \tag{4.17}$$

finding that Eq. 4.16 becomes

$$\tau \frac{d\,(a - f)}{dt} = -\,(b - f) + \gamma\,(a - f)\,(\rho - r)\ , \tag{4.18}$$

which we can solve for $\tau da/dt$,

$$\tau \frac{da}{dt} = f - b + \gamma\,(a\rho - ar - f\rho + fr)\ . \tag{4.19}$$

Introducing current ratios, we obtain

$$\tau \frac{d}{dt}\left(\frac{I_a}{I_1}\right) = \frac{I_f}{I_1} - \frac{I_b}{I_1} + \gamma\left(\frac{I_a I_\rho}{I_1^2} - \frac{I_a I_r}{I_1^2} - \frac{I_f I_\rho}{I_1^2} + \frac{I_f I_r}{I_1^2}\right), \tag{4.20}$$

and multiplying through by I_1, we find

$$\tau \frac{dI_a}{dt} = I_f - I_b + \gamma\left(\frac{I_a I_\rho}{I_1} - \frac{I_a I_r}{I_1} - \frac{I_f I_\rho}{I_1} + \frac{I_f I_r}{I_1}\right). \tag{4.21}$$

We chose to leave γ as a dimensionless scaling factor that will later be combined with a DC current.

We can remove the $\tau dI_a/dt$ expression through the introduction of the inverting output structure of Fig. 3.1 where the output current is related to the capacitor current by

$$\tau \frac{dI_a}{dt} = -\frac{I_a I_{Ca}}{I_\tau}, \tag{4.22}$$

where I_{Ca} is the capacitor current and I_τ is a function of the value of the capacitor, τ, the thermal voltage, and the weighting of the MITE inputs ($I_\tau \equiv CU_{\mathrm{T}}/w\tau$). Using this relationship and solving for the capacitor current, we obtain

$$I_{Ca} = -\underbrace{\frac{I_f I_\tau}{I_a}}_{I_{a1}} + \underbrace{\frac{I_b I_\tau}{I_a}}_{I_{a2}} - \underbrace{\frac{I_{\gamma\tau} I_\rho}{I_1}}_{I_{a3}} + \underbrace{\frac{I_{\gamma\tau} I_r}{I_1}}_{I_{a4}} + \underbrace{\frac{I_{\gamma\tau} I_f I_\rho}{I_a I_1}}_{I_{a5}} - \underbrace{\frac{I_{\gamma\tau} I_f I_r}{I_a I_1}}_{I_{a6}}. \tag{4.23}$$

By introducing intermediate currents, we obtain the following TL and KCL equations:

$$\begin{aligned} \text{KCL}: \quad & I_{Ca} = -I_{a1} + I_{a2} - I_{a3} + I_{a4} + I_{a5} - I_{a6} \\ \text{TL}: \quad & I_{a1} I_a = I_f I_\tau \quad I_{a2} I_a = I_b I_\tau \quad I_{a3} I_1 = I_{\gamma\tau} I_\rho \\ & I_{a4} I_1 = I_{\gamma\tau} I_r \quad I_{a5} I_a I_1 = I_{\gamma\tau} I_f I_\rho \quad I_{a6} I_a I_1 = I_{\gamma\tau} I_f I_r, \end{aligned} \tag{4.24}$$

where we define $I_{\gamma\tau}$ as γI_τ.

By following an almost identical procedure, we find the equations defining the capacitor current for the "b" side (where y has been replaced with an offset variable,

$b = y + f$):

$$I_{Cb} = \underbrace{\frac{I_f I_\tau}{I_b}}_{I_{b1}} - \underbrace{\frac{I_a I_\tau}{I_b}}_{I_{b2}} - \underbrace{\frac{I_{\gamma\tau} I_\rho}{I_1}}_{I_{b3}} + \underbrace{\frac{I_{\gamma\tau} I_r}{I_1}}_{I_{b4}} + \underbrace{\frac{I_{\gamma\tau} I_f I_\rho}{I_b I_1}}_{I_{b5}} - \underbrace{\frac{I_{\gamma\tau} I_f I_r}{I_b I_1}}_{I_{b6}} \tag{4.25}$$

and

$$\begin{aligned} \text{KCL}: \quad & I_{Cb} = I_{b1} - I_{b2} - I_{b3} + I_{b4} + I_{b5} - I_{b6} \\ \text{TL}: \quad & I_{b1} I_b = I_f I_\tau \quad I_{b2} I_b = I_a I_\tau \quad I_{b3} I_1 = I_{\gamma\tau} I_\rho \\ & I_{b4} I_1 = I_{\gamma\tau} I_r \quad I_{b5} I_b I_1 = I_{\gamma\tau} I_f I_\rho \quad I_{b6} I_b I_1 = I_{\gamma\tau} I_f I_r \,. \end{aligned} \tag{4.26}$$

4.2.3 Translinear Loops

We can configure the TL equations for the "a" side of the dynamic constraints in the following order:

$$\begin{array}{ccccccccccc} & & & & I_a & \leftrightarrow & I_f & \leftrightarrow & I_{a1} & \leftrightarrow & I_\tau \\ & & & & I_a & \leftrightarrow & I_b & \leftrightarrow & I_{a2} & \leftrightarrow & I_\tau \\ & & & & I_{a3} & \leftrightarrow & I_\rho & \leftrightarrow & I_1 & \leftrightarrow & I_{\gamma\tau} \\ & & & & I_{a4} & \leftrightarrow & I_r & \leftrightarrow & I_1 & \leftrightarrow & I_{\gamma\tau} \\ I_a & \leftrightarrow & I_f & \leftrightarrow & I_{a5} & \leftrightarrow & I_\rho & \leftrightarrow & I_1 & \leftrightarrow & I_{\gamma\tau} \\ I_a & \leftrightarrow & I_f & \leftrightarrow & I_{a6} & \leftrightarrow & I_r & \leftrightarrow & I_1 & \leftrightarrow & I_{\gamma\tau}\,. \end{array}$$

The MITE network for this ordering is shown in Fig. 4.3. The "b" side TL equations can be arranged in an ordering that is almost identical to the "a" ordering as follows:

$$\begin{array}{ccccccccccc} & & & & I_b & \leftrightarrow & I_f & \leftrightarrow & I_{b1} & \leftrightarrow & I_\tau \\ & & & & I_b & \leftrightarrow & I_a & \leftrightarrow & I_{b2} & \leftrightarrow & I_\tau \\ & & & & I_{b3} & \leftrightarrow & I_\rho & \leftrightarrow & I_1 & \leftrightarrow & I_{\gamma\tau} \\ & & & & I_{b4} & \leftrightarrow & I_r & \leftrightarrow & I_1 & \leftrightarrow & I_{\gamma\tau} \\ I_b & \leftrightarrow & I_f & \leftrightarrow & I_{b5} & \leftrightarrow & I_\rho & \leftrightarrow & I_1 & \leftrightarrow & I_{\gamma\tau} \\ I_b & \leftrightarrow & I_f & \leftrightarrow & I_{b6} & \leftrightarrow & I_r & \leftrightarrow & I_1 & \leftrightarrow & I_{\gamma\tau}\,. \end{array}$$

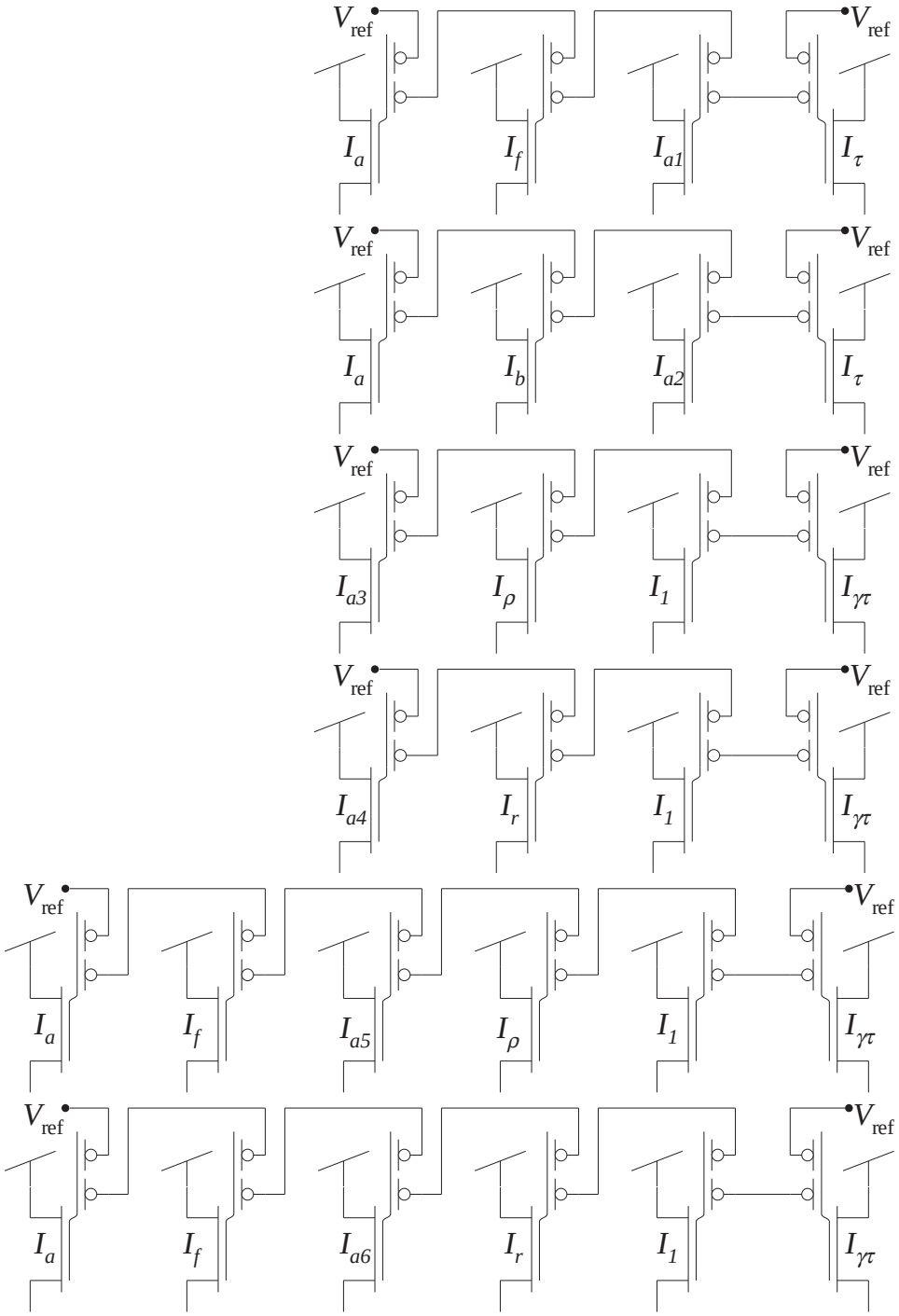


Figure 4.3: Initial connections for the "a" side of the dynamic constraint network.

This ordering was chosen from many possible choices by looking at the required inputs and the similarities of the "a" and "b" sides. The most notable of these similarities is that both sides include the $I_{\gamma\tau}$↔I_1 factor in eight of the twelve total TL equations. (The above ordering was determined by using the Perl program in Appendix C.)

4.2.4 Consolidation

Having taken the time to arrange the TL equations to maximize the chances for consolidation, we can now remove redundant terms. The highlighted currents below indicate which factors or combinations can be removed, because they have already appeared. For clarity, the list of shared terms are: I_a, I_a↔I_f, I_τ↔I_1, $I_{\gamma\tau}$↔I_1, $I_{\gamma\tau}$↔I_1↔I_ρ, and $I_{\gamma\tau}$↔I_1↔I_r.

$$
\begin{array}{cccccccccccc}
 & & & & I_a & \leftrightarrow & I_f & \leftrightarrow & I_{a1} & \leftrightarrow & I_\tau \\
 & & & & \boldsymbol{I_a} & \leftrightarrow & I_b & \leftrightarrow & I_{a2} & \leftrightarrow & \boldsymbol{I_\tau} \\
 & & & & I_{a3} & \leftrightarrow & I_\rho & \leftrightarrow & I_1 & \leftrightarrow & I_{\gamma\tau} \\
 & & & & I_{a4} & \leftrightarrow & I_r & \leftrightarrow & \boldsymbol{I_1} & \leftrightarrow & \boldsymbol{I_{\gamma\tau}} \\
\boldsymbol{I_a} & \leftrightarrow & \boldsymbol{I_f} & \leftrightarrow & I_{a5} & \leftrightarrow & \boldsymbol{I_\rho} & \leftrightarrow & \boldsymbol{I_1} & \leftrightarrow & \boldsymbol{I_{\gamma\tau}} \\
\boldsymbol{I_a} & \leftrightarrow & \boldsymbol{I_f} & \leftrightarrow & I_{a6} & \leftrightarrow & \boldsymbol{I_r} & \leftrightarrow & \boldsymbol{I_1} & \leftrightarrow & \boldsymbol{I_{\gamma\tau}}
\end{array}
$$

$$
\begin{array}{cccccccccccc}
 & & & & I_b & \leftrightarrow & I_f & \leftrightarrow & I_{b1} & \leftrightarrow & I_\tau \\
 & & & & \boldsymbol{I_b} & \leftrightarrow & I_a & \leftrightarrow & I_{b2} & \leftrightarrow & \boldsymbol{I_\tau} \\
 & & & & I_{b3} & \leftrightarrow & I_\rho & \leftrightarrow & I_1 & \leftrightarrow & I_{\gamma\tau} \\
 & & & & I_{b4} & \leftrightarrow & I_r & \leftrightarrow & \boldsymbol{I_1} & \leftrightarrow & \boldsymbol{I_{\gamma\tau}} \\
\boldsymbol{I_b} & \leftrightarrow & \boldsymbol{I_f} & \leftrightarrow & I_{b5} & \leftrightarrow & \boldsymbol{I_\rho} & \leftrightarrow & \boldsymbol{I_1} & \leftrightarrow & \boldsymbol{I_{\gamma\tau}} \\
\boldsymbol{I_b} & \leftrightarrow & \boldsymbol{I_f} & \leftrightarrow & I_{b6} & \leftrightarrow & \boldsymbol{I_r} & \leftrightarrow & \boldsymbol{I_1} & \leftrightarrow & \boldsymbol{I_{\gamma\tau}}
\end{array}
$$

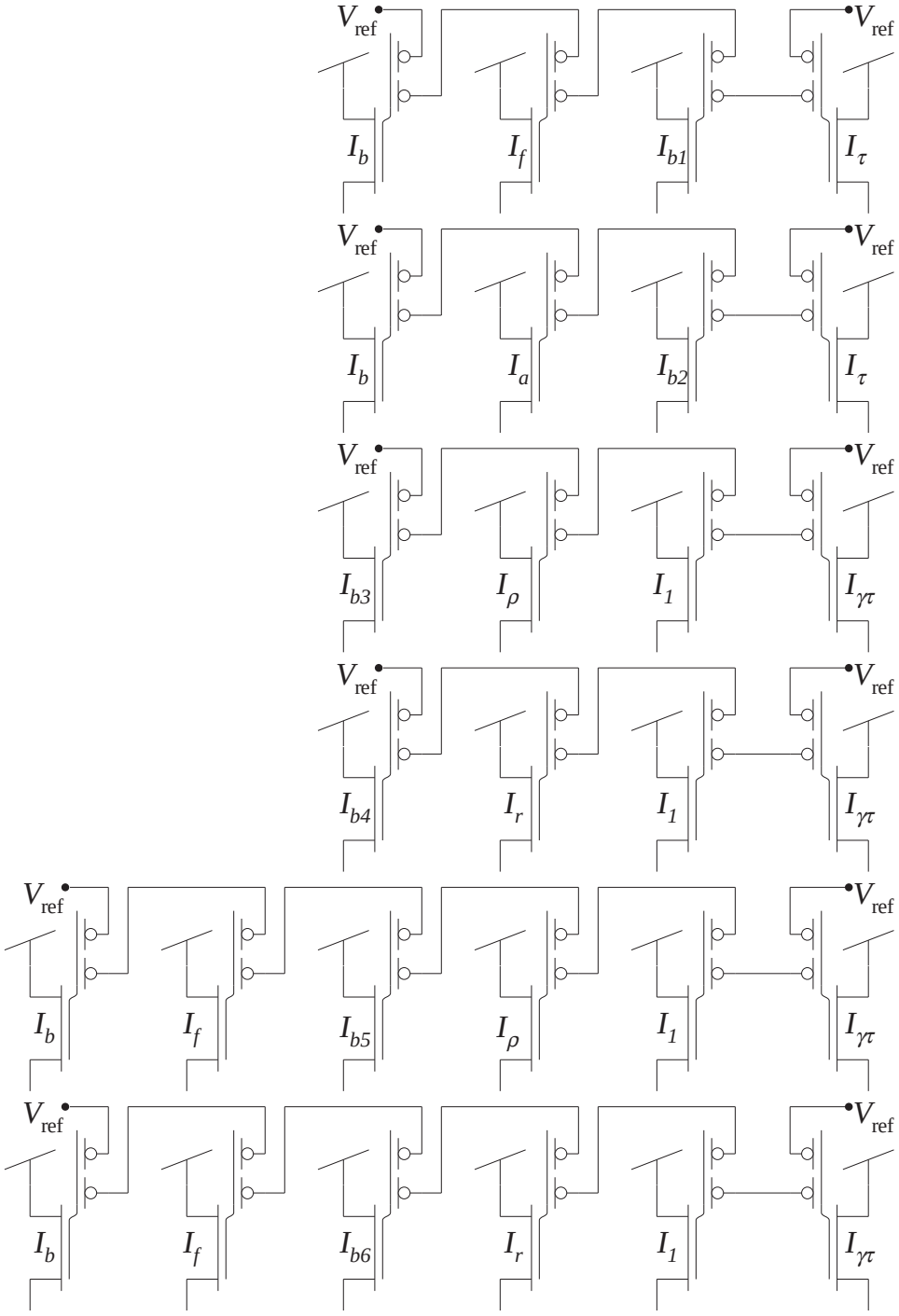


Figure 4.4: Initial connections for the "b" side of the dynamic constraint network.

Because the "a" and "b" sides share some similar terms, we can share a voltage from one to remove the MITEs in the other that are used to generate that voltage. These reductions are shown in Figs. 4.5 and 4.6. (Even though $I_{a3} = I_{b3}$ and $I_{a4} = I_{b4}$, we cannot remove the MITEs that generate these currents since they are required in distinct KCL equations.)

4.2.5 Biasing

Figure 4.7 shows the MITE networks of Figs. 4.5 and 4.6 rearranged into two connected rows for biasing. In order to bias the network, we can add current sources for all of the MITEs except the I_{ai}, I_{bi}, I_a, and I_b ones. Using the KCL equations,

$$\begin{aligned} \text{KCL}: \quad & I_{Ca} = -I_{a1} + I_{a2} - I_{a3} + I_{a4} + I_{a5} - I_{a6} \\ \text{KCL}: \quad & I_{Cb} = -I_{b1} + I_{b2} - I_{b3} + I_{b4} + I_{b5} - I_{b6}\,, \end{aligned} \tag{4.27}$$

we can add a capacitor to each side and sum the currents appropriately while mirroring them around to enforce the KCL constraints. Since we have two MITEs passing I_a and two passing I_b (the outputs), we bias these pairs with a set of NMOS current mirrors each. The fully biased circuit is shown in Fig. 4.8.

4.2.6 Diode Connections

The diode connections for this circuit follow the same kind of pattern as before. Starting at the left, we can choose the first MITE in each row to be the "output" MITE and diode connect around the NMOS to generate the voltages required for the mirrors. Connecting around the next two I_f MITEs leaves us at nodes that are part of the KCL constraints. Because this system is large, we skip these nodes until later. The next set of MITES are passing the output currents. Since we have

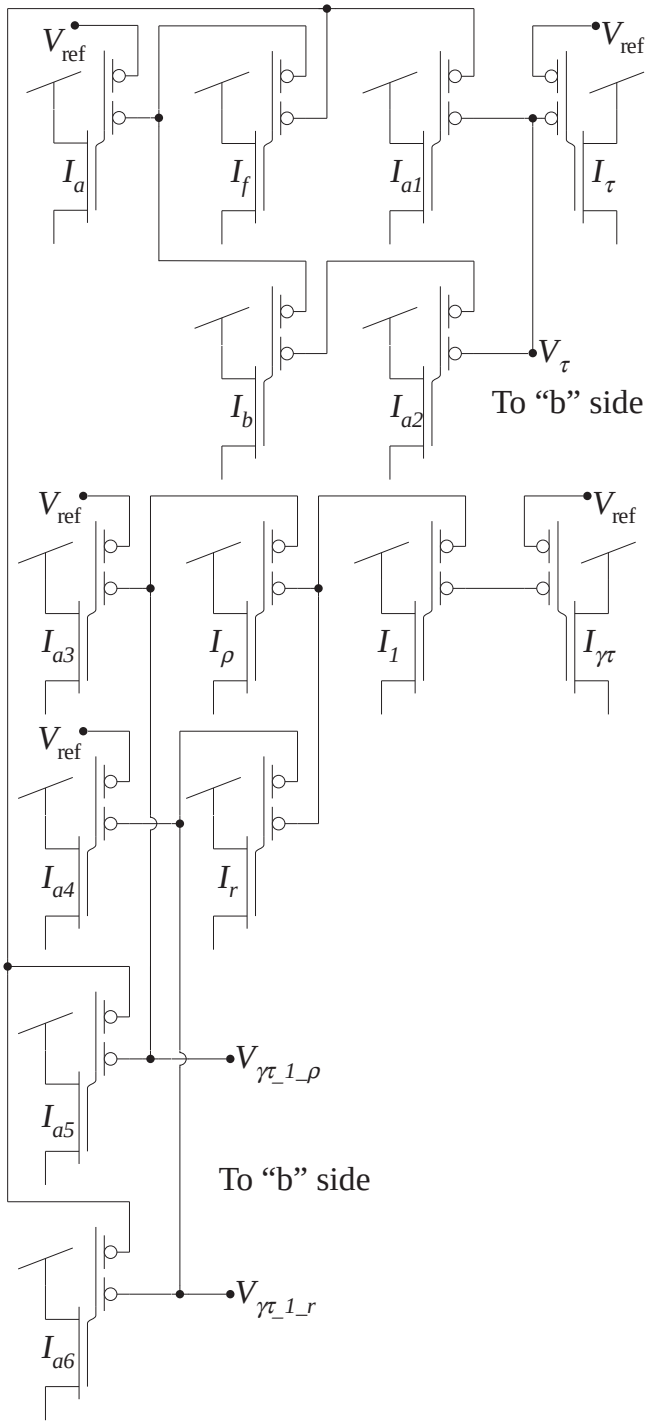


Figure 4.5: Consolidations for the "a" side of the dynamic constraint network.

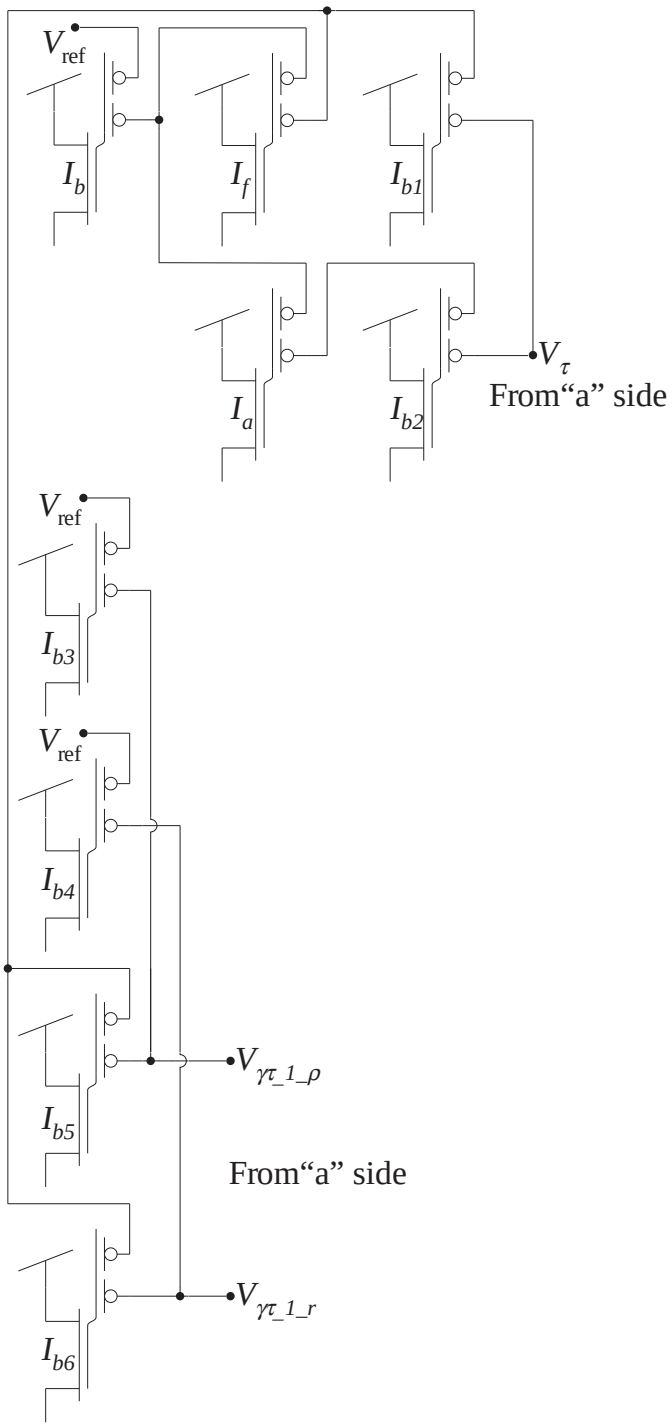


Figure 4.6: Consolidations for the “b” side of the dynamic constraint network.

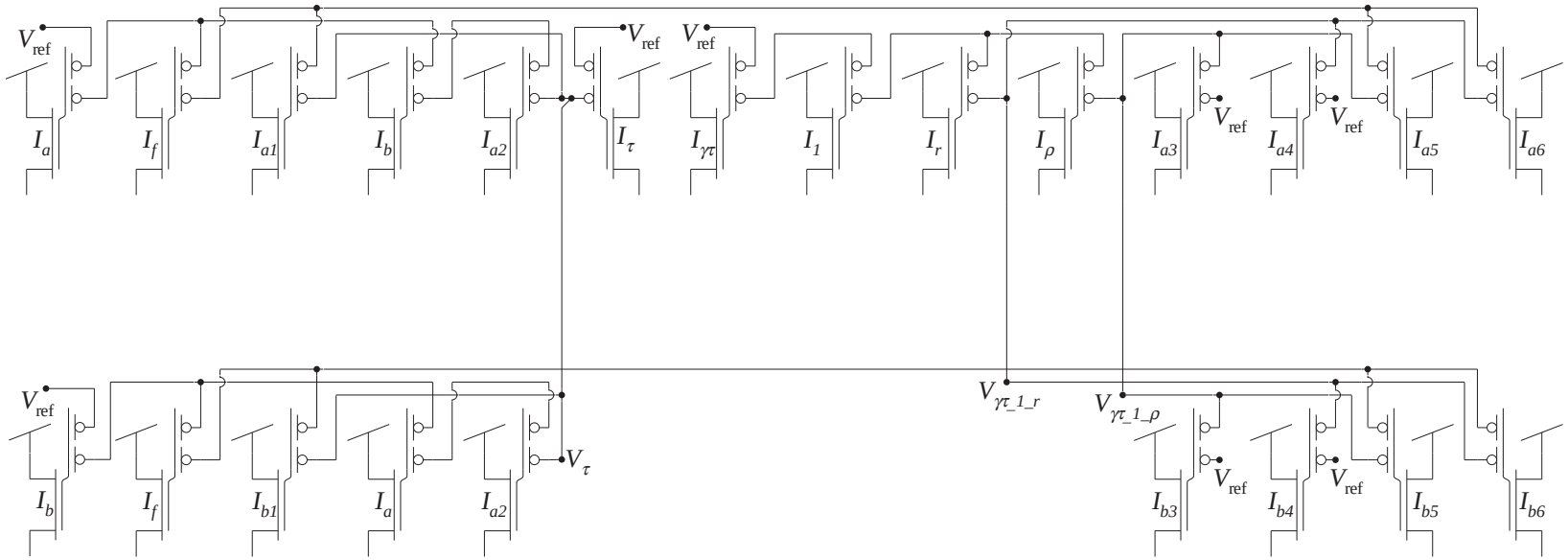


Figure 4.7: Rearranged consolidated network for both the “a” and “b” sides.

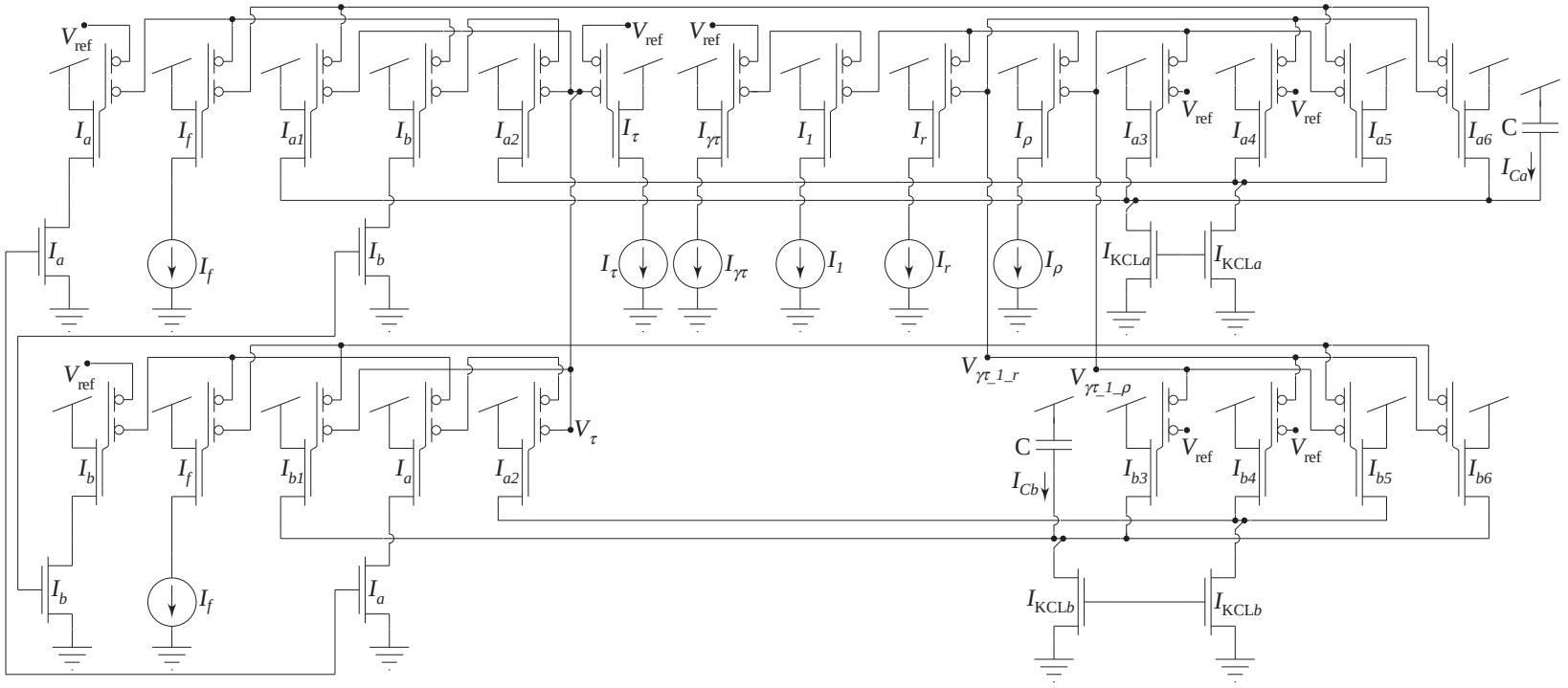


Figure 4.8: Rearranged consolidated network for both the "a" and "b" sides where the voltages, V_a and V_b, are shared from the radius calculation network.

already diode connected the other half of the mirror around the NMOS transistors, we diode connect around these MITEs. Skipping more KCL nodes, we then diode connect around the input current MITEs passing I_τ, $I_{\gamma\tau}$, I_1, I_r, and I_ρ. With just the KCL nodes remaining, we can diode connect around the I_{a6} and I_{b6} MITEs which also forces diode connections around the other half of the NMOS current mirrors. These diode connections are shown in Fig. 4.9.

Looking back at the vector magnitude circuit of Fig. 2.8, we observe that there are four places where the output currents of the dynamic networks are required. Since we already have voltages that represent log-compressed currents for I_a and I_b (labeled V_a and V_b in Fig. 4.8), we can remove the two input MITEs on the right end of the radius calculation network. Recognizing that I_a and I_b are not actual input current sources, we replace the two remaining current sources with NMOS transistors that mirror the output currents from the dynamic side of the system. Similarly, we need to mirror I_r from the radius calculation side to the dynamic side. Since we have not already mirrored I_r and do not even have a MITE passing this output current, we must generate this current by adding a MITE that sources I_r into a diode connected NMOS transistor allowing us to mirror it to the dynamic side. These changes, completing the oscillator circuit, are shown in Fig. 4.10.

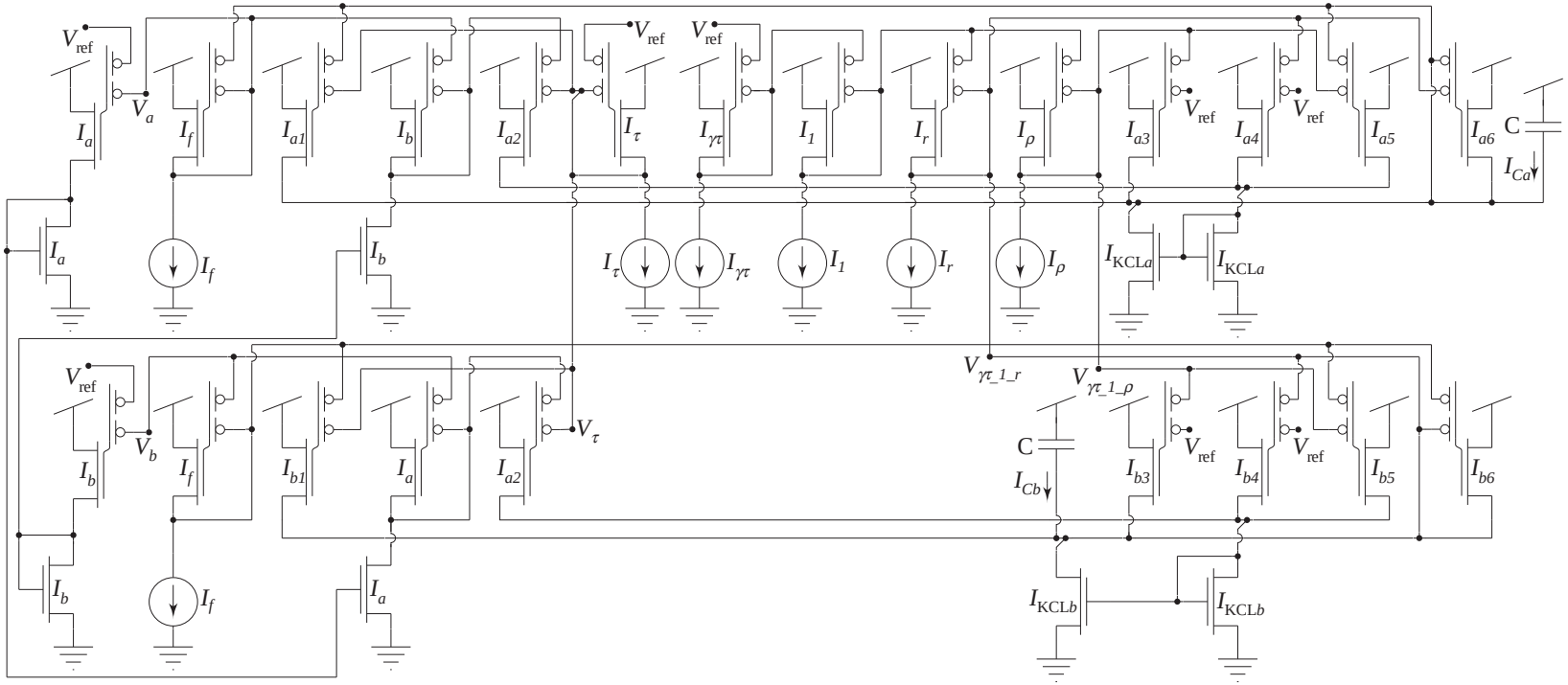


Figure 4.9: Diode connections for the dynamic constraints. Voltages V_a and V_b represent log-compressed currents and can be used to remove the two input MITEs in the radius calculation network.

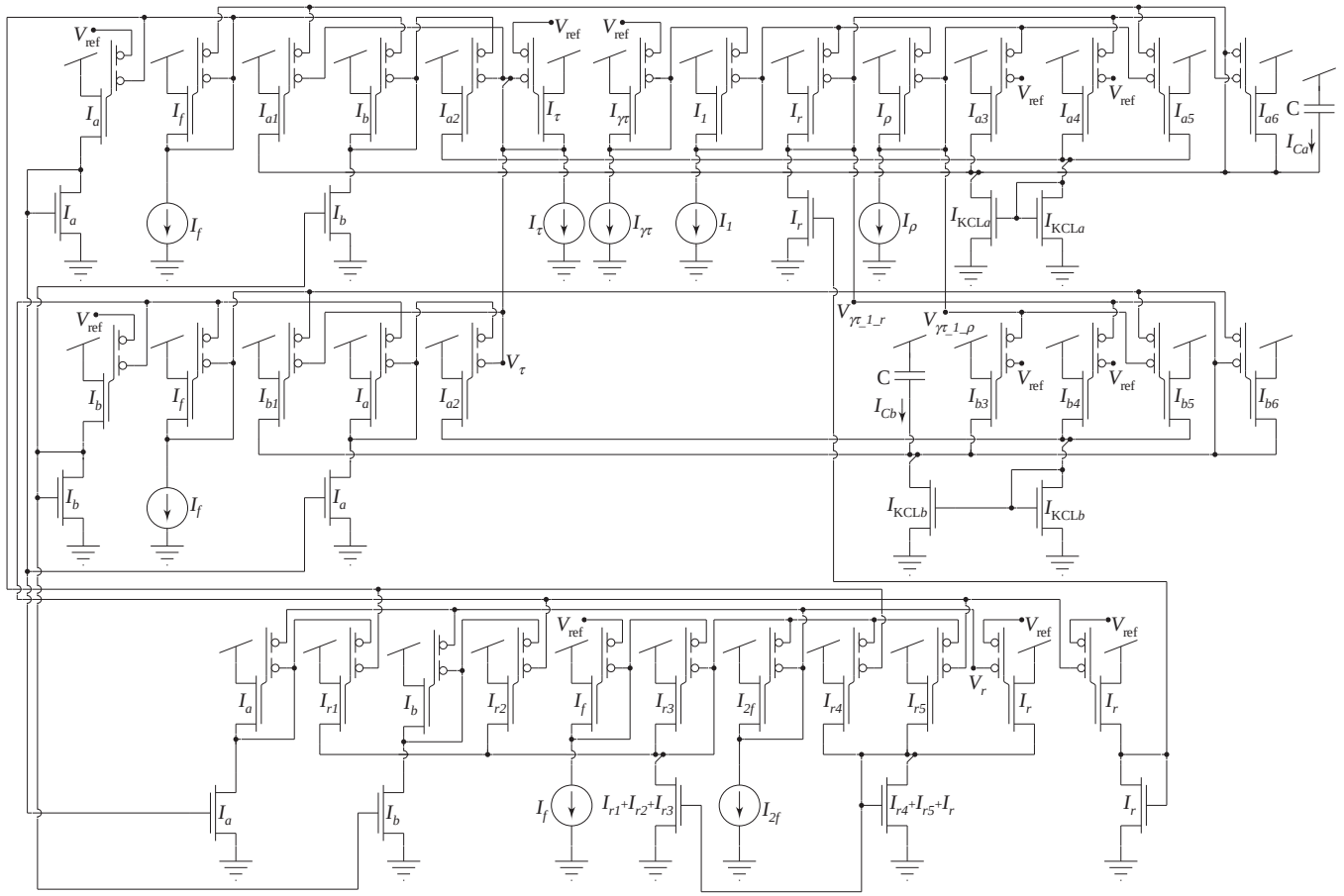


Figure 4.10: Final changes to complete the entire oscillator circuit linking the radius and dynamic sides.

Chapter 5

Phase-Locked Loop

5.1 System Decomposition

This final example illustrates how multiple MITE networks can be combined by integrating complete smaller networks into a larger complex system. We demonstrate this process by designing a phase-locked loop (PLL), as shown in Fig. 5.1. The input signal is expected to be a sinusoid whose frequency changes slowly in time. The feedback loop is expected to adjust the oscillator's output frequency to match that of the input by examining the phase difference between the two signals. When the phase difference becomes constant, the PLL is said to be "locked" onto the input signal's frequency.

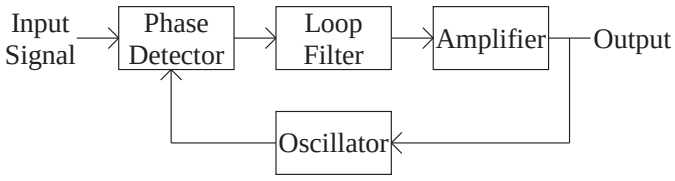


Figure 5.1: Phase-locked loop block diagram.

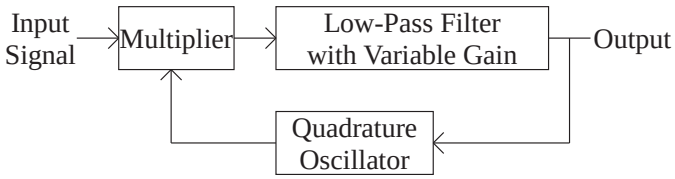


Figure 5.2: Phase-locked loop block diagram.

The phase detector can be realized by a simple multiplier resulting in a low-frequency component representing the frequency difference between the input and the oscillator's output. There will also be a high-frequency component (at approximately twice the input's frequency) that will be removed by the loop filter. A first-order low-pass filter is sufficient to accomplish this filtering operation. By introducing a variable gain into the low-pass filter, we can combine both the loop filter and amplifier into a single circuit as shown in Fig. 5.2. The quadrature oscillator from the previous chapter is sufficient for this system.

5.2 Multiplier

This section describes the process by which we transform the polynomial constraint for a multiplier,

$$z = xy\,, \tag{5.1}$$

into the necessary translinear loops. Because both the inputs and the output need to represent positive and negative values, we must introduce offsets to force the variables to be positive. Doing so, we obtain

$$\begin{aligned} & a = x + f\,, \quad b = y + f\,, \quad \text{and} \quad c = z + f \\ \Rightarrow \quad & (c - f) = (a - f)\,(b - f)\,. \end{aligned} \tag{5.2}$$

The dimensionless variables are replaced with ratios of signal currents to a unit current.

$$I_c - I_f = \frac{(I_a - I_f)\,(I_b - I_f)}{I_1} \tag{5.3}$$

Solving for the output current, I_c, defining I_{2f} as $2I_f$, and equating I_f to I_1 (to help reduce the number of separate bias currents), results in

$$I_c = \underbrace{\frac{I_a I_b}{I_f}}_{I_{\mathrm{TL}}} - I_a - I_b + \underbrace{2I_f}_{I_{2f}} \tag{5.4}$$

which can be represented by the following TL and KCL equation:

$$\begin{aligned} \text{KCL}: &\quad I_c = I_{\mathrm{TL}} - I_a - I_b + I_{2f} \\ \text{TL}: &\quad I_{\mathrm{TL}} I_f = I_a I_b\,. \end{aligned} \tag{5.5}$$

Having already constructed much more complicated MITE networks, this multiplier circuit should seem trivial. Arranging the TL equation as

$$I_a \leftrightarrow I_f \leftrightarrow I_b \leftrightarrow I_{\mathrm{TL}}\,, \tag{5.6}$$

we connect the MITEs, as shown in Fig. 5.3(a). We then bias with three input current sources on the first three MITEs and add several more current sources and an NMOS current mirror according to the KCL equation. We have added the current mirror to generate an usable copy of the output current, I_c. The biased circuit is shown in Fig. 5.3(b). Since we have to mirror the output current, we must diode connect around the left NMOS transistor. Diode connecting around the three input MITEs completes the multiplier circuit, as shown in Fig. 5.3(c).

5.3 Low-Pass Filter

The low-pass filter detailed in Section 3.2 could be used for the PLL loop filter if we could control the gain. Considering the transfer function for a low-pass filter

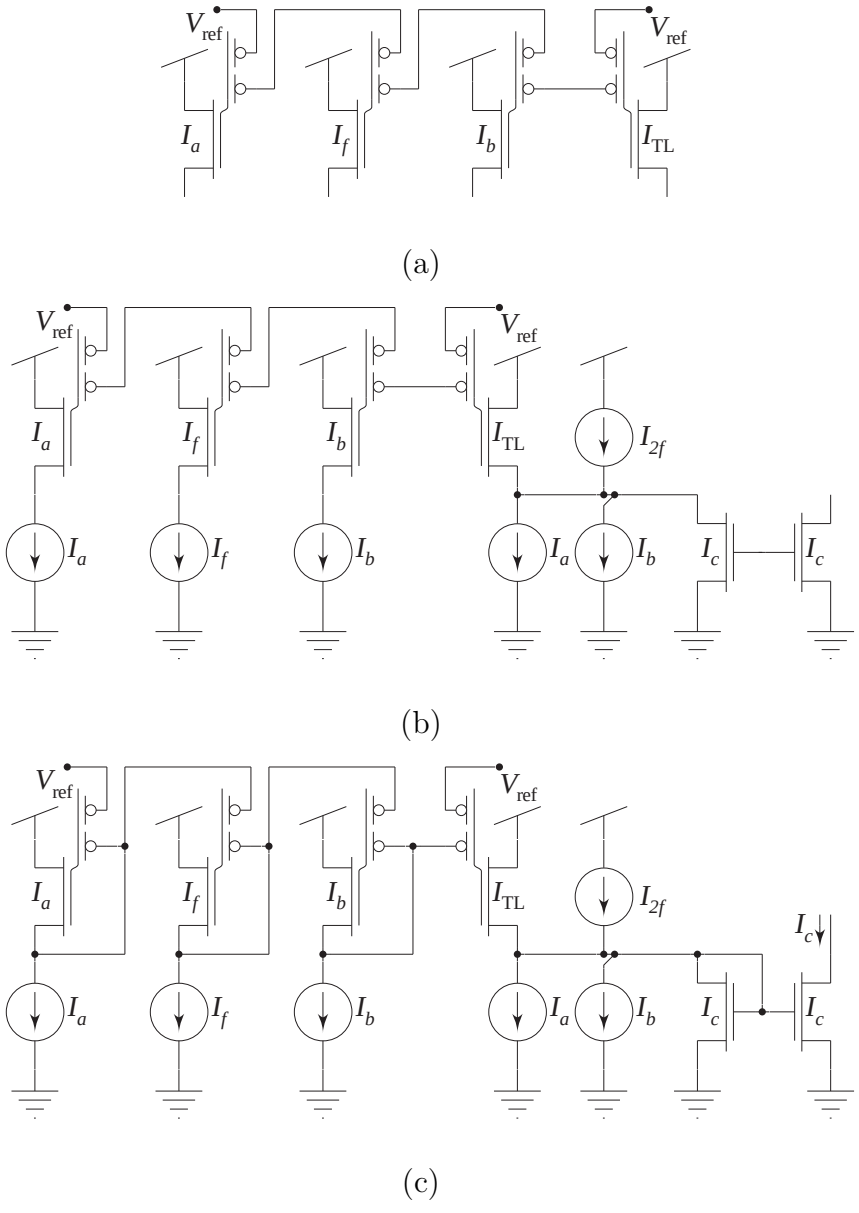


Figure 5.3: Construction of the multiplier circuit. (a) Initial connections. (b) Biasing for the circuit. (c) Completed circuit.

with a DC gain of k,

$$H(s) = \frac{k}{1+\tau s}\,, \tag{5.7}$$

rearranging to find

$$\tau s y(s) = kx(s) - y(s) \tag{5.8}$$

allows us to use the inverse Laplace transform to get the differential equation for a first-order low-pass filter with gain, k,

$$\tau \frac{dy}{dt} = kx - y\,. \tag{5.9}$$

Replacing the variables with current ratios, we find that

$$\tau \frac{d}{dt}\left(\frac{I_y}{I_1}\right) = k\frac{I_x}{I_1} - \frac{I_y}{I_1}\,, \tag{5.10}$$

which we can reduce to

$$\tau \frac{dI_y}{dt} = kI_x - I_y\,. \tag{5.11}$$

Using an inverting output structure, we replace the derivative to find

$$-\frac{I_{\mathrm{C}} I_y}{I_\tau} = kI_x - I_y\,, \tag{5.12}$$

which becomes

$$I_{\mathrm{C}} = I_\tau - \underbrace{\frac{kI_\tau I_x}{I_y}}_{I_{\mathrm{TL}}}\,. \tag{5.13}$$

Absorbing the gain factor, k, into one of the I_τ's, we get the following KCL and TL equations.

$$\begin{aligned} \text{KCL}: &\quad I_{\mathrm{C}} = I_\tau - I_{\mathrm{TL}} \\ \text{TL}: &\quad I_{\mathrm{TL}} I_y = I_{k\tau} I_x \end{aligned} \tag{5.14}$$

Recognizing that the final decomposition is almost identical to that of the filter described in Section 3.2, we can simply use the same circuit by just varying the rightmost current source to be $I_{k\tau}$ instead of I_τ. This circuit is shown in Fig. 5.4.

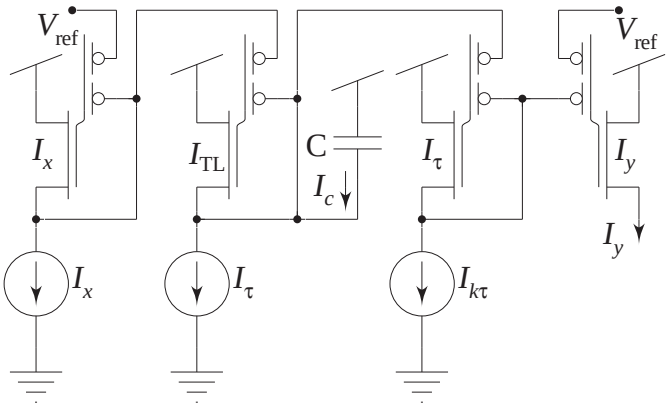


Figure 5.4: Modified low-pass filter from Section 3.2 to include a gain of k.

5.4 Inter-Network Connections

Now that we have the phase detector (multiplier), loop filter and amplifier (modified low-pass filter), and an oscillator (quadrature oscillator), we can connect them all to form the PLL. Starting with the multiplier, we chose to have the external input be defined as I_{in} (replacing I_a) and the output of the oscillator that is fed back to the phase detector as I_{osc}. Since either of the oscillator's outputs will work (only a 90 degrees phase shift between them), we choose to use the I_a output. Since the multiplier is expecting two current sinks passing I_{osc}, we can replace the I_b current sources with NMOS transistors whose gates are tied to the diode connected NMOS from the oscillator circuit that is passing the I_b output current. The relevant sections of the circuits are shown in Fig. 5.5.

The output of the multiplier can be passed to the input of the loop filter in a similar manner. Since we already have an NMOS transistor passing the multiplier's output current and the filter is expecting the input to be supplied as a current sink, we can replace the input current source of the filter with the NMOS transistor from

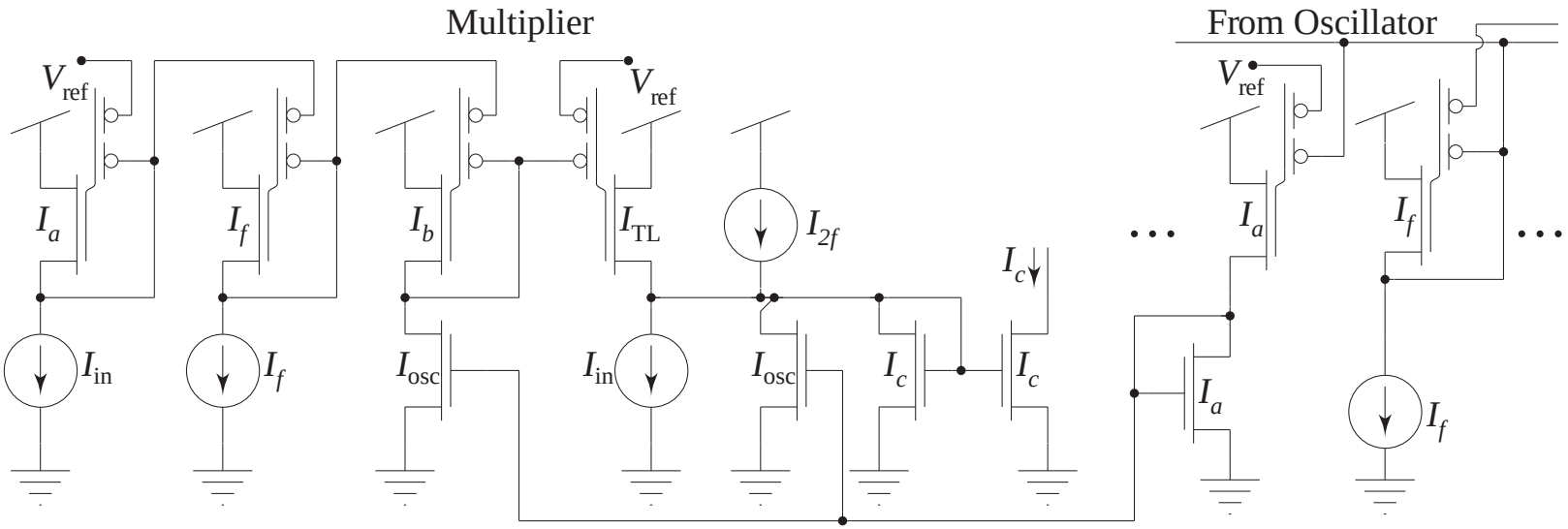


Figure 5.5: Connecting the output of the oscillator to the second input of the multiplier.

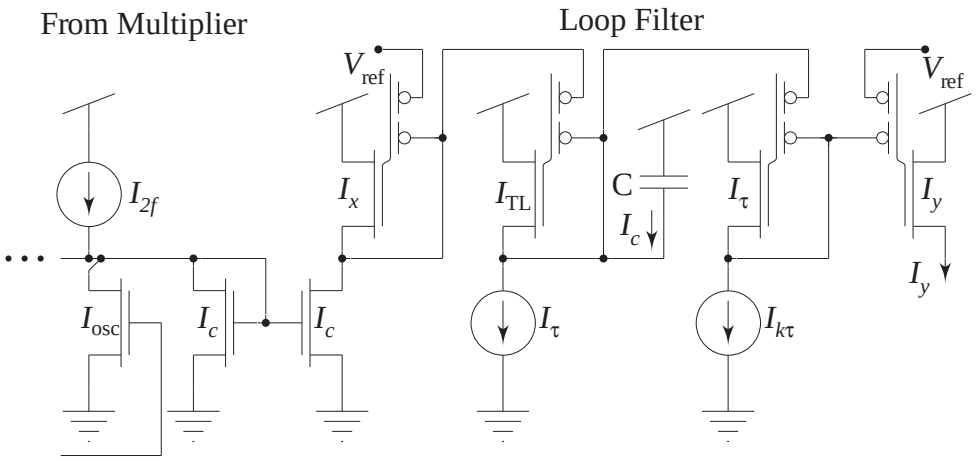


Figure 5.6: Connecting the output of the oscillator to the second input of the multiplier.

the multiplier, as shown in Fig. 5.6. Note that the output of the multiplier, I_c, becomes the input of the filter, I_x.

The output of the loop filter, I_y, becomes the input to the oscillator, I_τ. The oscillator is expecting a current sink passing I_τ, so we can mirror the output of the filter using two NMOS transistors. However, we also need a scaled version of I_τ, $I_{\gamma\tau} = \gamma I_\tau$.

We approach calculating $I_{\gamma\tau}$ just as we would any other function. Beginning by replacing the dimensionless variable, γ, with a current ratio, we find that

$$I_{\gamma\tau} = \frac{I_\gamma}{I_1} I_\tau \,. \tag{5.15}$$

We can then rearrange Eq. 5.15 into the following TL equation:

$$\text{TL}: \quad I_{\gamma\tau} I_1 = I_\gamma I_\tau \tag{5.16}$$

Arranging the currents into the order,

$$I_\tau \leftrightarrow I_{\gamma\tau} \leftrightarrow I_\gamma \leftrightarrow I_1 \,, \tag{5.17}$$

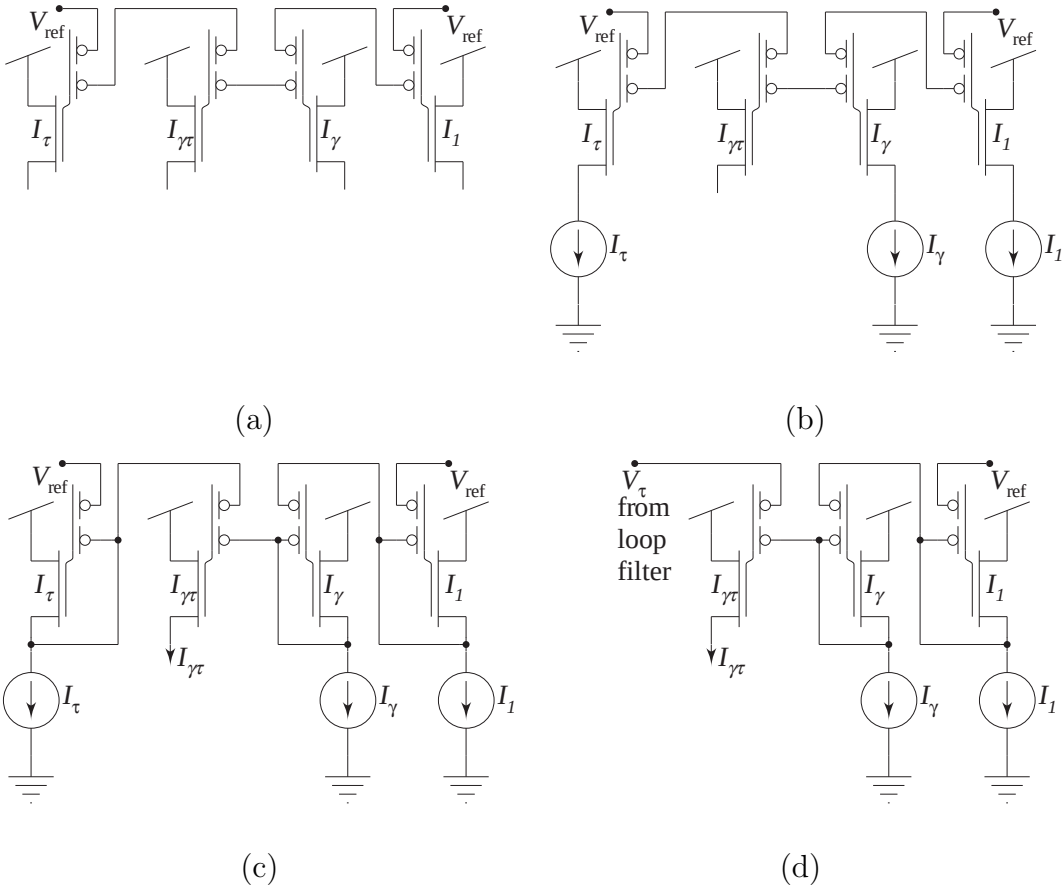


Figure 5.7: Additional circuit to generate the γ-scaled version of I_τ.

we can connect the MITEs, as shown in Fig. 5.7(a). The biasing and diode connections for the circuit are shown in Fig. 5.7(b) and (c). Recognizing that the input current, I_τ, is the output current of the loop filter, we are able to share the voltage to remove the input MITE, as shown in Fig. 5.7(d). Figure 5.8 shows the connections from the loop filter that are used to generate the required I_τ and $I_{\gamma\tau}$.

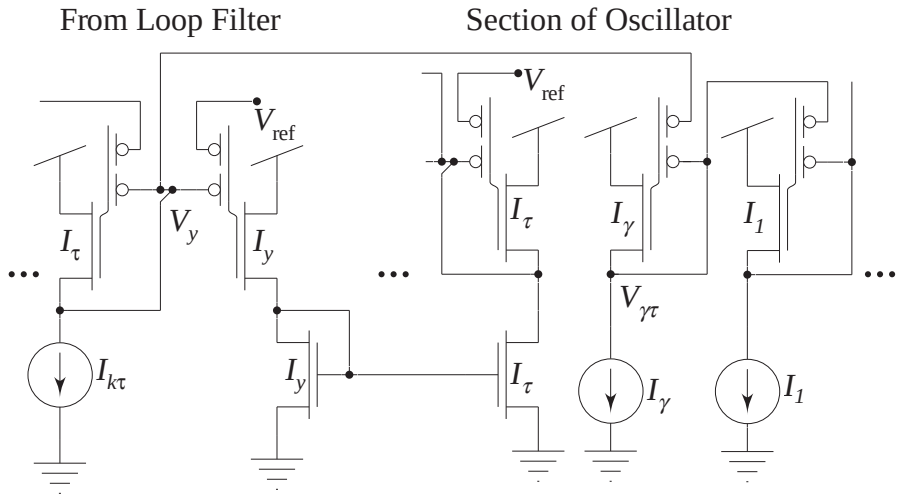


Figure 5.8: Connecting the output of the filter, I_y, to the oscillator to generate both I_τ and the scaled version, $I_{\gamma\tau}$.

Chapter 6

Results and Conclusions

The following sections present results from the majority of the circuits presented in the previous chapters. I present each circuit's results separately and address global issues in Section 6.7. Comparing the results found in this dissertation to the results of similar circuits would only mislead the reader because the comparisons would rarely be fair. The majority of translinear circuits are implemented using bipolar junction transistors fabricated in a BiCMOS process allowing for much higher current levels ($\sim$milliamperes) and thus, higher frequencies. All results in this dissertation are measured from circuits implemented with floating-gate PMOS transistors operated in weak inversion (limiting current levels to a maximum of approximately 100nA). It follows that BiCMOS implementations will operate for higher frequencies but require more power than their MITE network counterparts. Additionally, signal-to-noise (SNR) ratios are not quoted in the results because for these large signal circuits, the mere definition of the SNR becomes ambiguous and the measurement is difficult. (Noise levels are dependent upon the signal levels and therefore, the best SNR will most likely not be found for the maximum allowable signal levels.)

Alternate implementations of translinear circuits can be found in [4,6–8,15,30, 32,36] for log-domain filters, in [9,28] for RMS-to-DC converters, in [33,41,45] for oscillators, and in [40,42,45] for phase-locked loops.

6.1 Vector Magnitude Results

Data collected from the circuit described in Section 2.2 is shown in Fig. 6.1. The vector magnitude was calculated for values of I_x and I_y over the range of 1nA to 50nA and a Vdd of 2V. The MITEs were programmed to pass a nominal current of 10nA with control gate voltages at 1V (under a 1% variance in current at that operating point). Investigation into the reason for the error in the results lead to a discovery that the subthreshold slopes of the floating-gate transistors did not match. See Section 6.7 for a detailed discussion of reasons for error in the collected data.

6.2 First-Order Low-Pass Filter Results

The frequency response for the first-order low-pass filter described in Section 3.2 is shown in Fig. 6.2. Data was collected for five values of I_τ (corner frequencies ranging from 3kHz to approximately 13kHz for values of I_τ from 5nA to 150nA). Evidence of higher order effects start appearing above 10kHz preventing the phase to level off at the expected -90 degrees and altering the roll-off rate in the magnitude response. This is probably the result of higher order effects or feed through from various control gates to others. What appears to be a double-zero around 16kHz is most likely a direct feed through of the input through either the off-chip circuitry or the global reference signal, V_{ref}. See Section 6.7 for a more detailed

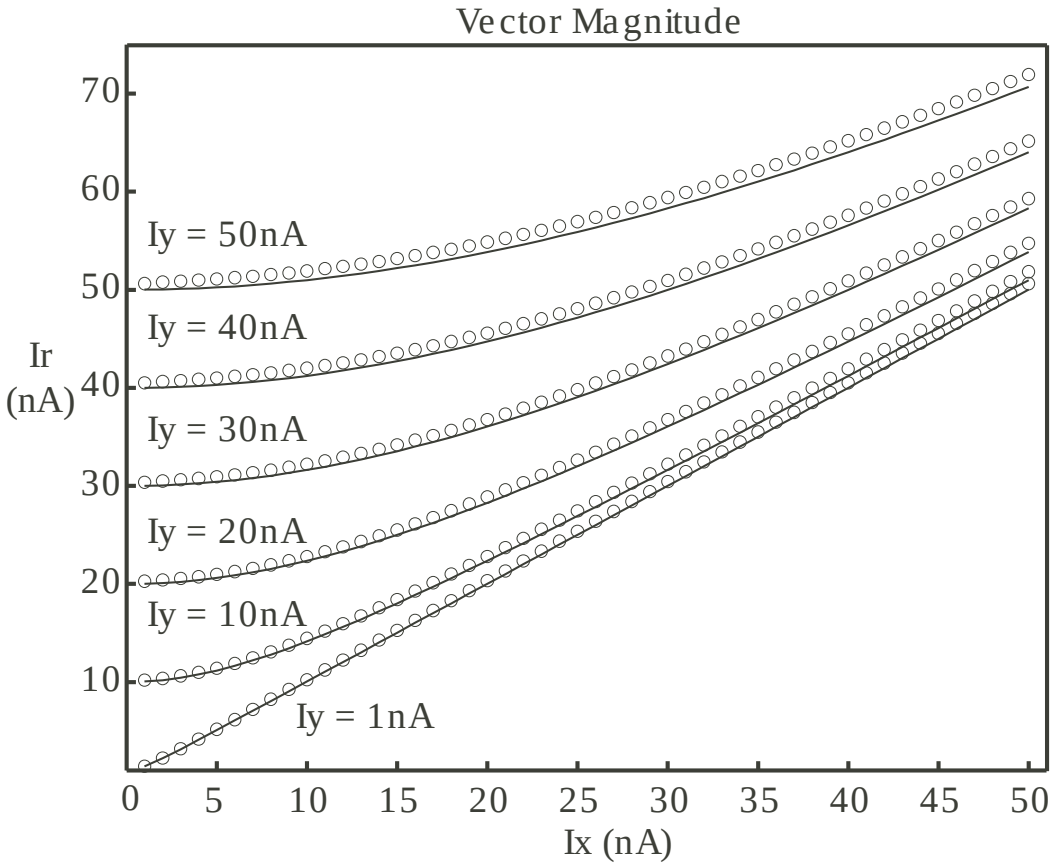


Figure 6.1: Measured data from the vector magnitude circuit. Measured data is shown with circles and the ideal curves are shown with solid lines.

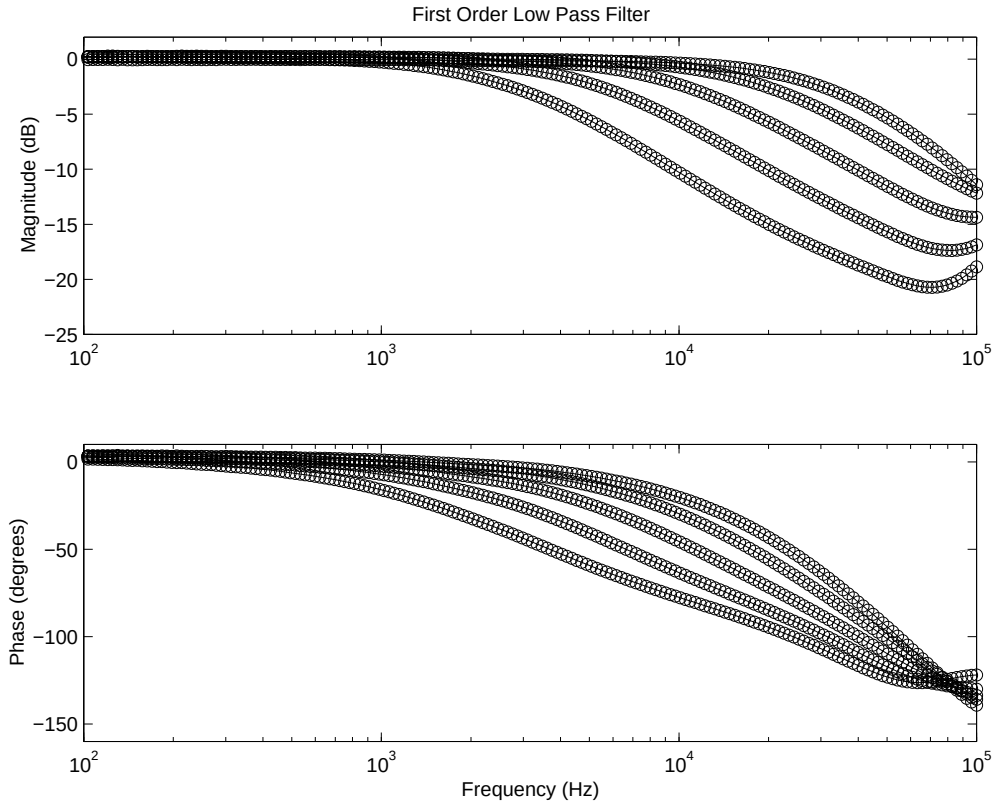


Figure 6.2: Frequency response for a first order low-pass filter.

discussion of experimental results.

6.3 RMS-to-DC Converter Results

Figures 6.3 and 6.4 show input and output traces measured from the RMS-to-DC converter along with the ideal expected value. Due to high frequency feed through (>10kHz) and a limited range of corner frequencies (>1kHz, limited by the on-chip capacitor and a minimum value for I_τ) it was not possible to completely filter out the AC variations of the squared input signal. Considering that a first order low-pass filter can only approximate the mean of a signal, the circuit performs within

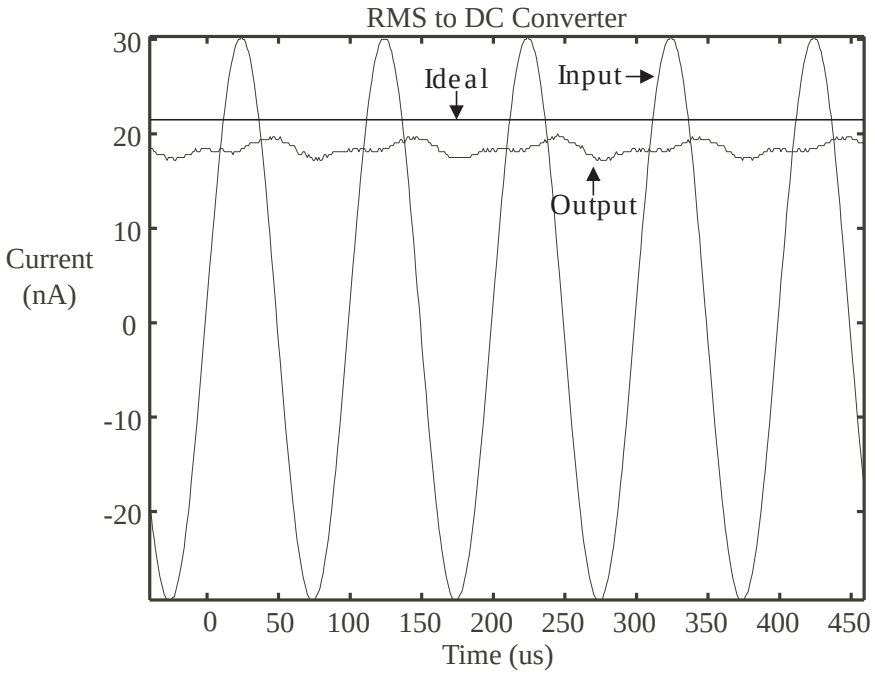


Figure 6.3: Results from the RMS-to-DC converter circuit with a sinusoidal input signal.

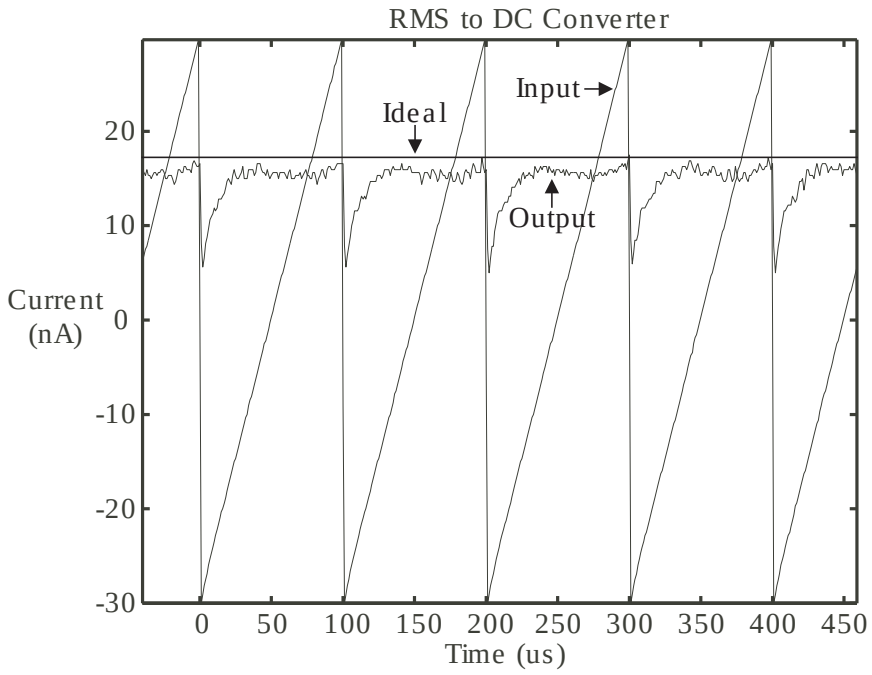


Figure 6.4: Results from the RMS-to-DC converter circuit with a sawtooth input signal.

expectations. An obvious gain or offset error can be seen that may be the result of mismatch or transistors coming out of saturation. Results may be very sensitive to operating levels since this circuit must be able to handle a wide range of current levels. The input is initially squared, creating a large current which can force transistors into non-ideal operating conditions (the input current is not centered about zero since an offset is required to ensure strictly positive currents). The time constant can be observed in Fig. 6.4 by examining the output after sharp changes in the input. While no formal comparison to alternate implementations of RMS-to-DC converters is presented here, it is worthwhile to note that the implementation detailed in Section 3.3 does not assume a rectified input signal as do most published implementations. Additional RMS-to-DC converters are published in [28] and [9].

6.4 Second-Order Low-Pass Filter Results

Frequency responses for the second-order low-pass filter of Section 4.1 are shown in Figs. 6.5, 6.6, and 6.7 for three values of I_τ (three corner frequencies of about 4kHz, 8kHz, and 10kHz). Each plot shows the responses for various quality factors (0.25, 0.5, 1, and 2). What appears to be a double-zero around 11kHz is most likely a direct feed through of the input through either the off-chip circuitry or the global reference signal, V_{ref}. See Section 6.7 for a more detailed discussion of this anomaly.

6.5 Quadrature Oscillator Results

Plots of various experimental results from the quadrature oscillator of Section 4.2 fabricated in an AMI 0.5- μm process are shown in Figs. 6.8 – 6.12. Figures 6.8 –

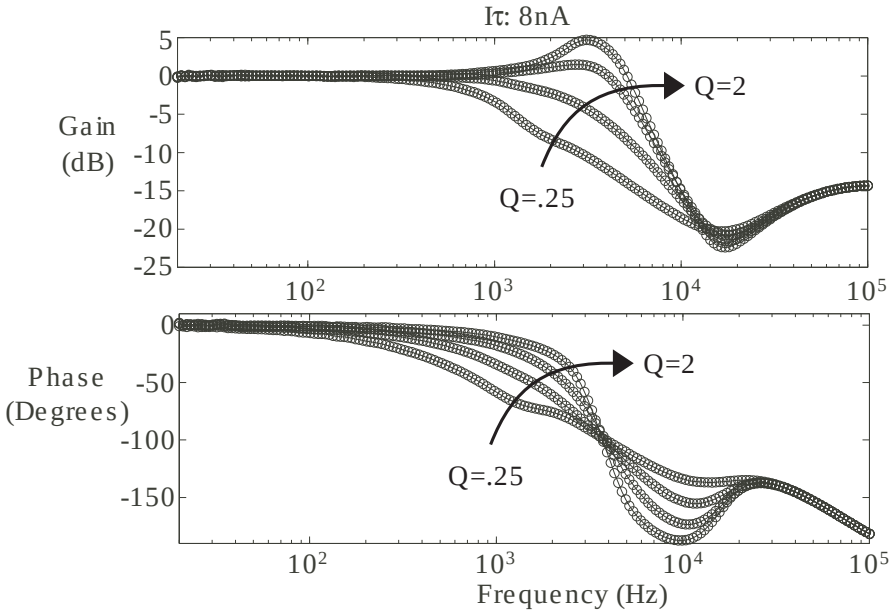


Figure 6.5: Frequency response for a second-order low-pass filter with various quality factors (Q=0.25, 0.5, 1, 2) and an approximate corner frequency of 4kHz.

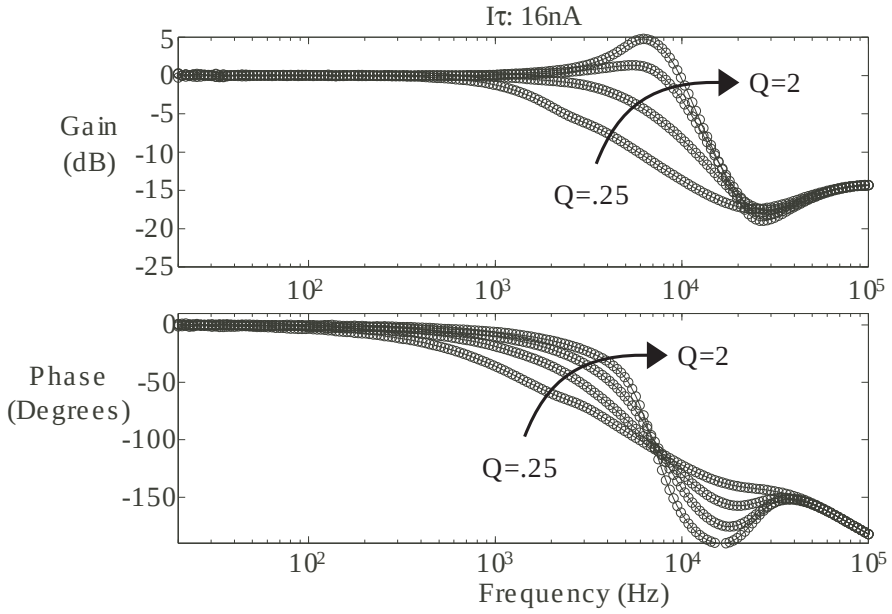


Figure 6.6: Frequency response for a second-order low-pass filter with various quality factors (Q=0.25, 0.5, 1, 2) and an approximate corner frequency of 8kHz.

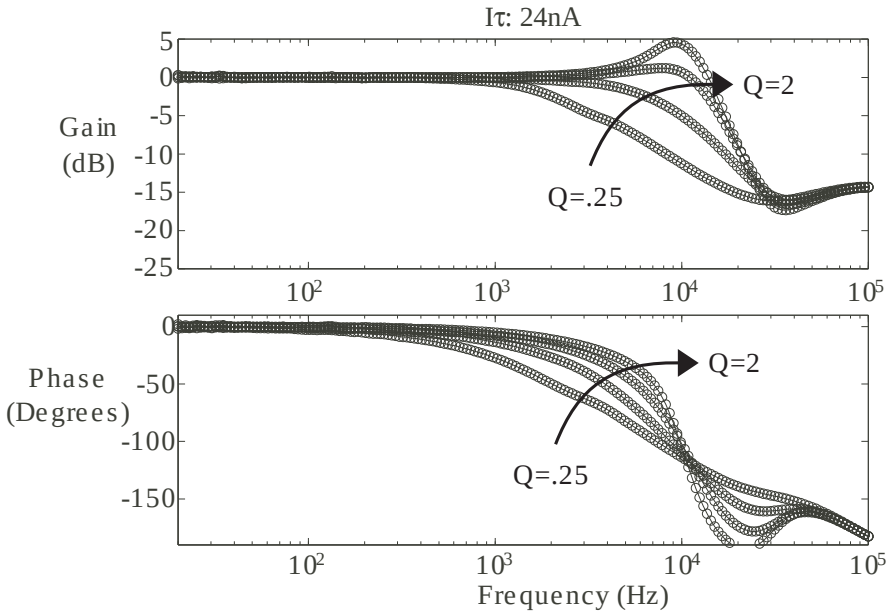


Figure 6.7: Frequency response for a second-order low-pass filter with various quality factors (Q=0.25, 0.5, 1, 2) and an approximate corner frequency of 11kHz.

6.10 show a sample of the two oscillator outputs over varying oscillation frequencies where I_τ was swept from 10nA to 200nA. The only bias current that was changed during these data collections was I_τ. It is possible to tweak other biases to get less distorted output signals for a given I_τ. By tweaking other bias signals, valid output signals can be generated at oscillation frequencies as low as a few hundred Hz (where $I_\tau \approx 0.1$nA). Figure 6.11 shows a plot of the oscillation frequency versus I_τ. By plotting one output versus the other, it is possible to graphically examine the phase difference as shown in Fig. 6.12. Two sinusoids at the same frequency with a 90 degrees phase shift will appear as a perfect circle. Using zero-crossings, the phase difference for this frequency (8.93kHz) was calculated as 87.4 degrees. The phase jitter was measured to be approximately 4% and the total harmonic distortion (THD) ranged from 6% (10kHz) to 10% (90kHz). The THD

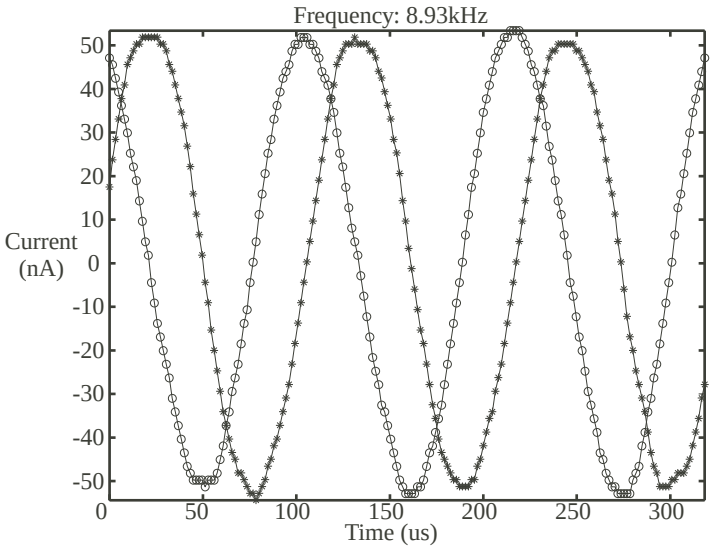


Figure 6.8: Scope capture of the two oscillator outputs at 8.93kHz.

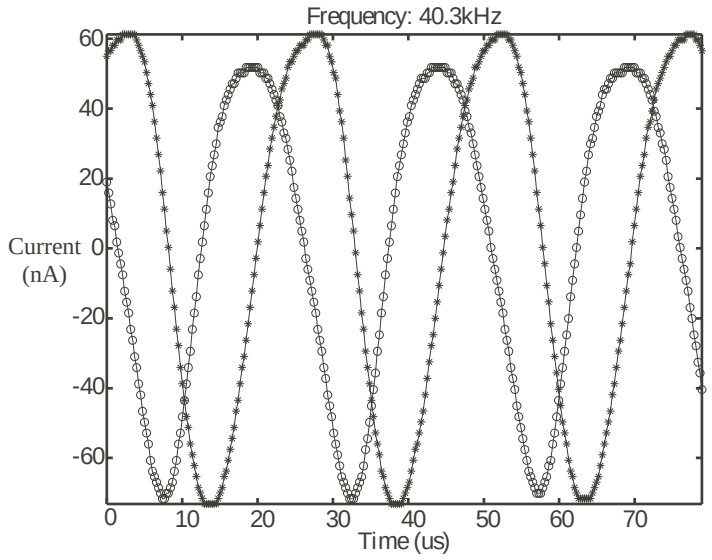


Figure 6.9: Scope capture of the two oscillator outputs at 40.3kHz.

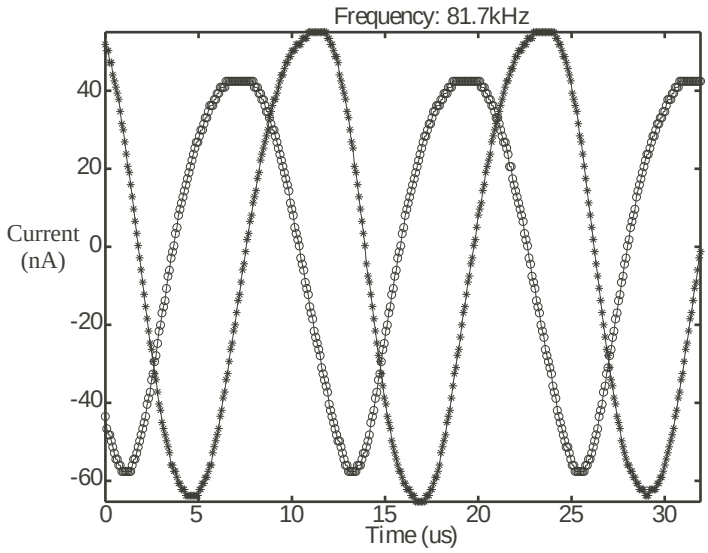


Figure 6.10: Scope capture of the two oscillator outputs at 81.7kHz.

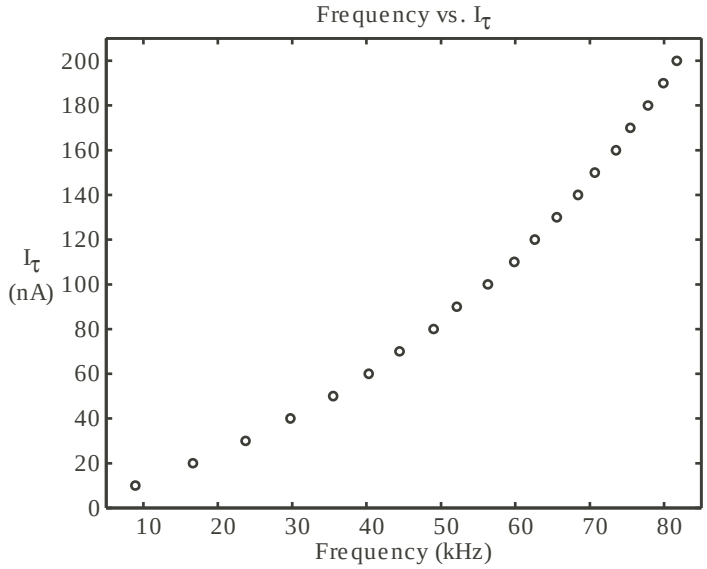


Figure 6.11: Plot of the relationship between the oscillation frequency and I_τ.

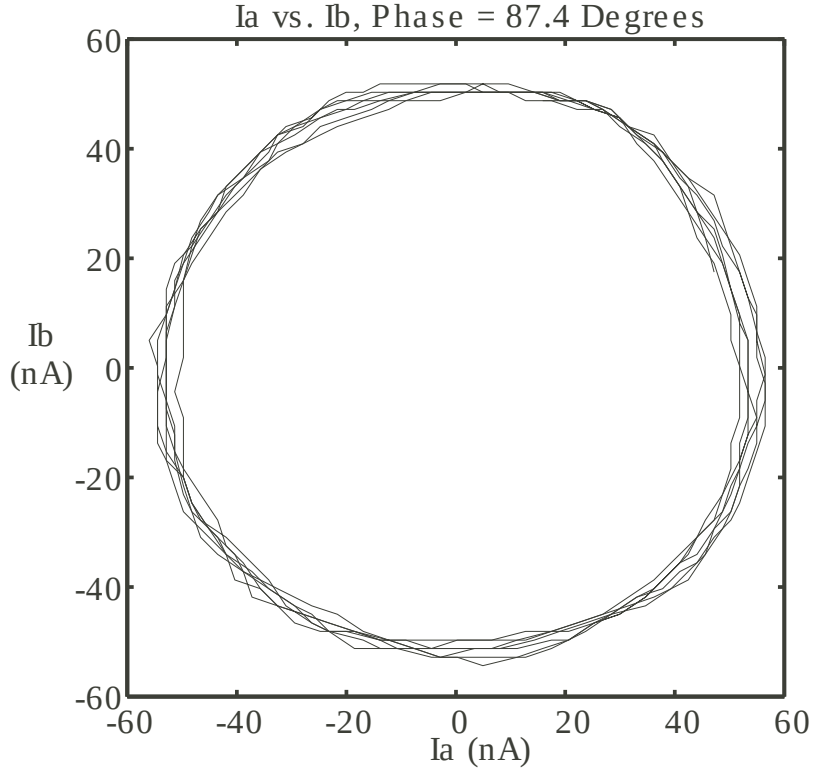


Figure 6.12: Plot of the two oscillator outputs, I_a versus I_b, showing an approximate phase difference of 87.4 degrees. A perfect circle would be the equivalent of a 90 degrees phase difference.

can be expected to be fairly high considering the frequency range and the capacitive nature of the circuit (becoming increasingly worse at higher frequencies).

It is worthwhile to note that this circuit is sensitive to certain biasing conditions with an exceptionally strong dependence on the cascode voltages. This implies that small gain errors around the feedback loop have a significant impact on the output signals' distortion, phase, and frequency.

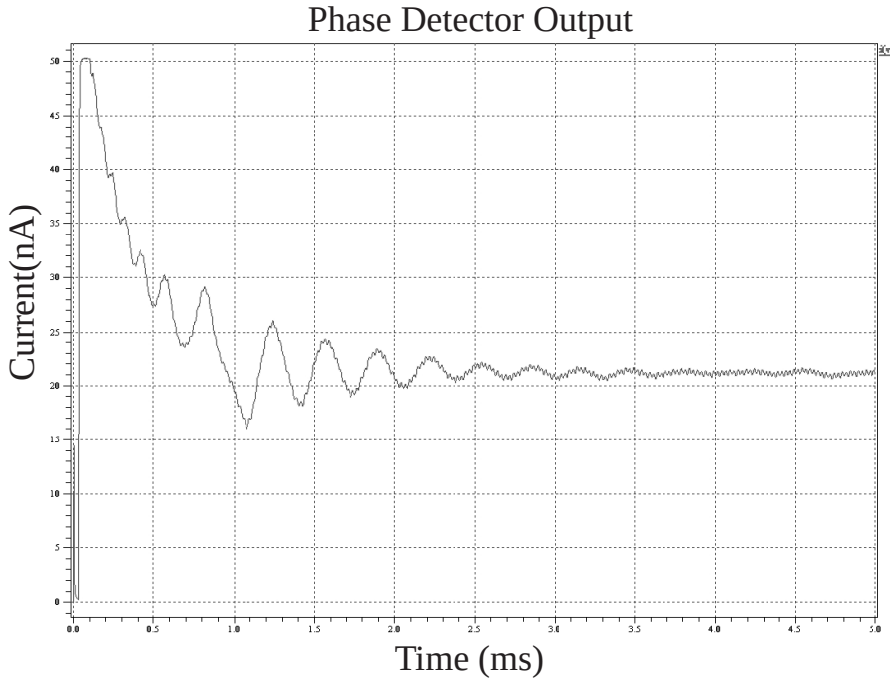
6.6 Phase-Locked Loop Results

Simulations run in TSpice showed that the PLL was able to lock onto frequencies in the range of 20 to 30kHz when the free-running frequency was set to approximately 23kHz. Figure 6.13(a) shows the output of the filtered phase detector signal demonstrating the locking behavior. Traces of the input and output signals after locking are shown in Fig. 6.13(b).

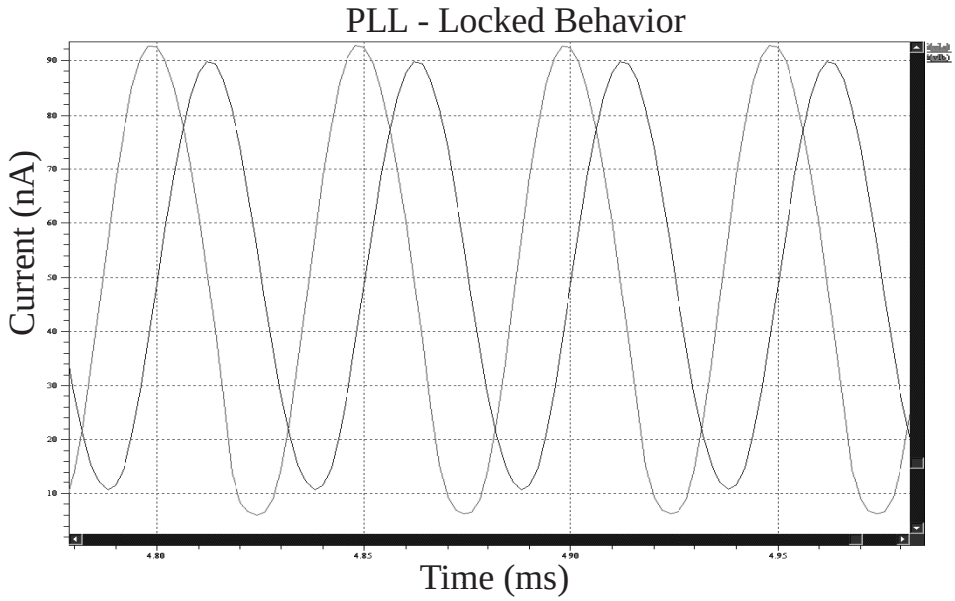
The fabricated phase-locked loop of Chapter 5 was unable to lock onto the input signal's frequency. Figure 6.14 shows the oscillator output and the output of the loop filter (which controls the oscillator frequnecy). It is clear that the loop filter's output was able to modulate the oscillator frequency. However, the signal was too noisy (primarily from 60Hz interference) in order to be able to serve as an effective phase detector.

6.7 Results Summary

The results presented in this dissertation show that the synthesis methodology is both sound and viable for a wide range of applications. The three most limiting factors in preventing better results are mismatch of the the subthreshold current-



(a)



(b)

Figure 6.13: PLL simulation results. (a) Frequency-controlling current showing the locking behavior. (b) Traces of the input and output signals.

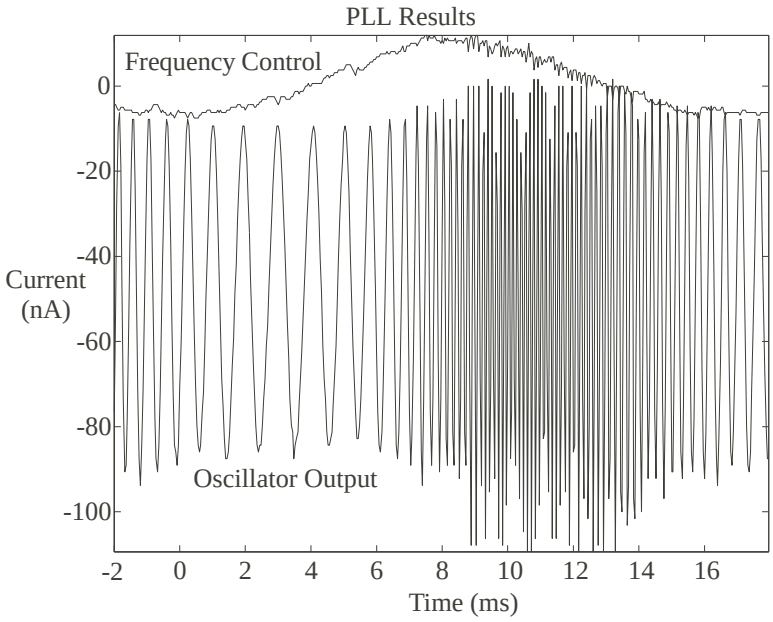


Figure 6.14: Experimental PLL results showing the output of the loop filter (which controls the oscillator frequency and is dominated by 60Hz interference) and the output of the oscillator.

voltage curves, difficulty in getting clean and accurate measurements due to the low current levels, and the potential for higher order effects to create unexpected behaviors due to the capacitive nature of MITE networks.

6.7.1 Mismatch

Being able to inject the floating-gate transistors such that they passed the same current within under one percent variance for the same control gate voltages, did not guarantee the same tolerance for different control gate voltages. For instance, programming with a 0.5% tolerance for a current level of 20nA at a control gate voltage of 1V, might still mean that a change of 50mV on the control gates causes the variance in current to change to several percent. It is not exactly clear what is the fundamental cause for this error. Looking at the relationship for a two-input MITE where the control gates are shorted,

$$I_d = I_s e^{\kappa(2wV + Q/C_{\text{total}})/U_{\text{T}}}\,, \tag{6.1}$$

we can find the slope of the plot of $\ln(I_d)$ versus V by examining

$$\ln(I_d) = \ln(I_s) + \frac{\kappa 2wV}{U_{\text{T}}} + \frac{\kappa Q}{C_{\text{total}} U_{\text{T}}}\,, \tag{6.2}$$

resulting in a slope of $2\kappa w/U_{\text{T}}$. By programming the MITEs, we are able to remove any offsets due to the variance in the trapped floating-gate charge, Q. However, it appears that there still remains a noticeable variance in the slopes which implies that either κ or the weights are not matched well (U_{T} is the thermal voltage, kT/q, and should remain constant across all MITEs). Since both κ and the weights are dependent upon the geometry of the transistors and the capacitors, it is possible that using larger-sized devices might improve these errors. For the work presented here, the variance in these slopes amounted to errors as large as 6% at the edges

of the operating range. This is most likely the cause for the offset and gain errors present in the results.

6.7.2 Measurement Errors and Noise

The second biggest problem was trying to get clean unamplified signals into and out of a chip. With currents on the order of nanoamperes, simply dropping a chip into a breadboard and connecting up devices is not advisable. A gain factor of ten was achieved by duplicating many MITEs to effectively accept or generate ten copies of the input or output currents. Even with this boost, wires connecting the breadboard to the various biasing equipment and oscilloscopes introduced a noticeable amount of noise (especially 60Hz line noise). Additionally, since these circuits are inherently current-mode circuits, to supply and measure any AC signals the AC currents were required to run through off-chip voltage-to-current or current-to-voltage converters which served as another potential noise source.

For optimum performance, these circuits should be run off of a single battery and the bias currents provided by on-chip programmable current sources, such as proposed in [5]. Additionally, custom designed printed circuit boards should help to reduce noise by shortening the wires and moving signal sources closer to the chip. Ideally, on-chip amplifiers would be used to reduce high-amplitude input signals down to the nA level and boost output signals to be easily measured by an oscilloscope.

If the off-chip noise influences can be reduced or eliminated, it will then become more important to investigate the nature of the noise generated within the MITE networks themselves. Because of the inherent non-linear behavior of translinear circuits, analyzing the noise is non-trivial. Additionally, the large signal behavior

of these circuits introduce signal × noise intermodulation as well as resulting in non-stationary noise sources. A complete treatment of noise present in translinear circuits is worthy of an entirely separate dissertation. Work has been done on noise analysis for various types of translinear circuits [13, 27, 31, 34, 35, 48–50].

6.7.3 Feed Through and Higher Order Effects

With a class of circuits that is so inherently dependent upon capacitive coupling to relay signals, it is always possible that under certain conditions these capacitors will no longer act as one-way ports and allow a significant amount of signal to pass both ways. Using the first-order low-pass filter from Section 3.2, if the reference voltage, V_{ref}, were not driven by a source with a low output impedance it is probable that at some higher frequency the input signal would bypass the filter capacitor by following the path shown in Fig. 6.15. What appears to be a double zero in the frequency responses for the first and second-order low-pass filters can probably be attributed to this kind of feed through. The best way to avoid such problems in dynamic MITE networks would be to make certain that the only signal path from one side of a filtering capacitor to the other is either through that capacitor node or through a required feedback connection. For the first-order low-pass filter, this would mean having a separate voltage source for the left-hand side V_{ref} and a separate voltage source for the right-hand side V_{ref}. As would be expected, increasing the transistor sizes to allow for higher current levels while reducing the control gate capacitor size will help to alleviate this phenomenon.

Even for static MITE networks, where no large filtering capacitor is present, the capacitive coupling becomes increasingly more relevant for higher signal frequencies. Analyzing these higher order effects is non-intuitive and is beyond the scope

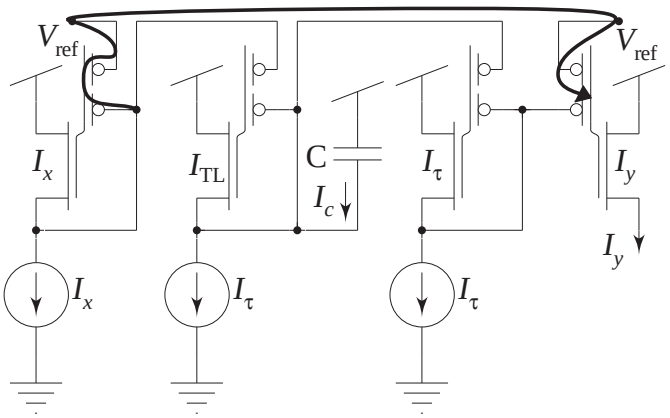


Figure 6.15: Path by which a high frequency input signal could bypass the filter capacitor and show up at the output if the driving source for V_{ref} does not have a low enough output impedance to keep V_{ref} fixed to an effective DC potential.

of this dissertation. While higher order effects in MITE networks is not addressed, some work addressing higher order effects for log-domain filters implemented with BJTs and ways to compensate for these non-idealities can be found in [10, 14, 48].

6.8 Conclusions

This synthesis methodology is a powerful tool for designing a vast range of analog circuits. However, more research must be invested before it can become a reliable tool for the average circuit designer. With its very structural mathematical basis, it lends itself well to computer aided design as demonstrated by the circuit consolidation script included in Appendix C. It is not unreasonable to imagine software that accepts high-level differential equations describing a system and then returns a low-level system decomposition, a fully connected and biased MITE network, and even layout. By removing some of the ambiguity, such as limiting all MITEs to having only two control gates, the intuition required during the circuit construc-

tion can become unnecessary allowing for a fixed set of rules to govern the entire synthesis process.

6.9 Contributions

My contributions to this body of work include the synthesis of more complex systems (vector magnitude with offsets, quadrature oscillator, and phase-locked loop). These examples provide a greater insight into the various options that arise during synthesis that are not present for simpler circuits. While working on the decomposition of several systems, I developed the graphical representation for MITE connections used in this dissertation. This graphical representation helped me to recognize that the circuit construction phase along with the consolidation step could be simplified by restricting MITEs to two control gates and one to one connections. Once restricted, the connection ordering that provided the most opportunities to consolidate MITEs could be determined (somewhat visually) through a process of permutations on the odd and even currents within the graphs that was similar to the way in which equations are factored. Inspired to find the absolute minimum MITE configuration for the oscillator circuit of Section 4.2, I developed the Perl program in Appendix C to perform all the possible permutations for a given set of TL equations and by comparing the order of terms from the outsides progressing inwards, calculate the required number of MITEs to implement a given ordering after consolidation. This graphical representation of connections and the consolidation program should also be valid for standard translinear circuit synthesis.

After an ambitious attempt to synthesize an 8-tap adaptive filter that failed

(presumably due to our inability to balance the trapped charge on the floating-gates), I fabricated a test chip with the programming infrastructure shown in Appendix A.7 to determine if it was possible to accurately program the floating-gate charge for a MITE network. After much experimentation and a steep learning curve, I sought advice from Paul Hasler at Georgia Institute of Technology on methods of fast and accurate programming, which lead to the modeling outlined in Appendix A.5. Having designed the largest MITE networks to date, I have included in Appendix B some generalizations about the nature of these circuits as well as some practicalities when designing large networks.

Appendix A

Programming Floating-Gate Charge

A.1 Programming Overview

One of the underlying assumptions that simplifies the synthesis methodology is that each floating-gate's trapped charge is equal such that for matched floating-gate transistors with equal control gate voltages, the drain currents will be equal. Due to the processes involved during fabrication, the trapped floating-gate charge across an entire chip can vary greatly. One method of equalizing this floating-gate charge was through the exposure of the silicon to UV light. Variations in fabrication technologies made it difficult to be sure that UV light exposure was sufficient in balancing the trapped floating-gate charge. For instance, the opaqueness of the overglass could vary drastically reducing the effectiveness of this method requiring additional design considerations (i.e., making cuts in the overglass layer in the layout). Since it is essential that the floating-gate charge be balanced, the combination of Fowler-Nordheim tunneling and hot-election injection are used to guarantee this

condition. (Tunneling can be used on both PMOS and NMOS floating-gate transistors but due to the lightly doped drain implants employed in most fabrication processes, hot-electron injection is only effective for PMOS transistors.) The goal of this charge-balancing process is to get the drain current of each MITE within a certain variance when all the control gates are fixed at a set voltage. The tunneling process is used to globally remove electrons from the floating-gates and injection is used to add electrons to individual floating-gates in a controlled manner.

The process by which the above techniques are used to balance the charge is as follows. Initially, all floating-gate transistors are globally tunneled to remove electrons from the floating-gate. This process continues until the "least tunneled" transistor's drain current (with a particular voltage at all control gates) is less than some threshold value. Individual transistors are then "programmed" through injection until they hit the target [V_{CG}, I_{d}] condition. Since this balancing of charge of crucial to the correct operation of the circuits, great care must be shown during this process and the following section details one way in which this can be accomplished. It is important to understand the basics of tunneling and injection before we look at the overall programming process.

A.2 Fowler-Norheim Tunneling

The basic principle by which this process is used to remove electrons from the floating-gate involves creating a large voltage across the tunneling capacitor. (A tunneling capacitor is created when a small "finger" of the floating-gate's polysilicon overlaps highly doped N within an N-well.) This potential difference is great enough that electrons are forced through the thin oxide that separates each half

of this capacitor. (With MOSIS' AMI 0.5 μm 5V CMOS process, I found that tunneling may begin with a tunneling voltage, V_{tun}, around 10-12V with a 5V Vdd and should be sufficiently fast for a quick global "erase" with V_{tun} at 15V. Note that too great a potential across the tunneling capacitor can damage the oxide insulator and therefore caution should be used. Also, additional care must be used when creating the layout since the average CMOS process is not rated for such high voltages.

A.3 Hot-Electron Injection

Hot-electron injection occurs when electrons traveling through the channel achieve a high enough velocity that they can travel through the gate oxide over the channel. The conditions for this to occur involve the combination of a significant source-drain voltage (approximately 4V for the MOSIS AMI 0.5 μm process) and the correct "tilt" from the channel to gate (i.e., gate voltage must be more positive than the channel in order to steer the electrons into the oxide surface). Specifically, holes travel down the length of the channel from source to drain and collisions at the drain cause electron-hole pairs. These electrons then travel back up the channel with increasing velocity and are steered towards the oxide. A very small percentage of these electrons reach a high enough velocity and are able to travel through the oxide and make their way to the floating-gate.

A.4 Programming Method

A simple self-convergent method for a controlled injection process is described in [2] but was found to result in an unacceptably inaccurate programming for

any reasonable length process. (The self-regulation of this method does allow for increased accuracy at a cost of a longer programming time.) The method presented here involves modeling the injection process over a range of operating conditions that will allow for a faster and more accurate programming method. This method involves "pulsing" the drain voltage for various pulse lengths.

A.5 Derivation

Injection current is described as

$$I_{\text{inj}} = I_{\text{inj0}} \left(\frac{I_d}{I_{do}} \right)^{\alpha} e^{\left(\delta V_{\text{sd}}/V_{\text{inj}}\right)} \tag{A.1}$$

where I_{inj0} is a scaling current, I_d is the drain current, I_{do} is the bias drain current, δV_{sd} is the change in V_{sd} from the bias value of V_{sd0}, and α and V_{inj} are extracted values. Note that

$$\left(\frac{I_d}{I_{do}} \right)^{\alpha} = \frac{\%\ change\ @\ I_d}{\%\ change\ @\ I_{do}} \left(\frac{t_{\text{norm}}}{t_{\text{pulse}}} \right) \tag{A.2}$$

and that

$$e^{\delta V_{\text{sd}}/V_{\text{inj}}} = \frac{\%\ change\ @\ \left(V_{\text{sd0}} + \delta V_{\text{sd}}\right)}{\%\ change\ @\ V_{\text{sd0}}} \left(\frac{t_{\text{norm}}}{t_{\text{pulse}}} \right) . \tag{A.3}$$

Thinking of the charge flowing off of the floating-gate (or negative charge, electrons, flowing onto the floating-gate):

$$C_T \frac{dV_{fg}}{dt} = -I_{\text{inj}} . \tag{A.4}$$

The PMOS subthreshold current can be viewed as the present current, I_{do}, scaled as a function of ΔV_{fg},

$$I_d = I_{do} e^{-\left(\kappa \Delta V_{fg}\right)/U_{\text{T}}} . \tag{A.5}$$

Taking the derivative of the above results in

$$\frac{dI_d}{dt} = I_d \left(-\frac{\kappa}{U_{\text{T}}} \right) \frac{dV_{fg}}{dt} . \tag{A.6}$$

Substituting in dV_{fg}/dt into the above, we find

$$\frac{dI_d}{dt} = I_d \left(-\frac{\kappa}{U_{\mathrm{T}}}\right)\left(-\frac{I_{\mathrm{inj}}}{C_T}\right) . \tag{A.7}$$

Inserting the representation for I_{inj} (under the assumption that I_{inj} remains constant throughout the pulse and has no dependency on time) results in

$$\frac{dI_d}{dt} = I_d \frac{\kappa}{C_T U_{\mathrm{T}}} \left[I_{\mathrm{inj0}} \left(\frac{I_d}{I_{do}}\right)^{\alpha} e^{\delta V_{\mathrm{sd}}/V_{\mathrm{inj}}} \right] . \tag{A.8}$$

Rearranging and dividing both sides by I_{do}, we find

$$\frac{d}{dt}\left(\frac{I_d}{I_{do}}\right) = \underbrace{\frac{\kappa I_{\mathrm{inj0}}}{C_T U_{\mathrm{T}}}}_{1/t_{\mathrm{pulseo}}} \left[\left(\frac{I_d}{I_{do}}\right)^{1+\alpha} e^{\delta V_{\mathrm{sd}}/V_{\mathrm{inj}}} \right] . \tag{A.9}$$

Note that an I_d/I_{do} was grouped resulting in the $1+\alpha$ exponent. Defining t_{pulseo} as $C_T U_{\mathrm{T}}/\kappa I_{\mathrm{inj0}}$ and x as I_d/I_{do} simplifies the above to

$$t_{\mathrm{pulseo}} \frac{dx}{dt} = x^{1+\alpha} e^{\delta V_{\mathrm{sd}}/V_{\mathrm{inj}}} . \tag{A.10}$$

δV_{sd} is fixed during the pulse so defining a new t_{pulseo},

$$t'_{\mathrm{pulseo}} = t_{\mathrm{pulseo}} e^{-\delta V_{\mathrm{sd}}/V_{\mathrm{inj}}} , \tag{A.11}$$

results in

$$t'_{\mathrm{pulseo}} \frac{dx}{dt} = x^{1+\alpha} . \tag{A.12}$$

Integrating across the pulse length we find

$$\int_{x(t=0)}^{x(t=t_{\mathrm{pulse}})} \frac{dx}{x^{1+\alpha}} = \int_0^{t_{\mathrm{pulse}}} \frac{dt}{t'_{\mathrm{pulseo}}} , \tag{A.13}$$

which becomes

$$-\frac{1}{\alpha}\frac{1}{x^{\alpha}}\Big|_{x(t=0)}^{x(t=t_{\mathrm{pulse}})} = \frac{t_{\mathrm{pulse}}}{t'_{\mathrm{pulseo}}} . \tag{A.14}$$

Evaluating the above results in

$$\frac{1}{x(t=0)^{\alpha}} - \frac{1}{x(t=t_{\text{pulse}})^{\alpha}} = \alpha \frac{t_{\text{pulse}}}{t'_{\text{pulseo}}} \,. \tag{A.15}$$

Solving for t'_{pulseo}, we find

$$t'_{\text{pulseo}} = \frac{\alpha t_{\text{pulse}}}{\frac{1}{x(t=0)^{\alpha}} - \frac{1}{x(t=t_{\text{pulse}})^{\alpha}}} \,, \tag{A.16}$$

which becomes

$$t_{\text{pulseo}} e^{-\delta V_{\text{sd}}/V_{\text{inj}}} = \frac{\alpha t_{\text{pulse}}}{\frac{1}{x(t=0)^{\alpha}} - \frac{1}{x(t=T)^{\alpha}}} \tag{A.17}$$

when the expression for t'_{pulseo} is inserted. Solving for δV_{sd}, we find

$$\delta V_{\text{sd}} = V_{\text{inj}}\ ln\left(\frac{\alpha t_{\text{pulse}}/t_{\text{pulseo}}}{\frac{1}{x(t=0)^{\alpha}} - \frac{1}{x(t=t_{\text{pulse}})^{\alpha}}}\right) \,, \tag{A.18}$$

which simplifies to

$$\delta V_{\text{sd}} = V_{\text{inj}}\ ln\left(\alpha \frac{t_{\text{pulse}}}{t_{\text{pulseo}}}\right) - V_{\text{inj}}\ ln\left(\frac{1}{x(t=0)^{\alpha}} - \frac{1}{x(t=T)^{\alpha}}\right) \,. \tag{A.19}$$

$$x(t=0)^{\alpha} = \left(\frac{I_{do}}{I_{do}}\right)^{\alpha} = 1 \tag{A.20}$$

$$x(t=T)^{\alpha} = \left(\frac{I_d}{I_{do}}\right)^{\alpha} = \left(\frac{I_{do} + \Delta I_d}{I_{do}}\right)^{\alpha} = (1+PC)^{\alpha} \,, \tag{A.21}$$

where PC represents the percent change of I_d. Using these expressions for x, we substitute into Eq. A.19 and set $\delta V_{\text{sd}} = 0$ to solve for the prefactors.

$$0 = V_{\text{inj}}\ ln\left(\alpha \frac{t_{\text{pulse}}}{t_{\text{pulseo}}}\right) - V_{\text{inj}}\ ln\left(1 - \frac{1}{(1+PC)^{\alpha}}\right) \tag{A.22}$$

Simplifying, we find

$$\alpha \frac{t_{\text{pulse}}}{t_{\text{pulseo}}} = 1 - \frac{1}{(1+PC)^{\alpha}} \,. \tag{A.23}$$

Once modeling data is collected and α is extracted, a point can be chosen (giving t_{pulse} and PC) to determine t_{pulseo}. Equation A.19 can then be solved for the expected current after a certain programming pulse, $[t_{\text{pulse}}, \delta V_{\text{sd}}]$, when the starting current is I_{do}.

$$I_d^{\alpha} = I_{do}^{\alpha}\left(1 - \alpha \frac{t_{\text{pulse}}}{t_{\text{pulseo}}} e^{-\delta V_{\text{sd}}/V_{\text{inj}}}\right) \tag{A.24}$$

A.6 Data Collection

The modeling of the injection process requires data spanning across a variety of programming conditions (i.e., $V_{sd} = 4\text{V} \leftrightarrow 6\text{V}$, $t_{\text{pulse}} = 0.1\text{ms} \leftrightarrow 5\text{s}$, $I_{\text{start}} = 1\text{nA} \leftrightarrow 100\text{nA}$). Initially, all transistors should be tunneled so that they pass very small currents for the chosen programming V_{cg} (about an order of magnitude smaller than any current that would be chosen as the target current). Each transistor is injected according to a $[V_{sd}, t_{\text{pulse}}]$ pair. These pairs are determined such that the first transistor starts at the smallest V_{sd} and the largest t_{pulse}. The V_{sd} values should be scaled linearly such that the last transistor will be using the largest V_{sd}. The t_{pulse} values should be scaled logarithmically such that the last transistor will be programmed with the smallest t_{pulse} value. Measuring the drain current before every pulse, each transistor should be pulsed according to the prearranged pulsing conditions. (It might be useful to also pulse V_{cg} up by a fixed amount to offset the effect of the drain coupling into the gate through the overlap capacitance to keep the transistor in weak inversion. If the transistor is forced out of weak inversion, the modeling data will not be useful.) When the current exceeds some maximum level (ie. 100nA), this transistor is finished and pulsing should be continued with the appropriate pulsing conditions until all transistors have been completed. For an array of ten transistors with $V_{dd} = 5\text{V}$, $V_{sd} = 4.5 \leftrightarrow 5\text{V}$, and $t_{\text{pulse}} = 0.1 \leftrightarrow 10\text{s}$:

Transistor	V_{sd}	t_{pulse}
1:	4.5V	10s
2:	4.55V	5.99s
⋮	⋮	⋮
10:	5V	0.1s

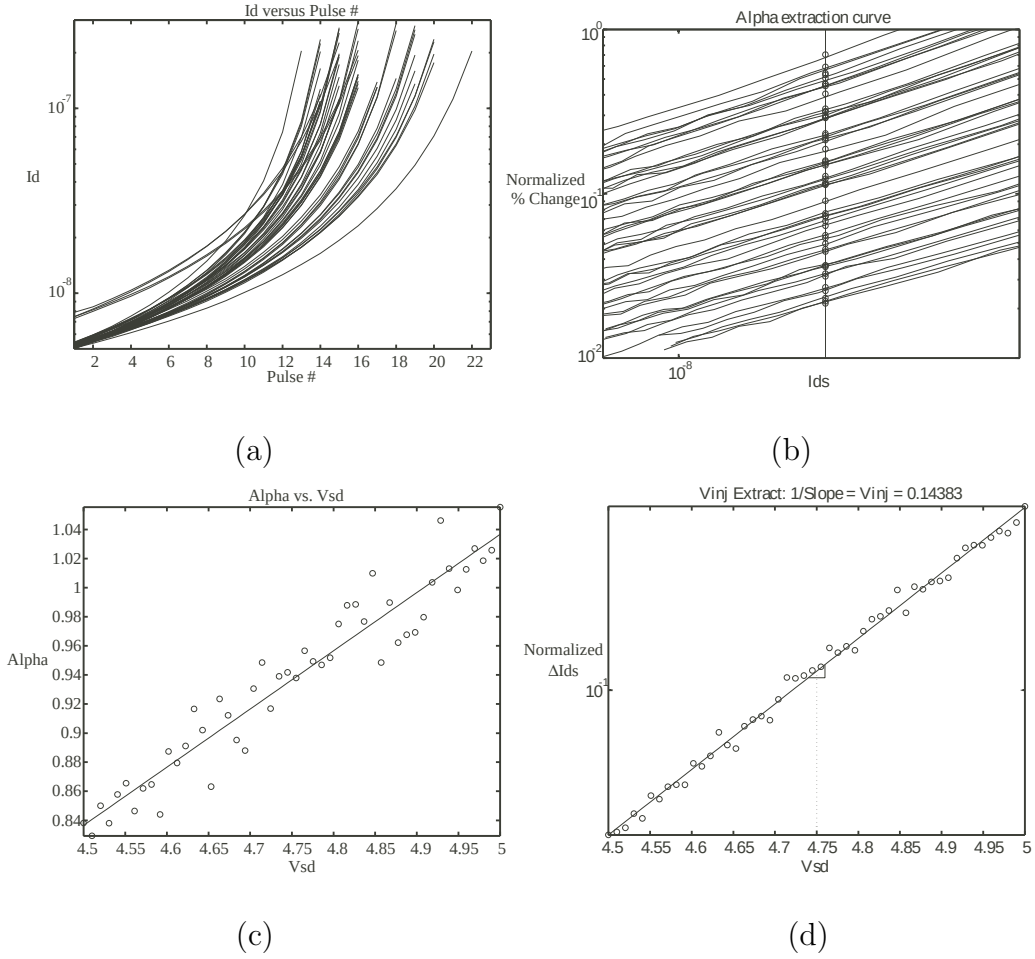


Figure A.1: Plots of injection data used to extra modeling parameters.

Plotting $log(I_d)$ vs. pulse iteration should display an exponential relationship, as shown in Fig. A.1(a). (Displayed as plotting I_d on a semilogy plot-type.) This signifies a "double exponential" relationship. Plotting $log\left(\Delta I_d \frac{1sec}{t_{\text{pulse}}}\right)$ (normalized δI_d) versus $log(I_d)$ should result in straight parallel lines with slopes of approximately 0.7-1.2, as shown in Fig. A.1(b). The slope of these lines is extracted as α. Choose a fitting current, I_{do}, in the middle of the programming range (ie. $I_{do} = 10nA$). The slope values at this current should be interpolated and plotted versus the corresponding V_{sd}, as shown in Fig. A.1(c). (Note that the initial

normalization vertically shifts the log-log plot lines and only comes into play on the computation of V_{inj}below.) The slope (i.e., α) may increase significantly with higher V_{sd} values. You can account for this by extracting the slope in Fig. A.1(c) and replacing α with $\alpha_0 + \alpha_{slope}V_{sd}$. Otherwise, an average of these α values can be used in the below model.

Using the same I_{do}, extract the "Normalized % Change" (y-value in Fig. A.1(b)) and plot these extracted values as $log(\Delta I_d @ I_d = I_{do})$ versus V_{sd}. This should result in a straight line from V_{sdmin} to V_{sdmax}. Choose a fitting V_{sd} level, V_{sdo}, in the middle of the pulse range (ie. $V_{sdo} = 4.75V$). The extracted slope at this point is equal to $1/V_{inj}$. Figure A.1(d) shows the final plot used for the V_{inj} parameter extraction.

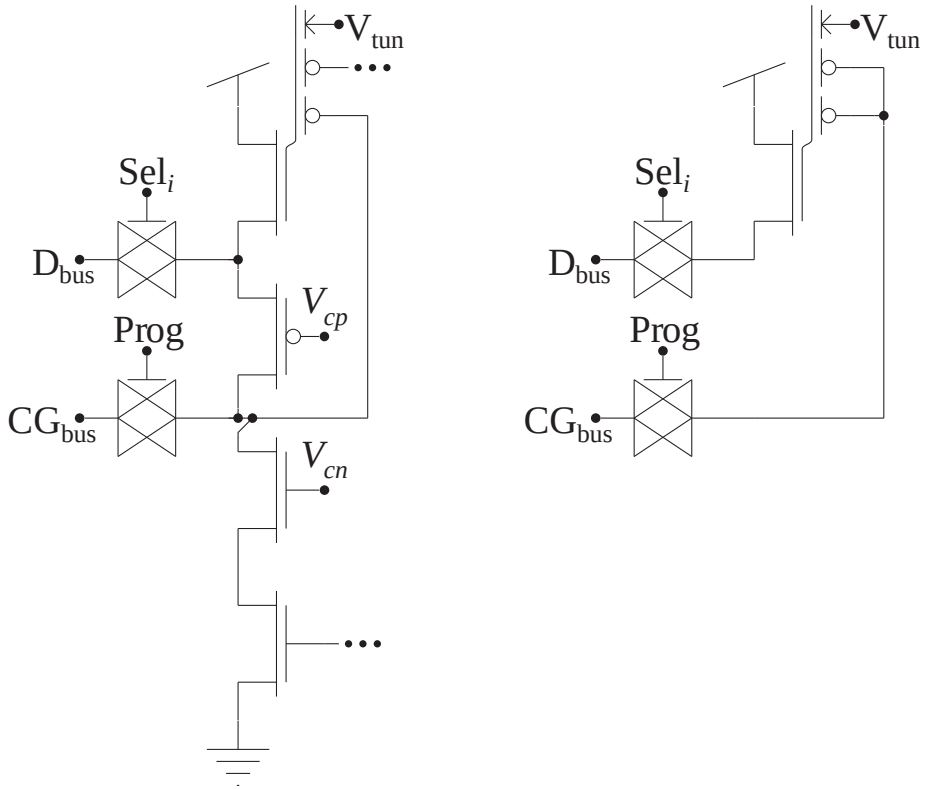
To summarize, α takes into account the injection current's dependency on the starting current level and V_{inj} accounts for the dependency on V_{sd}.

A.7 Programming Infrastructure

In Sections. 1.4 and 2.2, I mentioned that all MITEs are implemented with cascoded floating-gate PMOS (FGPMOS) transistors and all current mirrors are implemented with cascoded NMOS transistors. While these cascode transistors are needed to improve circuit performance, they also serve the additional purpose of assisting in the programming scheme. Figure A.2(a) shows the actual implementation of a typical "slice" of a MITE circuit. By turning the cascode transistors off (i.e., $V_{cp} = V_{dd}$ and $V_{cn} = \text{Gnd}$), we effectively isolate the drain of the floating-gate PMOS and remove any connections to the control gates, as shown in Fig. A.2(b). Setting the global programming signal, Prog, high, all control gates are shorted

together through the transmission gates to the control gate bus, CG_{bus}. Using a simple shift register, a single FGPMOS can be selected by setting Sel_i high, which shorts the FGPMOS drain to the drain bus, Dbus. With global control of all the control gate voltages and individual control of FGPMOS drains, each FGPMOS can be programmed separately.

The "third" control gates shown in Fig. A.2 represent the tunneling capacitor needed for the global "erase". This tunneling capacitor is formed by running a thin poly "finger" from the floating-gate across a highly doped NWell. All MITEs share the same tunneling line, V_{tun}, and therefore, tunneling can only be used to tunnel all the MITEs. Individual tunneling lines are impractical because the necessary tunneling voltages ($\sim$ 10-15 V) exceed the voltage limitations for most CMOS processes, preventing standard digital circuitry (transmission gates, multiplexers, etc.) from passing these high voltages. When creating layout for nodes that will exceed the recommended voltage for a process, additional care must be used to prevent unexpected behavior since the standard design rules are no longer valid. i.e. The minimum well-to-well spacing for wells at different potentials should probably be doubled to prevent any unexpected well-to-well current flowing through the substrate. As an aside, the effects of the tunneling fingers on the total capacitance should be investigated to ensure that they are not the cause of mismatch across all MITEs.



(a) (b)

Figure A.2: Programming infrastructure that allows for a global "erase" through the shared tunneling line, V_{tun}, and individual hot-electron injection through the use of the MITE select signal, Sel_i, control gate bus, CG_{bus}, and drain bus, D_{bus}. (a) MITE cell including two cascode transistors and two transmission gates. (b) Effective MITE cell during the programming phase when both cascode voltages are turned off.

Appendix B

Circuit Practicalities

B.1 Power Supply

Because all transistors are cascoded, most "slices" of a MITE network will involve a stack of four transistors between the power supply and ground (FGPMOS, p-cascode, n-cascode, NMOS mirror). In order to remain saturated, each transistor must have a minimum of approximately $4U_{\mathrm{T}}$ ($\sim$100mV) across its drain-source nodes, requiring a total power supply of at least 400mV. Additionally, there must be enough headroom to bias the cascode transistors appropriately (i.e., $V_{cp} \sim V_{dd} - 1V$ and $V_{cn} \sim 0.8$V). Because the voltages are logarithmically compressed signals, a few hundred millivolts of swing can produce several decades of current levels. (The number of control gates and their size affect the coupling each control gate requiring more voltage swing for circuits with smaller control gates.) Therefore, if the FGPMOS are programmed such that the average control gate voltage is at $V_{dd}/2$, a 1V power supply would allow for a control gate voltage swing of 600mV ($1V - 4U_{\mathrm{T}} = 600$mV). With careful programming, operation, and biasing, it would be reasonable to expect these circuits to work down to a power supply as low as

600-700mV. (Below a V_{dd} of 1V, it would probably be necessary to consider making the cascode transistors wider to reduce the required V_{gs} voltages.)

B.2 Current Levels

Maintaining transistor operation within the subthreshold region is necessary to take advantage of the exponential I-V relationship. Reasonably sized transistors (W/L $\sim$ 60/4) in the MOSIS AMI 0.5 μm process start to show deviation from the exponential curve around 150nA. Using stacked transistor layout, it should be possible to create moderately sized transistors that allow for subthreshold current levels approaching 1uA or higher. Alternately, lateral BJT's from a Bi-CMOS process could be used, allowing for much higher current levels.

B.3 Frequency Limits and Higher Order Effects

Methods for increasing the current levels (i.e., a bipolar MITE implementation), will allow for higher frequency operation before higher order effects cause unacceptable performance. If technology improvements allow for greater control gate coupling, decreasing the control gate size (while maintaining sufficient coupling) will help to increase the frequency range before higher order effects begin to limit operation.

B.4 General Design and Layout Techniques

Because these circuits inherently require many current biases and often times, several of each, it is recommended that when a current sink is required, it be gen-

erated by sourcing a current into an NMOS mirror. Alternately, if a current source is required, sinking a current from a PMOS mirror is recommended. Another important reason for using mirrors to provide currents is that taking an internal node out to a bonding pad that is not at the beginning of a MITE array adds a significant amount of capacitance to that particular node and may cause unexpected behavior. Mirroring the necessary currents maintains a similar environment seen by all MITEs. Dummy devices at all "ends" should also be used to help maintain consistent coupling between neighboring devices.

Appendix C

Perl Code for Automated Consolidation

```
#!c:\perl\bin\perl
use strict;
use warnings;

our $min_mites;
our $best_sol_cnt;
our @shared_best;
our @best;
our @LR;

my $time;

$time = localtime time;
print $time;

my ($i,$j,$k); #temp variables

#alias used for powers greater than 1 and should
# be "undone" after optimizing
my @alias=(['u2','v2'],
           ['u','v']);
#avail used for previous knowledge of terms that
#are available for sharing (presumably from another
#circuit that is already constructed
```

```
my $avail='';

my @L=(); #left side TLP terms (GLOBAL)
my @R=(); #right side TLP terms (GLOBAL)

@L=(['r1','r'],
    ['r2','r'],
    ['r3','r'],
    ['r4','r'],
    ['r5','r']
    );

@R=(['u2','u'],
    ['v2','v'],
    ['2z','z'],
    ['2z','u'],
    ['2z','v']
    );

#NEED TO IMPLEMENT FLAG TO MARK CERTAIN
#MITES AS "UNSHARABLE" (ie. OUTPUT MITES)
# - PRECEED TERMS WITH AN UNDERSCORE?
#Need to implement as method for maintaining
#output structure form

my $rows=$#L+1; #rows in L and R
my $cols=$#{$L[0]}+1; #columns in L and R

$min_mites=$rows*$cols*2;
#best solution mite count update through 'mutate#()'
$best_sol_cnt=0; #counts matches to best solution

#stick @L and @R together for mutations by row
@LR=@L;
for($i=0;$i<$rows;$i++){
   @{$LR[$i+$rows]}[0..$cols-1]=@{$R[$i]}[0..$cols-1];
}

print "\nTLP's:\n";
for($i=0;$i<$rows;$i++){
    print @{$L[$i]}," = ",@{$R[$i]},"\n";
}

&mutate1();
```

```
print "\nShared Layout: $min_mites:$best_sol_cnt\n";
for($i=0;$i<=$#best;$i++){
   for($j=0;$j<=$#{$best[0]};$j++){
      if($shared_best[$i][$j]==0){
         print $best[$i][$j]," ";
         if(length($best[$i][$j])==1){
         #add space is not 2-char term
            print " ";
         }
      }else{
         print "   ";
      }
   }
   print "\n";
}
$time = localtime time;
print $time;


#Expects array-of-arrays of strings that contain TL terms
#Additional arrays requiring mutation are
#read from global array-of-arrays (@LR)
#If next loaded array is undef assume
#at lowest level and call consol()
#Current best-so-far mite count and
#configuration is compared/updated globally
#If not at lowest level, call mutate2()
#to mutate current array

#mutate1(@S[2-d])
sub mutate1{
   my $count;
   my @tmp;
   my $i;
   my $j;
   if($#_==$#LR){ #no more rows to mutate so call consol
      ($count,@tmp)=&consolidate(@_);
      if($count<$min_mites){
         #separate 2 matrices
         for($i=0;$i<($#tmp+1)/2;$i++){
@{$best[$i]}[0..$#{$tmp[0]}]=@{$tmp[$i]}[0..$#{$tmp[0]}];
@{$shared_best[$i]}[0..$#{$tmp[0]}]
    =@{$tmp[$i+($#tmp+1)/2]}[0..$#{$tmp[0]}];
```

```
            }
            $min_mites=$count;

            print "\nNew Best Layout: $min_mites\n";
            for($i=0;$i<=$#best;$i++){
               for($j=0;$j<=$#{$best[0]};$j++){
                  if($shared_best[$i][$j]==0){
                     print $best[$i][$j]," ";
                     if(length($best[$i][$j])==1){
                     #add space is not 2-char term
                        print " ";
                     }
                  }else{
                     print "   ";
                  }
               }
               print "\n";
            }
            $best_sol_cnt=1; #reset "multiple best" counter
         }elsif($count==$min_mites){
            $best_sol_cnt++; #add to multiple best
         }
      }else{
         push(@_,$LR[$#_+1]);
         #push next row from global @LR onto current set
         &mutate2(0,@_);
         #call individual row mutation sub with level '0' status
      }
   }

   #mutate2($lev,@strings[2-d])
   #Expects $lev to know where to start mutations
   #and expects array of strings that contain TLP terms
   #Current best-so-far mite count is compared
   # and updated globally ($min_mites)
   #If new 'best', update grid as well
   #Each new mutation calls mutate2()
   #recursively with that mutation

   sub mutate2{
      my $lev=shift @_;
      #how many terms are already 'set'
      #(lev=2 means [0] and [1] are 'set')
      my $rows=$#_; #rows in @_
```

```
   my $cols=$#{$_[0]}+1; #cols in @_

   if($lev==$cols){
   #call mutate1 for next row/consolidation
   #(if no more rows)
      &mutate1(@_);
   }else{
      &mutate2($lev+1,@_);
      #call for more mutations at next level (column)
      my $i=0;
      for($i=$lev+1;$i<$cols;$i++){
      #make mutations after init call to mutate2
         my $temp=$_[$rows][$lev];
         $_[$rows][$lev]=$_[$rows][$i];
         #swap terms at current level
         $_[$rows][$i]=$temp;
         &mutate2($lev+1,@_);
         #call for more mutations at next level (column)
      }
   }
}

#expecting (@grid[2-d])
#@grid - array of arrays of strings with first
#half/2nd half of rows being L/R sides of TLP equations
sub consolidate{
   my $rows=$#_+1; #get row count
   my $cols=$#{$_[0]}+1; #get $cols
   my @g; #Grid of terms alternating odd/even per line
   my @shared=(); #flag if shared mite
   my ($i,$j); #temp vars

   for($i=0;$i<$rows/2;$i++){
      for($j=0;$j<$cols;$j++){
         #creates odd/even mix of terms
         #(1st half of rows odd, 2nd even)
         $g[$i][$j*2]=$_[$i][$j];
         $g[$i][$j*2+1]=$_[$i+$rows/2][$j];
         $shared[$i][$j*2]=0;
         $shared[$i][$j*2+1]=0;
      }
   }
   #recalculate the new dimensions
   $cols=$cols*2;
```

```
$rows=$rows/2;

# Find the number of unique terms and
# create a space-separated string of them
my $terms=" ";  #string of terms (space-delimited)
my $nterms=0; #count of terms
my $share.=$avail;
#string to hold list of sharable chars/strings
#(add known values here)
my $nmites=$rows*$cols;
#holds the count of Mites needed
#(will decrement upon sharing)

# Left to Right terms (skipping last term)
for($i=0;$i<$rows;$i++){
   for($j=0;$j<$cols-1;$j++){
   #note that the entire line should not be added
      if($shared[$i][$j]!=1){ # if not shared so far
         my $temp=' '; #holds temporary string to match
         for($k=0;$k<=$j;$k++){
            $temp.=$g[$i][$k];
            #gathers terms from L->R to curr column
         }
         $temp.=' '; #will start/end match with spaces
         if(index($share, $temp)!=-1){
         #if term is already in shared list
            $shared[$i][$j]=1; #set that it is shared
            $nmites--; #remove from Mite count
            #print "Sharing: ", $g[$i][$j], "\n";
         }else{
            #print "Added: ",$temp,"\n";
            $share.=$temp; #add term to shared list
         }
      }
   }
} # Right to Left terms (skipping last term)
for($i=0;$i<$rows;$i++){
   for($j=$cols-1;$j>0;$j--){
   #note that the entire line should not be added
      if($shared[$i][$j]!=1){ # if not shared so far
         my $temp=' '; #holds temporary string to match
         for($k=$cols-1;$k>=$j;$k--){
            $temp.=$g[$i][$k];
            #gathers terms from R->L to curr column
```

```
            }
            $temp.=' '; #will start/end match with spaces
            if(index($share, $temp)!=-1){
            #if term is already in shared list
               $shared[$i][$j]=1; #set that it is shared
               $nmites--; #remove from Mite count
               #print "Sharing: ", $g[$i][$j], "\n";
            }else{
               #print "Added: ",$temp,"\n";
               $share.=$temp; #add term to shared list
            }
         }
      }
   }
   return ($nmites,@g,@shared);
}
```

Bibliography

[1] L. Carley, "Synthesis tools for mixed-signal ics: progress on frontend and backend strategies," in *Proc. of 33rd Design Automation Conference*, vol. 1, Las Vega, NV, June 1996, pp. 298–303.

[2] C. Diorio, "A p-channel mos synapse transistor with self-convergent memory writes," *IEEE Transactions on Electron Devices*, vol. 47, no. 2, pp. 464–472, 2000.

[3] E. Drakakis, A. Payne, and C. Toumazou, "Bernoulli operator: A low-level approach to log-domain processing," *Electronics Letters*, vol. 33, no. 12, pp. 1008–1009, 1997.

[4] ——, "Log-domain state-space: A systematic transistor-level approach for log-domain filtering," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 3, pp. 290–305, 1999.

[5] C. Duffy, E. Farquhar, and P. Hasler, "Practical issues using e-pot circuits," in *Proc. of the 2002 IEEE International Symposium on Circuits and Systems*, vol. 5, Phoenix, AZ, May 2002, pp. 26–29.

[6] C. Enz and M. Punzenberger, "1-V log-domain filters," in *Analog Circuit Design: Volt Electronics; Mixed-Mode Systems; Low-Noise and RF Power Amplifiers for Telecommunication*, J. Huijsing, R. van de Plassche, and W. Sansen, Eds. Boston: Kluwer, 1999, pp. 33–67.

[7] R. Fox and M. Nagarajan, "Multiple operating points in a cmos log-domain filter," *IEEE Transactions on Circuits and Systems II*, vol. 46, no. 6, pp. 705–710, 1999.

[8] D. Frey, "Log-domain filtering: An approach to current-mode filtering," *IEE Proc. G*, vol. 140, no. 6, pp. 406–416, 1993.

[9] ——, "Explicit log-domain root-mean-square detector," December 1996, u.S. Patent No. 5,585,757.

[10] ——, "Synthesis of distortion compensated log-domain filters using state space techniques," in *Proc. of the 1998 IEEE International Symposium on Circuits and Systems*, vol. 1, Monterey, CA, May 1998, pp. 321–324.

[11] B. Gilbert, "Translinear circuits: An historical review," *Analog Integrated Circuits and Signal Processing*, vol. 9, no. 2, pp. 95–118, 1996.

[12] ——, "Translinear circuits: A proposed classification," *Electronics Letters*, vol. 11, no. 1, pp. 14–16, 1975, see also errata, vol. 11, no. 6, p. 136, 1975.

[13] M. Kouwenhoven, J. Mulder, and A. van Roermund, "Signalnoise intermodulation in translinear filters," *Electronics Letters*, vol. 34, no. 8, pp. 705–706, 1998.

[14] V. Leung, M. El-Gamal, and G. Roberts, "Effects of transistor nonidealities on log-domain filters," in *Proc. of the 1997 IEEE International Symposium on Circuits and Systems*, vol. 1, Hong Kong, Hong Kong, June 1997, pp. 109–112.

[15] S. I. Liu and Y. H. Liao, "Table-based log-domain linear transformation filter," *Electronics Letters*, vol. 32, no. 19, pp. 1771–1772, 1996.

[16] J. Long and S. Colombano, "A circuit representation technique for automated circuit design," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 3, pp. 205–219, 1999.

[17] B. A. Minch, "Analysis, synthesis, and implementation of networks of multiple-input translinear elements," Ph.D., California Institute of Technology, Pasadena, CA, May 1997.

[18] ——, "Synthesis of multiple-input translinear element log-domain filters," in *Proc. of the 1999 IEEE Inter. Symposium on Circuits and Systems*, vol. 2, Orlando, FL, June 1999, pp. 697–700.

[19] ——, "Translinear analog signal processing: A modular approach to large-scale analog computation with multiple-input translinear elements," in *Proc. of the 20th Anniversary Conference on Advanced Research in VLSI*, D. S. Wills and S. P. DeWeerth, Eds. Los Alamitos, CA: IEEE Computer Society Press, 1999, pp. 186–199.

[20] ——, "Multiple-input translinear element log-domain filters," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 1, pp. 29–36, 2001.

[21] ——, "Synthesis of static and dynamic multiple-input translinear element networks," Cornell University, Ithaca, NY, Computer Systems Laboratory Technical Report CSL-TR-2002-1024, 2002.

[22] B. A. Minch, C. Diorio, and P. Hasler, "Multiple-input translinear element networks," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 1, pp. 20–28, 2001.

[23] B. A. Minch, C. Diorio, P. Hasler, and C. A. Mead, "Translinear circuits using subthreshold floating-gate MOS transistors," *Analog Integrated Circuits and Signal Processing*, vol. 9, no. 2, pp. 167–180, 1996.

[24] B. A. Minch, P. Hasler, and C. Diorio, "Synthesis of multiple-input translinear element networks," in *Proc. of the 1999 IEEE Inter. Symposium on Circuits and Systems*, vol. 2, Orlando, FL, June 1999, pp. 236–239.

[25] B. A. Minch and P. Hasler, "Floating-gate technology for digital cmos processes," in *Proc. of the 1999 IEEE International Symposium on Circuits and Systems*, vol. 2, Orlando, FL, May 1999, pp. 400–403.

[26] T. Morie, H. Onodera, and K. Tamaru, "A system for analog circuit design that stores and re-uses design procedures," in *Proc. of the 1993 IEEE Custom Integrated Circuits Conference*, vol. 5, San Diego, CA, May 1993, pp. 13.4.1–13.4.4.

[27] J. Mulder, M. Kouwenhoven, W. Serdijn, A. van der Woerd, and A. van Roermund, "Analysis of noise in translinear filters," in *Proc. of the 1998 IEEE International Symposium on Circuits and Systems*, vol. 1, Monterey, CA, May 1998, pp. 337–340.

[28] J. Mulder, W. A. Serdijn, A. C. van der Woerd, and A. H. M. van Roermund, "Dynamic translinear RMS-DC converter," *Electronics Letters*, vol. 32, no. 22, pp. 2067–2068, 1996.

[29] ——, "A current-mode synthesis method for translinear companding filters," in *Proc. of the Fourth IEEE International Conference on Electronics, Circuits, and Systems*, vol. 3, Cairo, December 1997, pp. 1419–1422.

[30] ——, *Dynamic Translinear and Log-Domain Circuits: Analysis and Synthesis*. Boston: Kluwer, 1999.

[31] J. Mulder, A. C. van der Woerd, W. A. Serdijn, and A. H. M. van Roermund, "General current-mode analysis method for translinear filters," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 44, no. 3, pp. 193–197, 1997.

[32] D. Perry and G. W. Roberts, "The design of log-domain filters based on the operational simulation of *lc* ladders," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 43, no. 11, pp. 763–774, 1996.

[33] S. Pookaiyaudom and J. Mahattanakul, "A 3.3 volt high-frequency capacitorless electronically-tunable log-domain oscillator," in *Proc. of the 1995 IEEE International Symposium on Circuits and Systems*, vol. 2, Seattle, WA, June 1995, pp. 829–832.

[34] M. Punzenberger and C. Enz, "Noise in instantaneous companding filters," in *Proc. of the 1997 IEEE International Symposium on Circuits and Systems*, vol. 1, Hong Kong, Hong Kong, June 1997, pp. 337–340.

[35] ——, "Noise in high-order log-domain filters," in *Proc. of the 1998 IEEE International Symposium on Circuits and Systems*, vol. 1, Monterey, CA, May 1998, pp. 329–332.

[36] ——, "New 1.2 V BiCMOS log-domain integrator for companding current-mode filters," in *Proc. of the 1996 IEEE International Symposium on Circuits and Systems*, vol. 1, Atlanta, GA, June 1996, pp. 125–128.

[37] D. Rhodes, "A design language for analog circuits," *IEEE Spectrum*, vol. 33, no. 10, pp. 43–48, 1996.

[38] R. Sarpeshkar, "Analog versus digital—extrapolating from electronics to neurobiology," *Neural Computation*, vol. 10, no. 7, pp. 1601–1638, 1998.

[39] E. Seevinck, *Analysis and Synthesis of Translinear Integrated Circuits*. Amsterdam: Elsevier, 1988.

[40] W. A. Serdijn, J. Mulder, P. Poort, M. Kouwenhoven, A. van Staveren, and A. H. M. van Roermund, "Dynamic translinear circuits," in *Analog Circuit Design: Volt Electronics; Mixed-Mode Systems; Low-Noise and RF Power Amplifiers for Telecommunication*, J. Huijsing, R. van de Plassche, and W. Sansen, Eds. Boston: Kluwer, 1999, pp. 3–32.

[41] W. A. Serdijn, J. Mulder, A. C. van der Woerd, and A. H. M. van Roermund, "A wide-tunable translinear second-order oscillator," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 2, pp. 195–201, 1998.

[42] W. A. Serdijn, J. Mulder, and A. H. M. van Roermund, "Shortening the analog design trajectory by means of the dynamic translinear principle," in *Proc. of the 1997 ProRISC Workshop on Circuits, Systems and Signal Processing*, Mierlo, The Netherlands, November 1997, pp. 483–489.

[43] H. Shibata, "Automated design of analog computational circuits using cell-based structure," in *Proc. of the 2002 IEEE International Symposium on Circuits and Systems*, vol. 2, Phoenix-Scottsdale, AZ, MAY 2002, pp. 53–56.

[44] M. Shojaei and M. Sharif-Bakhtiar, "A method for automatic design of analog circuits based on a behavioral model," in *Proc. of the 1998 IEEE International Symposium on Circuits and Systems*, vol. 3, Monterey, CA, May 1998, pp. 399–402.

[45] A. Thanachayanont and A. Payne, "A current-mode phase-locked loop using a log-domain oscillator," in *Proc. of the 1997 IEEE International Symposium on Circuits and Systems*, vol. 1, Hong Kong, June 1997, pp. 277–280.

[46] A. Thanachayanont, S. Pookaiyaudom, and C. Toumazou, "State-space synthesis of log-domain oscillators," *Electronics Letters*, vol. 31, no. 21, pp. 1797–1799, 1995.

[47] E. Tielo-Cuautle and A. Diaz-Sanchez, "An heuristic circuit-generation technique for the design-automation of analog circuits," in *Proc. of the 2003 IEEE International Symposium on Circuits and Systems*, vol. 1, Bangkok, Thailand, May 2003, pp. 193–196.

[48] A. Tola and D. Frey, "Study of different class ab log domain first order filters," *Analog Integrated Circuits and Signal Processing*, vol. 2, no. 2.

[49] L. Toth, Y. Tsividis, and N. Krishnapura, "Analysis of noise and interference in companding signal processors," in *Proc. of the 1998 IEEE International Symposium on Circuits and Systems*, vol. 1, Monterey, CA, May 1998, pp. 143–146.

[50] Y. Tsividis, "Externally linear, time-invariant systems and their application to companding signal processors," *IEEE Transactions on Circuits and Systems II*, vol. 44, no. 2, pp. 65–85, 1997.

[51] A. van Schaik and C. Jin, "The tau-cell: a new method for the implementation of arbitrary differential equations," in *Proc. of the 2003 IEEE International Symposium on Circuits and Systems*, vol. 1, Bangkok, Thailand, May 2003, pp. 569–572.