2021

Floating-Gate Design and Linearization for Reconfigurable Analog Signal Processing

Steven Michael Andryzcik II
West Virginia University, sandryzc@mix.wvu.edu

Floating-Gate Design and Linearization for Reconfigurable Analog Signal Processing

Steven M. Andryzcik II

Dissertation submitted to the
Benjamin M. Statler College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy
in
Electrical Engineering

David W. Graham, Ph.D., Chair
Roy Nutter, Ph.D.
Parviz Famouri, Ph.D.
Xin Li, Ph.D.
Edward Sabolsky, Ph.D.

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2021

Keywords: Analog, Mixed-Signal, Reconfigurable, Floating-Gates

Abstract

Floating-Gate Design and Linearization for Reconfigurable Analog Signal Processing

Steven M. Andryzcik II

Analog and mixed-signal integrated circuits have found a place in modern electronics design as a viable alternative to digital pre-processing. With metrics that boast high accuracy and low power consumption, analog pre-processing has opened the door to low-power state-monitoring systems when it is utilized in place of a power-hungry digital signal-processing stage. However, the complicated design process required by analog and mixed-signal systems has been a barrier to broader applications. The implementation of floating-gate transistors has begun to pave the way for a more reasonable approach to analog design. Floating-gate technology has widespread use in the digital domain. Analog and mixed-signal use of floating-gate transistors has only become a rising field of study in recent years. Analog floating gates allow for low-power implementation of mixed-signal systems, such as the field-programmable analog array, while simultaneously opening the door to complex signal-processing techniques. The field-programmable analog array, which leverages floating-gate technologies, is demonstrated as a reliable replacement to signal-processing tasks previously only solved by custom design.

Living in an analog world demands the constant use and refinement of analog signal processing for the purpose of interfacing with digital systems. This work offers a comprehensive look at utilizing floating-gate transistors as the core element for analog signal-processing tasks. This work demonstrates the floating gate's merit in large reconfigurable array-driven systems and in smaller-scale implementations, such as linearization techniques for oscillators and analog-to-digital converters. A study on analog floating-gate reliability is complemented with a temperature compensation scheme for implementing these systems in ever-changing, realistic environments.

Dedication

This work is dedicated to Brittany Stump for being there with me every step of the way. I could not have done it without you.

To my mother, Cindy Walden, thank you for a lifetime of guidance that helped me get here and for everything you did to aid me through my years of study.

A special thanks to my advisor, Dr. David W. Graham, for years of direction and the opportunity to work within his lab.

Acknowledgments

This work is based on the generous support by the National Science Foundation under Award No. CNS-1148815, Department of Energy's National Energy Technology Laboratory under Federal Grant DE-FE0031825, the Benjamin M. Statler Ph.D. Research Fellowship, and WVU's Lane Department Fellowship.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As humans, we exist in an analog world. The digital domain, which has invaded almost all aspects of our lives, is simply a representation of the underlying analog world. At high levels of the electronics stack, computer engineers are able to deal with digital representation by removing their access to analog information. Digital "ones" and "zeroes" become a visual and logical representation which detail the amount of electrons at a specific location on an integrated circuit (IC). While the layers of abstraction remove direct control of the analog data, digital computation opens the door to other marvels of engineering. From the digital domain curated by the computer engineer, software engineers are able to create comprehensive systems that are accessible, useful, and appealing to end users.

The separation between layers of abstraction in modern electronics design is useful not only for a division of labor, but also to create application spaces that are not burdened by unnecessary knowledge of lower-level constraints. However, we must avoid losing the knowledge of low-level systems and not allow stagnation at the bottom of the ecosystem to ensure full-stack development continues to make forward progress. In the past, analog design has been pushed to the side in favor of the digital domain. Analog electronics are seen as overly complicated and require years of study to fully comprehend. This is not far from the truth. Analog design is a costly process, and many of its engineers are required to have graduate degrees to even compete in the modern workforce.

The undeniable fact, however, is that analog design continues to be a steady market share in the electronics realm [2]. Not only is the IC, at the lowest level of the stack, governed

by analog device physics, but I bring us full circle to my opening statement: We exist in an analog world. Any time we want to send information from our world into the digital world, a conversion from analog to digital is required. This information can come from us in the form of keystrokes, which are analog buttons sending signals to a digital computer. Additionally, this information can come from sensors that detect light, sound, or heat, all of which exist exclusively in the analog domain and require an analog-to-digital converter (ADC). For any information to return to the analog world, it must be converted from digital to analog. When we view information on a computer monitor, we are viewing the analog light emitted from diodes (or other sources). It appears to us in the analog world after being passed through a digital-to-analog converter (DAC).

Traditionally, the sensor interface has been used to simply pass information to the digital domain for digital signal processing (DSP). DSP has been the standard way of processing sensor data since the rise of the digital computer and has taken a firm hold since the inception of the micro-controller. However, DSP comes with its own pitfalls, especially when related to power consumption. The DSP methodology utilizes an ADC to make all sensor data immediately available to the digital system for processing. Events are detected from the digitized signal and then decisions are made. The cost of the method is found in the requirement to digitize all data with an ADC, otherwise it can not be processed [3].

Considering the requirement of "always-on" DSPs, we begin to think how the sensor interface can be more than just a medium to pass information. It is at this point we can begin to implement analog signal processing (ASP) that goes beyond just data conversion. In the realm of ASP, we begin to see an implementation of event detection that was previously only seen in the digital domain. This type of data processing leverages the strengths of low-power analog systems without removing the capabilities of digital systems. Analog preprocessing allows for high-power-consuming digital systems and their required ADC to be kept "asleep" while the ASP monitors for specific events. When an event occurs, a "wake-up" signal is sent to the digital system, causing it to turn on and begin its power-hungry computation or radio transmissions [4–6].

The interface between our analog world and the digital domain acts as an unwavering toll that must be paid. Information must be converted and handled to be passed from one

domain to the other. Acknowledging that properly handling this interface is a requirement leads us to re-think how we can make analog design more accessible to ourselves and future generations of engineers. ASP opens up a new avenue for improving on not just the interface, but also the system as a whole. If implemented correctly, the floating-gate (FG) transistor is the key to this ASP space. The FG transistor can be the bridge between the digital way of thought that currently controls modern design and the older analog way of thought that has been shown to be a steadfast part of the electronics world.

I have dedicated my PhD to the study of FG transistors and their application in the analog and mixed-signal domains as a tool for circuit biasing, linearization, and temperature compensation. More importantly, I aim to show how FG transistors can be used as powerful tools for signal processing in reconfigurable sensor interfacing and help bridge the gap between old and new analog design. The FG transistor is most often thought of as a digital memory cell. Flash drives and solid-state hard drives are the common application of these devices. After being an oddity in analog electronics design for a prolonged period of time, in recent years, FG transistors have begun to find their place on the analog and mixed-signal side of design.

The first report of an FG transistor came out of Bell Labs in 1967 [7]. The first FG transistor was a proof of concept of its application as a memory source, much like the digital flash memory of today. Six years later in 1973, the first "non-memory" analog application of an FG transistor was reported and expanded upon in 1974 [8, 9]. The application leveraged the signal-to-noise performance of charge-coupled devices (a floating-gate input) to create a low-noise two-stage amplifier. In 1988, an FG transistor was first used as a tool to remove offset voltage in an amplifier, commonly referred to as "trimming" [10].

Analog applications of FG devices took a back seat to the more alluring digital memory applications until the late 1990s. At this time, FG transistors began to be used as programmable current sources for biasing analog circuits [11]. The ability to create a permanent bias current with a compact device caused many eyes to turn back to FG transistors, and was the first step towards large-scale integration.

In 2002, the first field-programmable analog array (FPAA) was reported that leveraged FG transistors as the core programmable element [12]. FPAAs are similar to their cousin, the

field-programmable gate array (FPGA), but instead of leveraging digital components, the FPAA makes use of analog circuits. Until 2004, this had been an unwieldy endeavor. Many core analog components require precise biasing; a stipulation that added massive overhead to the FPAA, complicating its design, understanding, and use. Since the combination of the FG transistor and the FPAA, these very-large-scale integrated (VLSI) systems have flourished and taken root in other areas of electronics applications, including as a tool for sensor interfacing and processing [13].

1.1 Objective

The objective of this work is to improve the accessibility and capability of reconfigurable analog signal processing via the implementation of FG transistors within FPAAs and ADCs. The objective will be achieved through a deeper understanding of utilizing FG transistors as the core component of reconfigurable systems like the FPAA and their peripheral circuitry. FG transistor utilization and reliability will be investigated with an emphasis on in-the-field application.

In terms of power consumption, ASP has been proven as a superior alternative to its DSP counterpart. DSP is still implemented, but the need for "always-on" processing is left to the analog domain. The problem lies in how ASP is implemented and approached. Application-specific integrated circuits (ASICs) have been the reasonable choice since ASP started to be seen as a viable solution. This work aims to show that reconfigurable ASP, through the use of FG-based FPAAs will open the door to widespread use of ASP. Making ASP available to more electronics designers will help improve overall system power consumption, especially in resource constrained applications without requiring the use of a costly ASIC.

1.2 Significance of Results

This work builds on the foundation of success with FG transistor-based FPAAs and broadens the application space for both FG transistors themselves and FG-based FPAAs. The results of this work illustrate that problems previously only solvable by custom designs

are now within the realm of the FG-based FPAA.

The combined work culminates in the advancement of electronics technology through improvements to the FG transistor's role as a core component of accessible reconfigurable ASP. The FG transistor's reliability has been improved through a proven temperature compensation scheme. With temperature independence and a realistic understanding of the long term effects of humidity, this work has improved FG transistor-based FPAAs and their ability to operate in-the-field.

Significant advancements to reconfigurable ASP are shown by demonstrating an ASP front-end in an audio based event-detector for vehicle classification utilizing an FG-based FPAA (Chapter 6). This task has been performed previously with a fabricated ASIC and will be compared directly, showing the competitive performance of the reconfigurable system. By implementing the system on an FPAA, this work demonstrates that ASP is usable by electronics designers without the need for high level training in analog systems. Other solutions that were previously only within the ASIC realm are now possible with reconfigurable systems.

The strengths of the FG transistor have been applied to not only the FPAA space, but also to prove its use for linearization in the modern analog-digital interface. To demonstrate FG transistor capabilities, a novel linearization method for a voltage-controlled oscillator (VCO) is presented. Continuing with the premise of increased signal processing capabilities via FG transistors, the linearized VCO is used to implement an ADC designed to leverage common FPAA components and is designed to operate within the bounds of the ASP space (Chapter 7). The significance of this work can be summarized by figure 1.1. This block diagram shows how the motivation, objective, and results fit together within the work presented.

1.3 Outline

The structure of this paper will be as follows: Chapter 2 will cover a general overview of analog sensor interfacing, FG transistors, FPAAs, and ADCs. Chapter 3 will cover a temperature compensation scheme for keeping array-based FG devices operating correctly outside of a controlled lab environment, which will be necessary if FPAAs will ever see widespread

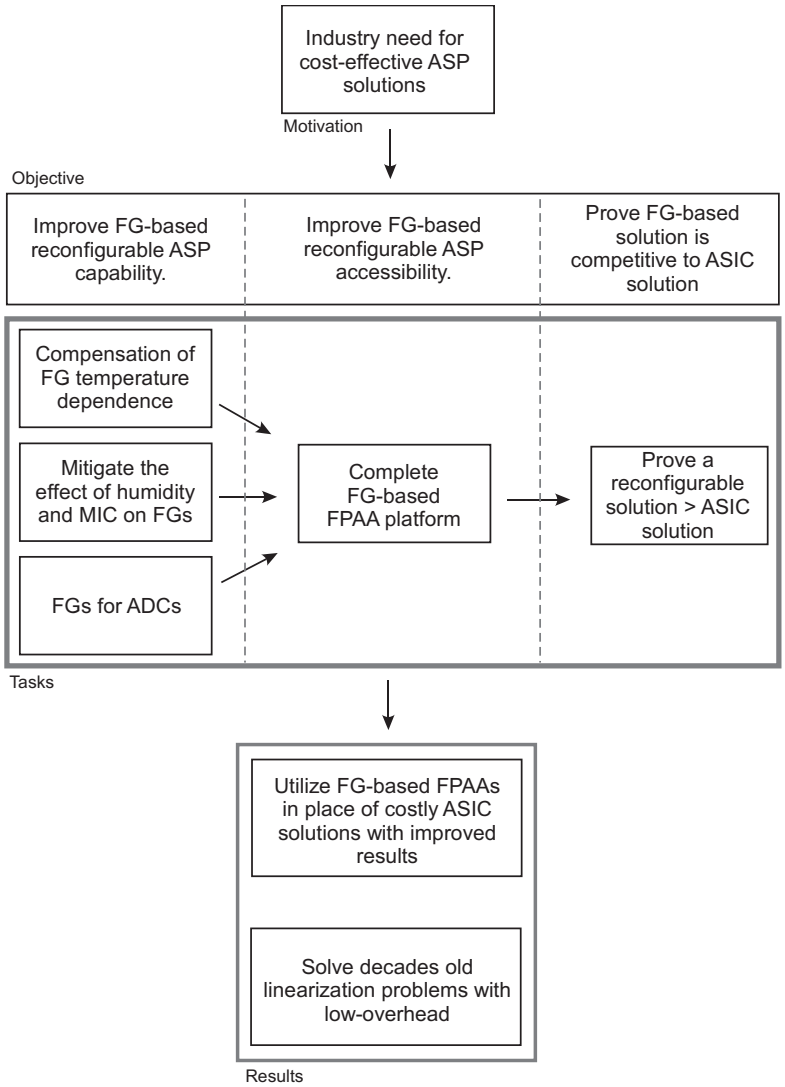


Figure 1.1: Summary of how the motivation, objective, tasks, and results fit within this work.

use. Chapter 4 will discuss characterization of the effects of mobile ionic charge on FG transistor performance, improving on in-the-field operation by allowing prediction of system startup transients. Chapter 5 will cover the current version of the FPAA designed in our lab, (the Reconfigurable Analog and Mixed-signal Platform (RAMP) version 1.1), showcasing FG transistor operation in a VLSI system and advanced ASP capabilities. The development and implementation of a wireless programming scheme using low-energy Bluetooth (BLE) communication as well as printed circuit boards (PCBs) for the testing and use of the RAMP system will also be covered. Chapter 6 will cover utilizing an FPAA (specifically the RAMP)

in an application as an acoustic vehicle detection and classification system, a task previously only done by custom systems. Chapter 7 will discuss a voltage-controlled oscillator-based analog-to-digital converter with FG linearization, showing how an FG transistor can be used in smaller-scale applications, leveraging its properties to solve an age-old problem with simple elegance. Chapter 8 will conclude the dissertation, discuss how this work has improved the reconfigurable ASP space, and how future researchers could continue to improve on the ideas presented.

Chapter 2

Background

This Chapter will cover an overview of the various subjects that will be presented in this work. A concise background of the subjects that will occur in subsequent Chapters helps set the stage for the upcoming discussion. The background will cover general information on the chosen subjects as well as the current state-of-the-art for relevant work. After a short discussion on generic analog sensor interfacing, this Chapter will cover analog signal processing (ASP) with a focus on the floating-gate (FG) transistor so that its usefulness can be highlighted for field-programmable analog arrays (FPAAs) and as a trimming method for data converters. The order was selected to best present the natural progression from analog sensing to advanced ASP and highlight how the FG transistor is the necessary component for wide-scale implementation, especially as computational tasks become more demanding.

2.1 Analog Sensor Interfacing

Low-power analog and mixed-signal systems are most often utilized as the interface between the real world analog domain and the digital ecosystem. The interface is regularly implemented between an analog sensor and a digital embedded system, acting as a data converter. The data converter can be implemented as a simple transconductor or other form of domain conversion such as current-to-voltage, resistance-to-frequency, etc. This is implemented in cases such as bridging the gap between an analog sensor that outputs a current and an embedded system that can only read voltage values.

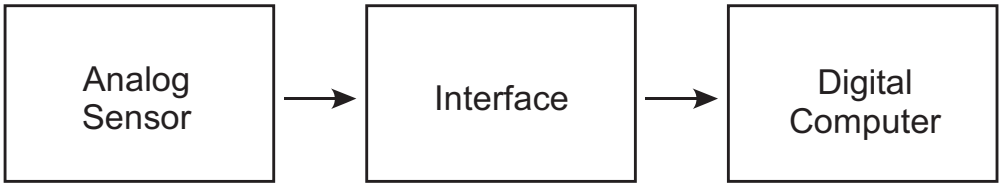


Figure 2.1: Analog sensor interfacing flow chart.

When sensor data need to be read or sent directly between a digital system and the analog domain, the interface can be implemented as a more complicated analog-to-digital converter (ADC) or digital-to-analog converter (DAC) depending on which direction the information needs to flow. As the state-of-the-art continues to focus on the Internet of Things (Iot) and small wearable devices, analog sensing has continued to be an ever-present necessity of electronics design.

2.2 Analog Signal Processing

Analog computation and pre-processing has been used in a wide variety of systems to improve energy savings, showing in some cases the equivalent of a 20-year leap in digital scaling [14]. Traditional analog pre-processing stages tend to be highly specialized application-specific systems, but developments in reconfigurable field-programmable analog arrays (FPAAs) [15, 16] have allowed these analog techniques to be applied to systems without *a priori* knowledge of the application space.

Digital signal processors (DSPs) are oft-used in wireless sensor nodes due to several merits (implemented with the sensor connected directly to the ADC as shown in Figure 2.2 (a)). For one, DSPs can be readily repurposed for a wide assortment of tasks, providing great benefit to node flexibility and programmability [1]. Additionally, digital systems have good immunity to electrical noise and environmental fluctuations, which make them a reliable choice for long-term applications. Recent applications of digital signal processing include a low-power feature extractor for speech recognition [17] and a front-end for an electrocardiogram acquisition system [18]. DSP energy-efficiency has historically followed Gene's Law, doubling every 18 months due to the power savings afforded by rapid technology scaling [19].

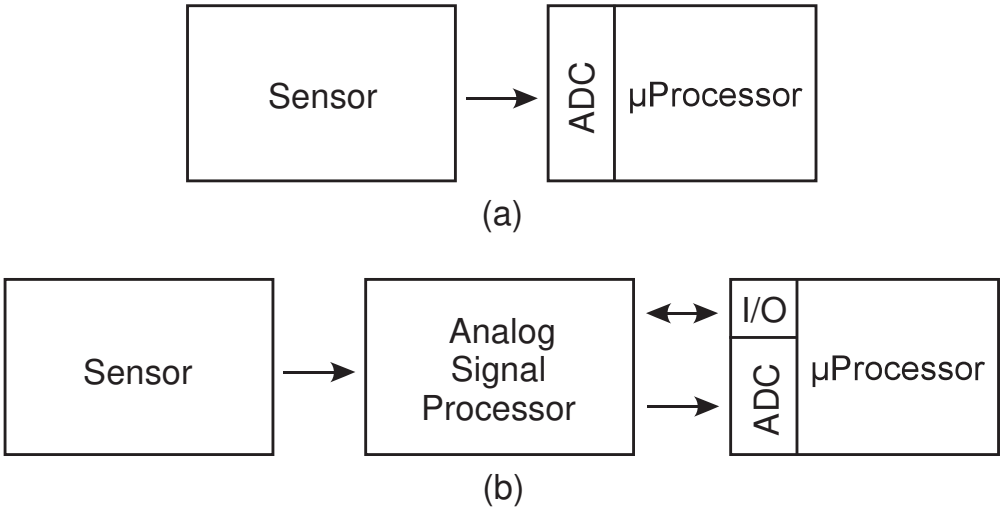


Figure 2.2: (a) All sensor information is processed directly by an embedded system. (b) An ASP is used as the interface between the sensor and embedded system, allowing for the embedded system to remain in a low-power state until it's processing power and radio capabilities are needed.

Yet, ADCs, which are at the heart of DSP-based sensor nodes that interface with analog sensors, have not kept up with Gene's Law [19] and are still a dominant contributor to overall power draw in digital sensor nodes [20]. Improvements in DSP energy efficiency are becoming infrequent as technology scaling becomes physically difficult and prohibitively expensive. Resource-constrained digital systems may suffer from high latency as well, which can be a concern in certain scenarios [21].

In contrast, ASP-based solutions offer real-time, low-power signal processing and often require less infrastructure than their digital counterparts. ASPs can directly interface with analog sensors, forgoing the need for always-on ADCs. Analog electronics endow designers with a rich assortment of powerful computational components at ultra-low energy cost [22, 23]. In fact, [19] notes that ASP energy-efficiency has historically led DSP technology scaling by 20 years.

These are some of the reasons why analog circuits are used in sensor interfaces like the capacitive sensing front end demonstrated in [24] and the delay line for ultrasonic imaging demonstrated in [25]. ASPs are also widely used in biopotential acquisition and analysis systems, finding recent applications in the current-mode front-end proposed in [26] and the electrocardiogram feature detector proposed in [27]. There have also been recent demon-

strations of ASPs in machine learning, such as the convolutional neural network processor shown in [28] and the framework for computing Gaussian kernels shown in [29].

An ASP "front-end" to an embedded system (implemented as shown in Figure 2.2 (b)) handles the task of initial event detection. In contrast, the system shown in Figure 2.2 (a) requires the Microcontroller Unit (MCU) to process all information from the sensor. As discussed, this requires the power-hungry ADC to be on at all times, constantly converting data to be read by the MCU. After implementing the ASP in Figure 2.2 (b), a new avenue for information processing is opened. Instead of keeping the MCU and ADC on at all times, the ASP is the only part of the system that is "always-on." Sensor data is often very sparse in relevant information, which leads to an "always-on" MCU/ADC to be a waste of precious power (especially in resource constrained systems).

The ASP monitors the sensor until a relevant event is detected. Once an event has occurred, it signals the MCU to wake up from its low power state, turn on its ADC, and begin processing information. The MCU can make the final decision on whether the information is relevant and if so, turn on its radio to broadcast the information to the appropriate destination. However, the methodology described leads to one key question: How can an ASP be calibrated to detect the correct event for a specific application? The application-specific integrated circuit (ASIC) has long been the answer to this question. An ASIC is designed for a specific application and as history has proved, does its job very well. However, implementing an ASP as an ASIC carries the same risks of any practical design.

In the field, environmental variables such as temperature can easily change or be completely unknown until deployment. As ASP is most useful in resource-constrained environments (where its power saving can actually be a boon), these environments are also the most prone to fluctuations in temperature. The ASIC designer must take into consideration this effect and try to design contingencies that will cover all possibilities. Given the volatile nature of weather and other environmental effects, this becomes a nearly impossible task, forcing a certain amount of error to be expected from in the field devices. Advances in reconfigurable analog circuitry have shed light on how this problem can be solved.

2.3 Floating-Gate Transistors

The FG transistor helps to enable the benefits of reconfigurable analog circuitry by providing a nonvolatile memory element and a tunable current source for precise analog biasing. An FG transistor is a transistor with a capacitor attached in series with its gate. This capacitor causes the gate of the transistor to have no DC path to ground. This lack of DC path causes the gate to be "floating" and gives the device its name. Charge can be applied or removed to the floating-gate, allowing for a myriad of uses. FG devices are most commonly implemented as flash memory in digital systems, but they also have a memory-like application in analog systems. In the digital domain, the FG can be used to create a memory cell as used in NAND-Flash memory.

In the case of the analog domain, the FG transistor can be used as a variable threshold device, allowing for applications such as mismatch correction, offset removal in differential pairs, threshold definition in flash ADCs, or for creating high-precision voltage references [30,31]. By adding or removing a specific amount of charge on the floating gate, the threshold of the device can be modified, and when a static DC voltage is applied to the control gate, a corresponding current is allowed through the transistor [32].

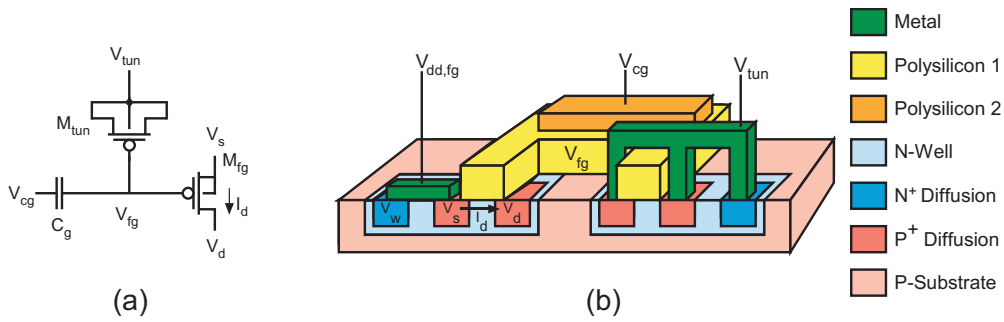


Figure 2.3: (a) Schematic of floating-gate transistor. (b) Cross-section of floating-gate transistor.

The control gate of the FG transistor is constructed via two polysilicon layers, as shown in Figure 2.3. Its construction is important, as the poly-poly layer interface produces a mostly linear capacitor: a metric important for FG reliability and accurate programming. There are two common programming mechanisms for modifying charge on an FG: Fowler-Nordheim (FN) tunneling and hot-electron injection [33].

FN tunneling is typically used as a global erasure for all FGs in an array since it is difficult to tunnel individual FGs. The procedure for FN tunneling is accomplished by significantly increasing the tunneling node capacitor voltage (referred to as V_{tun} in Figure 2.3). Under these conditions, charge is drawn off the FG node. A MOSCAP, M_{tun} (pFET transistor with source, drain, and body connected on one side, with gate on the other) is connected to the floating node and is used as the tunneling node capacitor. To initiate tunneling, V_{tun} is raised to a high voltage, typically higher than the reverse breakdown voltage of the source/drain, but less than the breakdown of the well-to-substrate junction [33]. The pFET FG transistor with pFET tunneling junction "V_{tun}" is shown in Figure 2.3 (a) with the FG transistor cross-section shown in (b).

Hot-electron injection (henceforth referred to as simply "injection") is typically used to add electrons to the FG. This is accomplished by raising the FG transistor's V_s to generate drain-current conditions favorable for high energy carriers to impact-ionize at the drain. A fraction of the ionized electrons disperse with enough energy to overcome the oxide barrier and inject onto the floating node [32], programming the FG up to the desired voltage V_{fg}.

The following equation represents the voltage seen on the floating node:

$$V_{fg} = \frac{Q + C_{cg}V_{cg} + C_dV_d + C_sV_s + C_{tun}V_{tun} + C_wV_w}{C_T} \tag{2.1}$$

The voltage is controlled by the charge stored on the gate, Q, and the capacitances seen at the floating node. This assumes that C_{cg}, C_d, C_s, C_{tun}, and C_w are the desired and parasitic capacitances between V_{fg} and the subsequent terminals: gate, drain, source, tunneling-junction, and well respectively. In most FG devices, it is assumed that the control gate capacitance C_{cg} is much larger than the parasitic capacitances and will dominate this relationship in this equation. With this in mind, the equation can be simplified to:

$$V_{fg} = \frac{Q + C_{cg}V_{cg}}{C_T} \tag{2.2}$$

Figure 2.4 details the schematic of a floating-gate transistor, as well as the IV relationship of the floating-gate after tunneling and after injection. State-of-the-art floating-gates are typically constructed from a P-type MOSFET (Metal-Oxide-Semiconductor Field-Effect-

Transistor) over an N-type. This is due to the process-control techniques that are implemented during the fabrication process [34]. Extended drain implants are employed to stop nFET devices from constantly "injecting" at normal operating voltages [35].

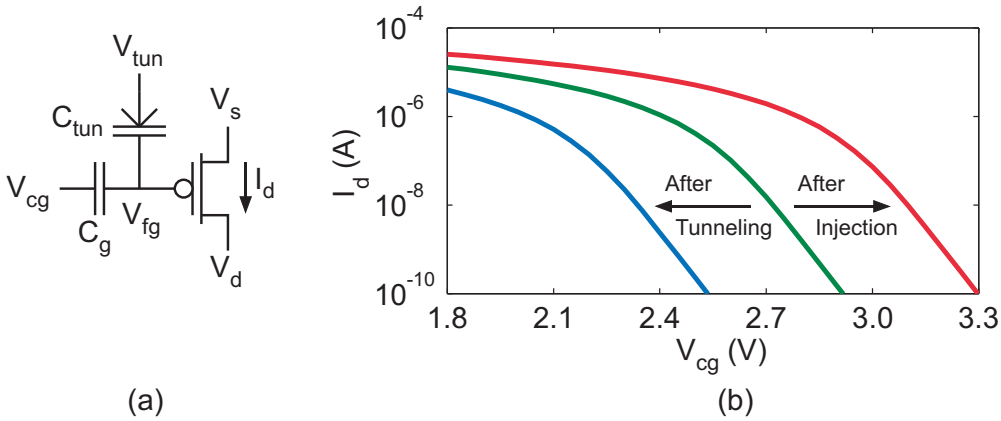


Figure 2.4: (a) Schematic of floating-gate transistor. (b) Floating-gate transistor IV relationship.

The benefit of an FG transistor lies in its operation as a variable threshold device. The two programming mechanisms (FN tunneling and injection) have been described. However, to implement an FG transistor in realistic application, a circuit must be utilized to control the programming mechanisms, specifically injection. The two common types of FG programming structures are pulse-based programming and continuous-time programming and are shown in Figure 2.5.

Pulse-based programming is a series of program and read cycles that are implemented until an FG transistor has the desired charge. From the perspective of a tunneled FG transistor (erased), a pulse-based method will cycle between program (inject) and read states to set a specific charge [36]. To initiate injection, there must be an adequate drain current present, and the source-to-drain voltage of the FG transistor must be large enough for some of the electrons to gain enough energy to overcome the barrier between the channel and the gate.

In the case of a 0.35µm process, a source-to-drain voltage of at least 4.5V must be applied to initiate injection. The first step is to increase the voltage seen by the gate, source, drain, and well of the pFET device by 4V. At this elevated voltage, the current is measured, and as the gate-to-source voltage has remained constant, will increase slightly as the parasitic

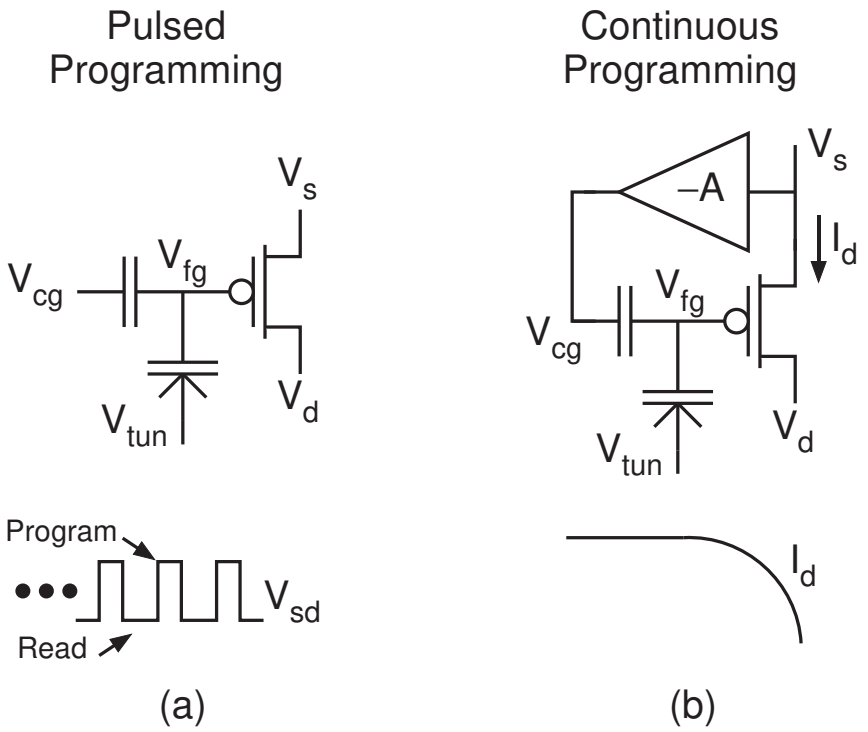


Figure 2.5: (a) Pulse-based programming method. (b) Continuous-time programming method.

capacitance to the substrate lowers V_{fg} relative to the supply.

The drain voltage is then dropped back down to 0V. With the other terminals held at the increased voltage, the source-to-drain voltage is large enough to initiate injection. The drain is then raised back to 4V, and the channel current is measured to see if the desired charge has been programmed. If the needed charge is not present, the process is repeated. If the FG transistor has been programmed to the correct level, the terminals are lowered to their operating voltages. The accuracy of a pulse-based method is dependent on the resolution of the pulse widths. To converge on a desired programming level, the pulse widths can be decreased so that a smaller amount of charge can be applied to the FG, allowing for precise targets to be reached. If the current is measured to be too high after the pulsed injection, the charge can be tunneled off, and the process begins from an erased state.

More-recent FG technologies have begun to implement continuous-time programming methods. This differs from the pulse-based programming method in a few ways. A pulse-based programming method typically requires off-chip sources and measurement equipment

to facilitate the read and write cycles needed, especially when variable pulse widths are required. A continuous-time method can leverage on-chip structures to program to a desired target. The standard continuous-time programming method is shown in Figure 2.5.

In its most basic form, the continuous-time method operates in one programming interval where the source-to-drain voltage is elevated to cause injection. The negative feedback between source and gate keeps the source voltage fixed throughout programming by modifying the gate voltage, causing a linear injection rate. However, the resulting programming level is controlled only by the length of time the elevated source-to-drain potential is present.

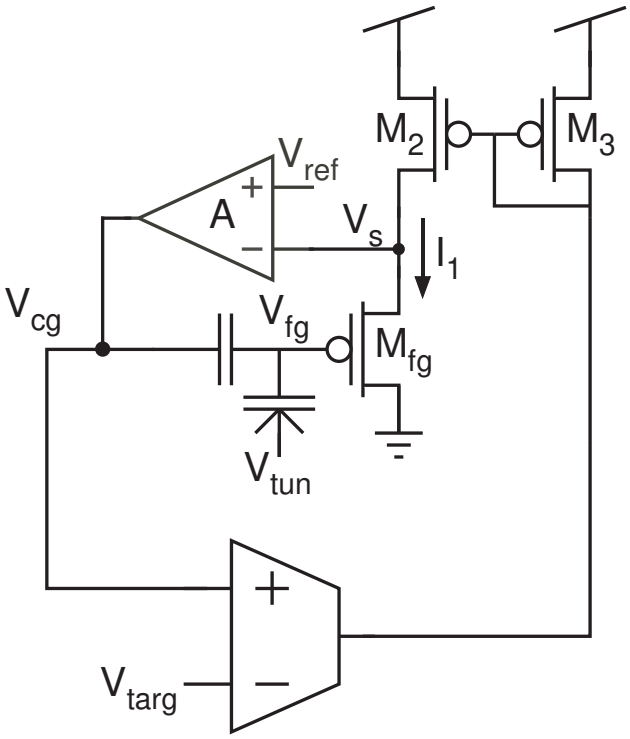


Figure 2.6: Continuous-time programming method with target setting capabilities.

The ability to set a target is enabled by the addition of an operational transconductance amplifier (OTA) as shown in Figure 2.6. The voltage applied to the negative terminal of the OTA is considered the target. As the voltage on the gate rises during programming, it eventually reaches the target voltage set on the OTA. When this occurs, the OTA turns off the current being supplied to transistors M_2 and M_3. Without an adequate current flowing through the FG transistor, injection stops. The applied target can be characterized for a

specific programming level on the FG [32].

2.4 Field-Programmable Analog Arrays

A field-programmable analog array (FPAA) is a reconfigurable integrated circuit (IC) that is the mixed-signal analogue to a field-programmable gate array (FPGA). Where FPGAs allow for post-fabrication synthesis of digital circuits, FPAAs allow for post-fabrication synthesis of analog and mixed-signal circuits. Designing and fabricating an ASIC is a costly and lengthy process. The first CMOS (complementary metal-oxide semiconductor) FPAA was reported in 1991, expanding on the principles established by the FPGA, and including analog/mixed signal computation [37].

FPAAs can be used to synthesize common circuits such as amplifiers or filters, making them useful for sensor interfacing applications. The reconfigurable, analog architecture of an FPAA allows for both general-purpose analog signal processing and complex, application-driven signal-processing tasks. However, while being able to address a wide range of applications due to the reconfigurable nature of the FPAA, the use of volatile shift registers and external (or permanently fixed) current sources lowered the practical usefulness of the system [38].

The FPAA showed promise, but the issues presented held back its use in mainstream systems. Volatile memory made battery operation unfeasible. If an FPAA was utilized out in-the-field, it would be required to operate on battery power in most useful scenarios especially where wireless operation was required. The FPAA was originally programmed serially via a shift register, loading configuration data into volatile memory arrays. These memory arrays required the FPAA to have a constant power source to keep the configuration data from being lost, resetting the device. This requirement led to higher general power consumption and overhead when replacing batteries. FPAAs had to be either plugged into a power source before batteries could be changed, or reprogrammed with every new set of batteries.

Volatile memory was not the only issue with the FPAA systems. Analog circuits most often require precise biasing. This can be achieved with power-hungry off-chip sources (not

viable for in-the-field applications). It can also be done utilizing static on-chip setups, causing the bias to be "set in stone" after fabrication and not tunable. Combining the need for precise, controllable bias currents and non-volatile memory, the FG transistor became a very alluring alternative. The FG transistor was able to be leveraged as both non-volatile memory and a programmable current source. This device opened the door to the modern FPAA and allowed for more realistic in-the-field operation [12, 16].

Implementing an FPAA with FG transistors as the core-memory component allowed for configuration data to be loaded into the FPAA and kept regardless of power. This non-volatility makes changing batteries a simple process and allows for lower power consumption during normal operation. Utilizing FG transistors as controllable current sources within an FPAA keep overhead costs low while allowing for precise biasing for as many analog circuits that can fit on the silicon die. An FG transistor has a small footprint, and its programming infrastructure can be reused for all devices in an array. The FPAA designed in our laboratory will be discussed in the following section.

2.4.1 The Reconfigurable Analog and Mixed-Signal Platform

The specific FPAA that will be discussed and used throughout this paper is the Reconfigurable Analog and Mixed-Signal Platform (RAMP). The current version of the RAMP (version 1.1) will be discussed in Chapter 5. The RAMP 1.0 (described at length in [39]) leverages a custom-designed FPAA structure to provide an abundant number of signal processing and computational elements.

The original version of the RAMP's printed circuit board (PCB) was designed to operate with either an Arduino or a TelosB wireless mote connected via a serial pin. The Arduino or TelosB was used to interface between a computer and the RAMP for programming.

A new, four-layer PCB has been designed that allows the RAMP to interface with a PanStamp wireless mote. Figure 2.7 shows the two sides of the PCB without population of the discrete devices (including the RAMP and PanStamp). Image (A) shows the top of the board, and (B) shows the bottom of the board where the footprint for the PanStamp can be seen (bottom center). The PanStamp is similar to the TelosB, but is housed in a smaller

package. By making this change, the PanStamp has been included directly on the RAMP PCB. Along with the change to mote interfacing, improvements were made to the external systems housed on the PCB.

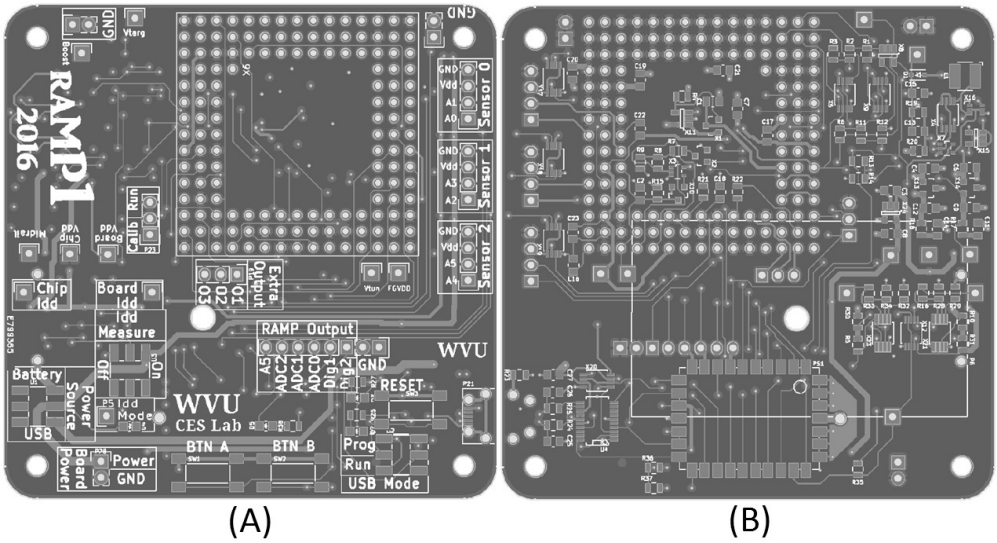


Figure 2.7: (a) Front side of new PCB. (b) Back side of new PCB.

The PanStamp NRG 2 [40] MCU is utilized by the RAMP for its IoT capabilities, small form factor, and low power consumption. Despite measuring 1.6 cm by 2.2 cm, the PanStamp NRG 2 has 32 kb of flash memory, a low sleep current of 1.5 μA, a maximum radio transmission power of 12 dBm, and AES encryption capabilities. The PanStamp not only enables transmission of ASP decisions made by the RAMP, but it also facilitates wireless reconfiguration of the RAMP IC. A photograph of a PanStamp NRG 2 soldered onto the backside of the RAMP PCB is displayed in Figure 2.8.

In addition to the changes to MCU, the PCB includes changes to the power-management sector. The power-management sector of the PCB houses the regulators used for generating the necessary voltages required by the RAMP (Digital Vdd, Analog Vdd, Mid-Rail, and the boosted voltage required for floating-gate injection) from either USB or battery power. This is also the sector where power measurements can be made. In the previous iteration of the PCB, a resistor was in series with both board power and chip power. This resistor was shorted using header pins. To measure how much power was being consumed by either the chip or the entire board, the short between the header pins was removed and the pins could

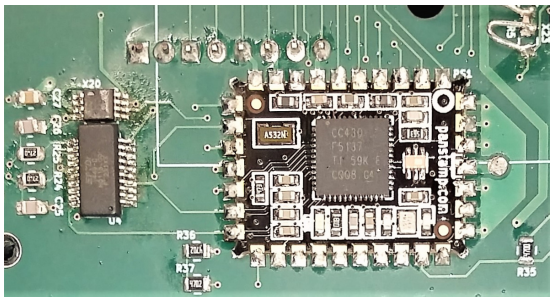


Figure 2.8: A PanStamp NRG 2 MCU affixed to the backside of the RAMP board.

be probed to measure power consumption. With the new design, whether or not the resistor is shorted or in series is controlled using a slide switch. The slide switch enables a digital switch that places the resistor in series with the power supplies. Included with the resistor is an on-board instrumentation amplifier with an output connected directly to an output pin. This allows for precision measurement of the power consumed by the total system as well as the power consumed by the RAMP itself.

In addition to including the PanStamp directly on the PCB, a USB transceiver was also included in the design. This transceiver allows for the board to be plugged in directly to the computer via USB port. The USB connection can be used for both power and data transmission or just one of the two. With the implementation of the PanStamp and transceiver, the RAMP, as well as all peripheral devices are now housed in one package.

Another improvement made by the inclusion of the PanStamp is more accessible wireless programming. A device called the PanStick is a peripheral PCB for the PanStamp. The PanStick houses a PanStamp, allowing for ease of operation. By using two PanStamps, one located on the RAMP PCB and one on the PanStick, commands can be sent to the FPAA via radio communication. The PanStamp on the RAMP queries the PanStick, receiving the new programming file, and builds the system on the RAMP. The populated, working form of the PCB that houses the RAMP can be seen in Figure 2.9.

The RAMP 1.1 will be discussed in Chapter 5. Along with the direct changes to the RAMP architecture, advances in MCU technologies constantly evolving, a new version of the RAMP PCB will be presented. The PanStamp, while an adequate tool for interfacing with

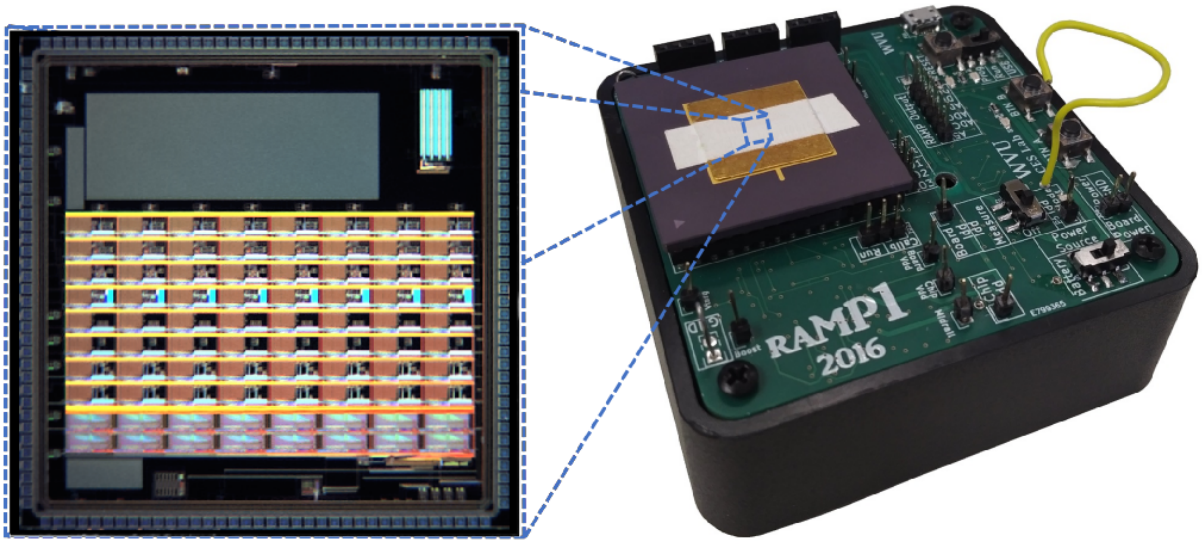


Figure 2.9: RAMP 1.0 die and PCB.

the RAMP, comes with one main drawback: it is manufactured by a small company without plans for future installments. To future-proof the RAMP architecture, the MCU platform has been converted to the Arduino family of MCUs. The new version implements a cutting edge IoT MCU with low-energy Bluetooth (BLE) to facilitate easy wireless reprogramming. By moving to the Arduino IDE, future upgrades to BLE devices within the Arduino family can be easily implemented to benefit the RAMP architecture.

2.5 Analog-to-Digital Converters

An ADC is a type of electronic system that converts data from the analog domain to the digital domain. Section 2.1 discussed the interface between the analog and digital domains. Section 2.2 highlighted how ASP can be used to improve on system performance and power savings at this junction. Section 2.4 covered a general overview of FPAAs and looked into the RAMP 1.0. The RAMP aims to fit in as an ASP while leveraging its reconfigurable nature to fit a myriad of different applications. But, to fulfill this goal, the RAMP needs to have reliable ADC capabilities. If the RAMP is being used in-the-field as an ASP and detects an event, the data need to be sent over the wireless medium. This requires an ADC to prepare the data for the MCU and radio transmission.

With an ASP front end in mind, two different types of ADCs will be investigated. To be applicable in an ASP scenario, ADCs are chosen for attributes that contribute to low-power consumption and selective sampling. The ADCs that will be discussed are a voltage-controlled oscillator-based ADC (VCO-ADC) and an asynchronous peak-sampling ADC (APS-ADC). The VCO-ADC is chosen for its simple and mostly digital design, allowing for it to operate at low supply rails. The APS-ADC is chosen for its ability to only convert data at local maxima and local minima, keeping the system dormant until relevant information has occurred. Before describing these specific ADCs, the metrics used for any ADC performance will be discussed.

ADC performance is generally looked at with two different groupings of error: quantization error and frequency response. Quantization error covers problems with offset error, integral nonlinearities, and differential nonlinearities. Quantization error is measured by inputting a slowly rising DC signal to the ADC and measuring its output. These errors manifest in step width, step height, and indefinite transfer points (a non-consistent transition from one output bit to the next).

The frequency response of an ADC system describes how the ADC performs with respect to an AC signal. A sinusoidal signal is input to the ADC and its output is tracked over a long period of time. The resulting output is converted to the frequency domain via a Fourier transform (FT). From the FT, the Spurious-Free-Dynamic-Range (SFDR) can be calculated. The SFDR is seen as the difference in decibels between the amplitude of the fundamental and the largest harmonic spike. The FT also enables the calculation of the Signal-to-Noise-Ratio (SNR) from equation 2.3.

$$SNR = \frac{FundamentalSignal}{NoiseFloor} - 10log_{10}[\frac{\frac{f_s}{2}}{BW}] \tag{2.3}$$

Where f_s is the sampling frequency of the ADC, and BW is the bandwidth of the logic analyzer used to read the output. The FT as well as the SNR can be used to calculate the Total Harmonic Distortion (THD) and Signal-to-Noise-and-Distortion-Ratio (SINAD). To find the THD, the amplitude of the second through sixth harmonics are measured and used with equation 2.4.

$$THD = 20log_{10}\sqrt{[10^{\frac{-V2}{20}}]^2 + [10^{\frac{-V3}{20}}]^2 + ... + [10^{\frac{-V6}{20}}]^2} \tag{2.4}$$

Where V2 through V6 correspond to the amplitude of the 2nd through 6th harmonic spurs. To calculate the SINAD, the SNR and THD values from equations 2.3 and 2.4 are used with equation 2.5.

$$SINAD = 20log_{10}\sqrt{[10^{\frac{-SNR}{20}}]^2 + [10^{\frac{-THD}{20}}]^2} \tag{2.5}$$

The final metric from the frequency response of an ADC is the effective number of bits (ENOB). The ENOB measurement is the best representation of an ADC's practical performance as it is a direct correlation to the systems dynamic range. Utilizing the SINAD value from equation 2.5 the ENOB can be calculated from equation 2.6.

$$ENOB = \frac{SINAD - 1.76}{6.02} \tag{2.6}$$

The ENOB can also be extracted by comparing the input (ideal) sine wave to the quantized output of the ADC. The root mean square (RMS) of both ideal and quantized signal is then calculated. The max deviation between the RMS of the ideal signal and the RMS of the quantized signal is found to be Q_a. Q_t is then calculated using the the weight of the least significant bit, q, from equation:

$$Q_t = \frac{q}{\sqrt{12}} \tag{2.7}$$

Utilizing Q_a and Q_t, the ENOB can be directly calculated from equation:

$$ENOB = N - log_2(\frac{Q_a}{Q_t}) \tag{2.8}$$

Where N is the number of bits in the ADC system. This alternative method of measuring the ENOB is useful for comparison to the ENOB value extracted from the FT, verifying the frequency response of the system. The metrics for ADC performance have been discussed to set a reference for the two ADCs that will be presented (the VCO-ADC and APS-ADC). The designs of both ADCs have inherent problems that will be addressed through the use

of FG transistors. Both designs have properties that are beneficial to ASP, especially in resource constrained environments.

2.6 Solving the Problems of Reconfigurable ASP with FG Transistors

The content within this Chapter has covered important background relating to the work I will be presenting from here forward. There is still a significant problem when dealing with a reconfigurable ASP. This subject requires high-level understanding of analog electronics, not just in design, but also in operation. The ability for electronics designers to implement the benefits of the front end ASP requires a reconfigurable system that is both accessible and powerful.

The research I have conducted to alleviate this issue will be discussed in the following Chapters. I have investigated how to improve preexisting reconfigurable FPAAs with their performance in-the-field both in the generic sense and with respect to application-specific design. The use of FG transistors as supplements to data converters has been explored with respect to system linearization, a requirement for any accurate signal processing. A study on the reliability of FG transistors will help establish rules for how and when FPAAs must be reconfigured to ensure optimal performance. The combination of these subsequent works result in the ability for the RAMP to be used reliably as a front end ASP in standard and harsh environments.

Chapter 3

Temperature Compensation of Floating-Gate Transistors in Field-Programmable Analog Arrays

**This Chapter was published at the 2017 Institute of Electrical and Electronics Engineers International Symposium on Circuits and Systems. It has been re-purposed to fit in this dissertation by removing redundant information that has already been discussed in previous Chapters. To read the original, unedited, published paper, see citation [41].

This Chapter is meant to solve an inherent issue that is present with very-large-scale integration (VLSI) systems that implement analog use of floating-gate (FG) transistors such as the Reconfigurable Analog and Mixed-Signal Processor (RAMP) described in Chapter 2.4.1. Analog pre-processing has been shown to be energy-efficient in a wide variety of low-power applications. Reprogrammable analog devices have leveraged this energy efficiency and applied it to a wider application space, but they still require accurate and stable bias currents for proper operation. For single-application devices, it is sufficient to create temperature compensation schemes that apply to a single bias current. But for reprogrammable or reconfigurable platforms which can be used in a variety of applications, temperature compensation must work well over a large range of potential bias currents and for a large number of different components. In this Chapter, temperature compensation for floating-gate transistors in a reconfigurable system which improves performance over a wide range of currents

and temperatures is presented.

One of the biggest hurdles in implementing reconfigurable analog systems lies in the infrastructure of the system. Temperature compensation is a particular challenge since the diverse application space demands a wide range of stable bias currents. Many reconfigurable analog systems utilize FG transistors to provide programmable bias currents [15, 16]. Unfortunately, the programmable bias currents generated by FG transistors are quite sensitive to temperature. There have been some successes in implementing temperature compensation for FGs employing large passive devices; however, these techniques are too area-hungry to be a viable option in dense FG arrays [42]. Others have employed a varactor on the FG node and use an additional voltage to modulate the capacitance at the FG node in response to temperature effects [43, 44]. This varactor-based method has been implemented on-chip and off-chip with great success, but has only been demonstrated for a temperature range between $25 - 43$ due to the small tuning range of the varactor. Moreover, no work has yet demonstrated the ability of a temperature compensation circuit to accurately regulate a multitude of currents across an array of FGs, as would be utilized in the RAMP, which is a field-programmable analog array (FPAA) system, without adding infeasible levels of power or area overhead.

This Chapter explores temperature compensation of FG transistors when the required number and values of currents remain unknown at design time. All plots depict measured results from an integrated circuit fabricated in a standard 0.35μm CMOS process.

3.1 Floating-Gate Device Performance in FPAAs

FPAAs often utilize large arrays of FG devices [15, 16] as bias generators. Within the FPAA system referenced throughout this work and described in Chapter 2.4.1, there are over 300 biases realized by FGs which control elements ranging in granularity from current-starved inverters to bandpass filters. This wide range of elements necessitates a very wide range of bias currents — creating a need for a temperature compensation circuit which can stabilize a wide range of currents.

Before operating an FG device as a current source, it must first be programmed. Methods

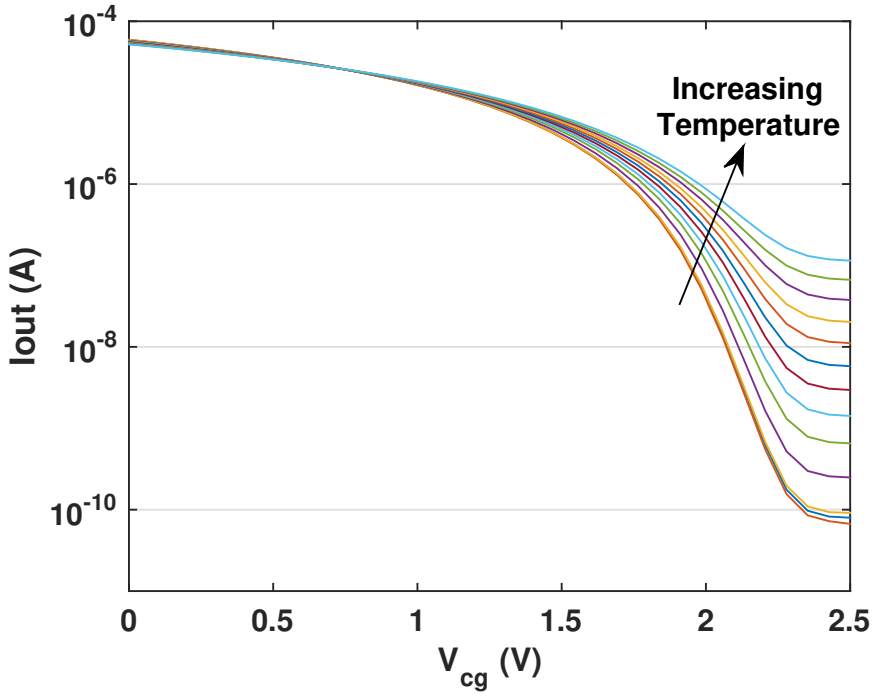


Figure 3.1: Temperature dependence of an FG transistor. The plateaued currents for $V_{cg} > 2.25V$ are artifacts of many junction connections to a single global connection where the current reading was taken. As a consequence, their collective leakage current becomes non-negligible and is manifested in these measurements. The current in a single FG transistor continues below these values.

for programming FGs have been discussed at a high level in Chapter 2.3 and with reference to FPAAs in Chapter 2.4.1. A transistor's operational characteristics will have an inherent dependence on temperature. Furthermore, transistors operating in the sub-threshold region, which is our main application operation area, experience more extreme changes (exponential) in channel current for a change in temperature than when in above-threshold operation.

Figure 3.1 shows the extent of temperature effects on an FG transistor. This example demonstrates for a single programmed FG device with a fixed V_{cg} — the voltage node for setting the target current bias — that the output current wildly varies with temperature change. Holding V_{cg} constant is the traditional method for setting target bias currents in FPAAs; however, a compensation circuit to modify V_{cg} in response to a change in temperature is clearly needed.

To generate temperature compensation, we use an FG current multiplier, as illustrated

in Fig. 3.2 [44]. Considering that M_{REF} and M_1 have the same W/L, the charge stored on their respective FGs can be modified and used to ratio the reference current I_{REF} to I_{M1}. Operating an FG in the sub-threshold saturation region can be characterized by the following:

$$I_d = I_o e^{-\kappa V_{fg} q/kT} e^{V_s q/kT} e^{V_d/V_A} \tag{3.1}$$

where all voltages are referenced to the well potential, V_{fg} is the FG voltage, V_A is the Early voltage, and $q\kappa/kT$ defines the subthreshold current slope with kT/q being the thermal voltage (k = Boltzmann constant). V_{fg} can be approximated as:

$$V_{fg} = \frac{Q_{FG}}{C_T} + \frac{C_{cg}}{C_T} V_{cg} + \sum \frac{C_{par}}{C_T} V_x \approx \frac{Q_{FG}}{C_T} + \frac{C_{cg}}{C_T} V_{cg} \tag{3.2}$$

where Q_{FG} is the amount of charge on the FG, C_T is the total capacitance seen at the FG node including parasitic capacitances, C_{cg} is the control gate capacitance, and C_{par} are the parasitic capacitances with V_x being the terminal voltages coupled to V_{fg}. The right-hand side is a reasonable approximation given that C_{cg} represents the majority of total capacitance C_T. Then, incorporating (3.1) and (3.2) into the FG current multiplier topology of Fig. 3.2 renders the following output current relationship:

$$I_{M_1} \approx I_{REF} exp \frac{q\kappa(Q_{M_{REF}} - Q_{M_1})}{C_T kT} \tag{3.3}$$

where $Q_{M_{REF}}$ and Q_{M_1} signifies the amount of charge on their respective FGs. Equation (3.3) shows that all temperature dependence is removed from the output current for charge-matched FGs in the mirror topology. For FGs with unmatched charges in the mirror, there still exists a temperature dependence, but its effects are greatly diminished compared to an FG without temperature compensation. Unmatched charges have a temperature dependence that can be characterized for the following two cases: $Q_{M_1} < Q_{M_{REF}}$ and $Q_{M_1} > Q_{M_{REF}}$, where a larger charge amount is the result of fewer electrons on the FG and will correspond to a smaller current. Defining the exponential terms in (3.3) as β with the exception of T

$$\beta = \frac{q\kappa(Q_{M_{REF}} - Q_{M_1})}{kC_T} \tag{3.4}$$

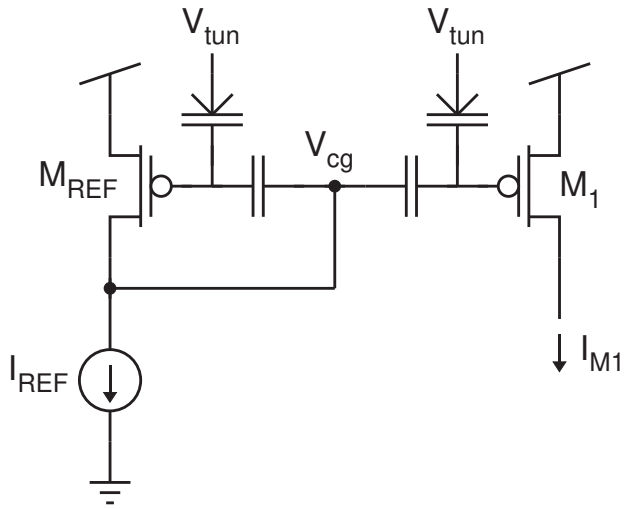


Figure 3.2: Floating-gate current mirror.

gives the following expression of I_{M_1} for the two differing charge cases:

$$I_{M1} = \begin{cases} I_{REF} e^{\beta/T} & Q_{M_1} < Q_{M_{REF}} \Rightarrow I_{M_1} > I_{REF} \\ I_{REF} e^{-\beta/T} & Q_{M_1} > Q_{M_{REF}} \Rightarrow I_{M_1} < I_{REF} \end{cases} \tag{3.5}$$

Taking the derivative of (3.5) with respect to temperature yields a negative temperature relationship for $I_{M_1} > I_{REF}$ and a positive relationship for $I_{M_1} < I_{REF}$

$$\frac{dI_{M1}}{dT} = \begin{cases} -\frac{I_{REF}\beta}{T^2} e^{\beta/T} & Q_{M_1} < Q_{M_{REF}} \\ \frac{I_{REF}\beta}{T^2} e^{-\beta/T} & Q_{M_1} > Q_{M_{REF}} \end{cases} \tag{3.6}$$

These temperature coefficients will be manifested in the current measurement slope over a temperature range and become more apparent with larger differences in charge.

3.2 FG Temperature Compensation

The application presented in this Chapter is applied to our FPAA, which is called the Reconfigurable Analog and Mixed-signal Platform (RAMP), discussed in Chapter 2.4.1 [15]. To provide temperature compensation to such a large-scale system, we are leveraging the FG current mirror shown in Fig. 3.2. As stated in Section 3.1 regarding FPAAs, the RAMP utilizes FGs to provide precise, but temperature-dependent current sources. The FG

current mirror is used to generate a control gate (V_{cg}) voltage that responds to changes in temperature and reduces its effect on current variation.

3.2.1 System Architecture

Our RAMP includes over 300 controllable current sources generated from FGs to be used as biases for the different circuits included within the RAMP. As an FPAA, the RAMP utilizes "computational analog blocks" (CABs), as building blocks for post-fabrication reconfiguration. The CABs can include simple devices or full circuits. By routing the CABs together and making connections to the FG biases, analog and mixed-signal systems can be synthesized directly on the RAMP. Figure 3.3 shows a block-level diagram of the RAMP and how the bias currents, as well as how temperature compensation fits in to the system as a whole.

To implement the FG current mirror for temperature compensation, a "reference" transistor is set-up in diode connection to dynamically set the global V_{cg} so that a steady current will be seen on any other FG connected to the mirror topology. To accomplish this, a specific CAB has been added to the RAMP that allows for the diode connection. Figure 3.3 shows the full schematic of the compensation circuit within an FG array. Transistor M_{REF} is placed in diode connection via the current mirrors comprised of transistors M_A-M_D. Transistor M_D mimics the behavior of the M_{REF}, specifically at its drain. All reference transistors are sized identically to ensure the same current flowing from the drain of M_{REF} is also flowing from the drain of M_D. The drain of M_D is then connected to the global V_{cg} node, allowing for the complete diode connection of M_{REF}.

This topology employing two current mirrors out of the reference FG transistor is used instead of a simple diode connection (i.e. drain connected to the control gate) to ensure that the drain of M_{REF} is kept at a relatively fixed potential. With a conventional diode connection, any fluctuations at the drain of the reference transistor would be parasitically coupled to the FG as indicated by (3.2), causing potentially large fluctuations in channel current. By using the current mirrors to create the diode connection, the voltage seen at the drain of the reference transistor will be more constant. A similar method is employed

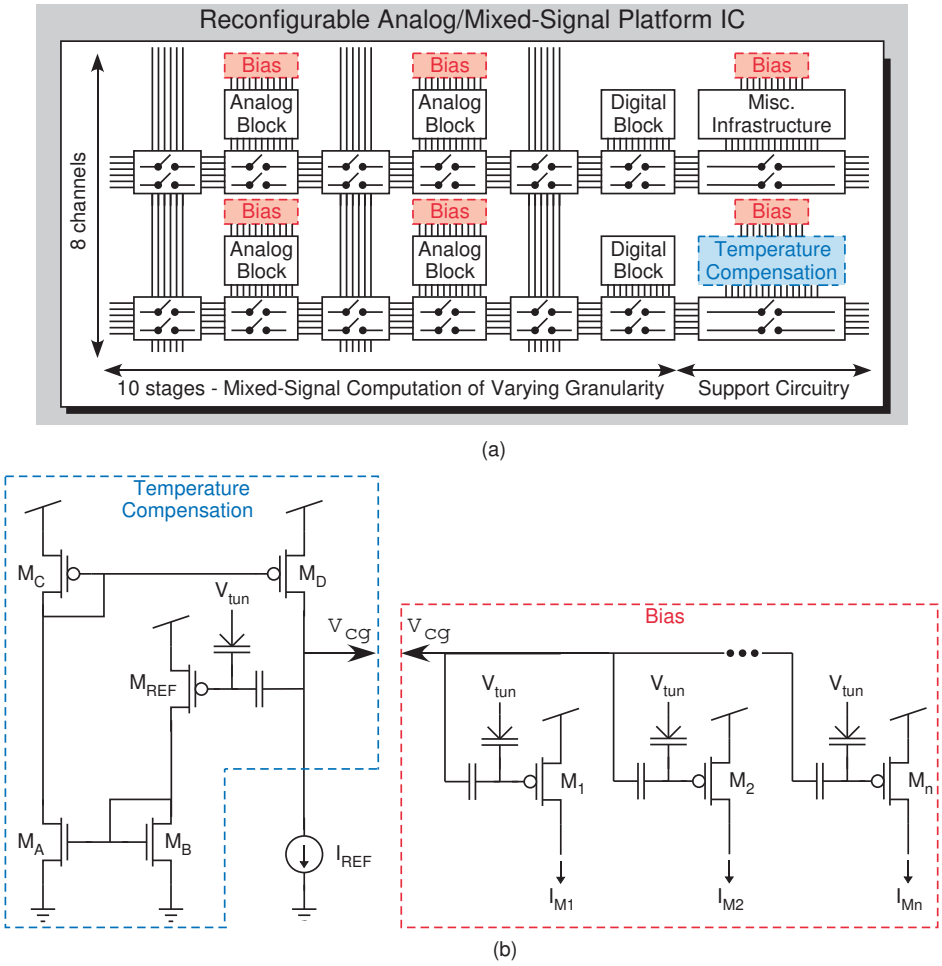


Figure 3.3: (a) FPAA block diagram showing the position of the specialized CAB which houses the FG temperature compensation structure. (b) Floating-gate temperature compensation structure showing connection to global V_{cg}. Currents $I_{M1} - I_{Mn}$ source to an nFET current mirror before connecting to the CAB circuits.

with the FGs used as current references. Instead of connecting the FG directly to a circuit as a bias, a single nFET-based current-mirror is used to ensure that any fluctuations in the circuit will not affect the FG output.

3.2.2 System Programming

The first crucial step in programming our temperature compensation system is to determine a value of I_{REF}. I_{REF} is the stable, temperature-independent, reference current which the rest of the system will refer to as the temperature of the environment fluctuates. The current can be generated from a number of different reference circuits as described in [45].

The closer I_{REF} is to the individual mirrored current values, the more accurately the system will be able to compensate. For this demonstration within the RAMP system, we chose a value typical of low-power analog bias currents – $10nA$. Generally, this choice should be made by matching I_{REF} to the average value of the expected currents of M_1-M_n or the current which is most sensitive to temperature that is being implemented in the design.

For a given I_{REF}, when M_{REF} is programmed and diode connected, a particular V_{cg} will occur. With a known value of V_{cg}, we can characterize the programming of M_1-M_n. The programming is controlled by a continuous-time feedback circuit, similar to the one presented in [46]. The continuous-time programmer injects the FG to some value dependent upon a user-specified target voltage. At a given V_{cg}, this FG value will generate some specific current in M_n which can be stored in a look-up table to relate the value of the programmer's target voltage to the resultant current in M_n for a given V_{cg}. We can then use this look-up table to program the currents of M_1-M_n to any specified value.

A comparative view on the effectiveness of temperature compensation is demonstrated in Fig. 3.4. This shows the temperature compensation performance relative to the room temperature programming target from -25 to 85. Each line in Fig. 3.4(a) corresponds to a different ratio between the current I_{REF} and the current flowing through transistor M_n (depicted in Fig. 3.3(b)). The current I_{Mn} was programmed at room temperature (25) and adjusted via the FG temperature compensation structure shown in Fig. 3.3.

3.3 System Performance

The large variance in current ratios shown in Fig. 3.4 is a constraint imposed by the nature of the RAMP, allowing for the device to span a wide range of applications without limiting the range of available bias currents. The best case is when the current targets between the FG (M_n) and FG reference (M_{REF}) are equal. As predicted by (3.3), ratios other than $1:1$ will result in less temperature compensation. With these ratios, which are a result of differing FG charges, a positive (negative) trend versus increasing temperature is the result of a negative (positive) difference in the numerator of (3.3). However, despite not working as well as a $1:1$ ratio, these cases still perform better than an uncompensated

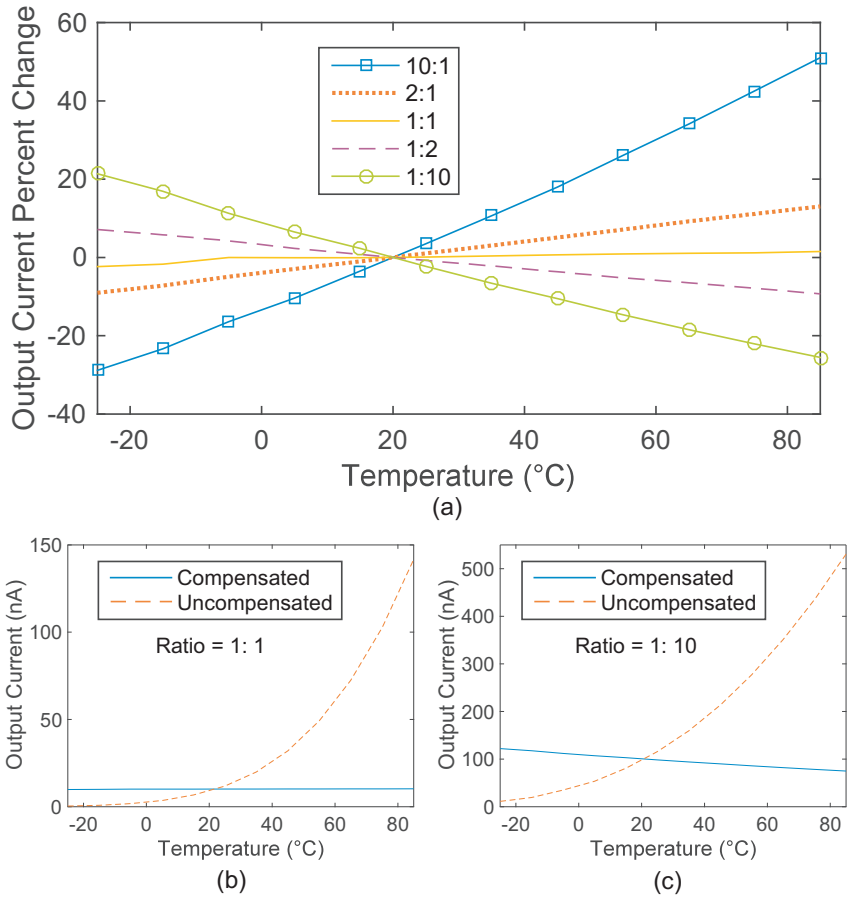


Figure 3.4: (a) Percent change in output current of an FG with temperature compensation normalized to room temperature. (b) and (c) show the effects of compensation compared to an uncompensated case for a ratio of 1:1 and 1:10 respectively.

scenario, as shown in Fig. 3.4(c).

Figure 3.1 shows the exponential dependence of temperature effects on uncompensated FGs. Due to the nature of the RAMP, target currents used for the operation of specific, synthesized circuits will be unknown until the end-user picks a desired application. Subsequently, depending on the complexity of the synthesized design, there will be more than one target current, at more than one target value. This calls for the ability to apply temperature compensation for a wide range of FG injection targets from a single targeted I_{REF}.

To show the advantage of temperature compensation in the RAMP system, a current-controlled ring oscillator has been synthesized from one of the CABs. This circuit utilizes an input current, I_{in} (generated by one of the FGs) to starve its odd number of inverters,

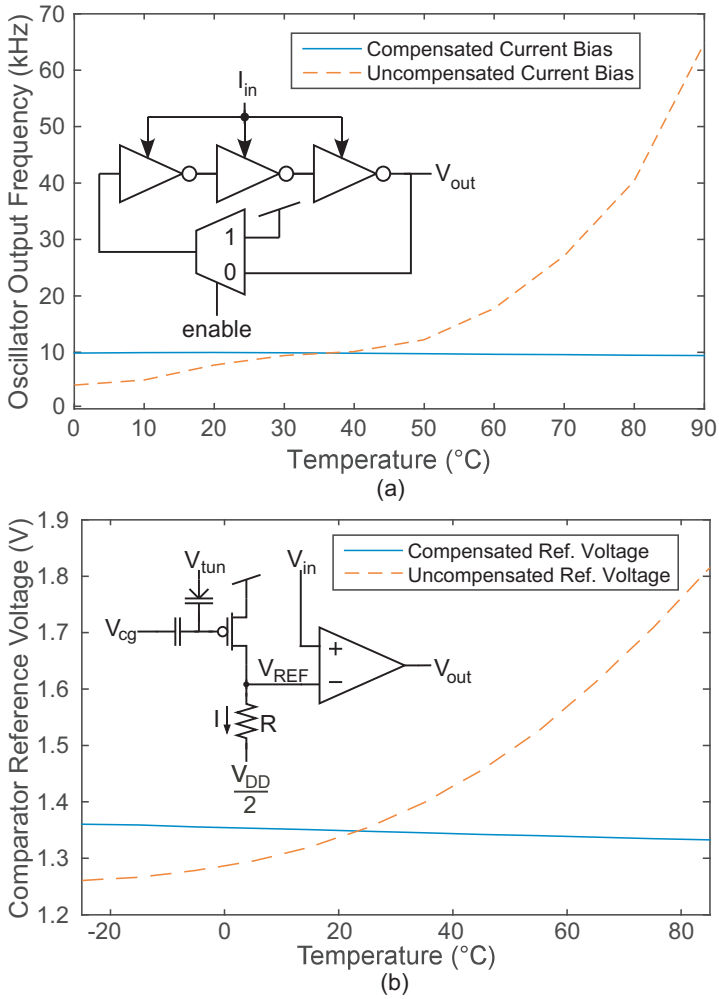


Figure 3.5: (a) Ring oscillator frequency output with respect to temperature for a compensated and uncompensated FG current bias. (b) Programmed comparator reference value with respect to temperature for a compensated and uncompensated FG current bias.

producing output frequency oscillations proportional to the input current (Fig. 3.5(a)). For an output frequency of 10kHz, I_{in} was set to $33nA$. The I_{REF} current chosen for the temperature compensation system was set at $20nA$. Utilizing a ratio of $1 : 1.65$ for I_{REF} and I_{in}, the compensation scheme is able to decrease the fluctuations of the bias current over the temperature sweep of $0 - 90$.

The ring oscillator was tested with an uncompensated and a temperature-compensated FG. The oscillator output frequency for both cases is shown in Fig. 3.5(a). The current bias using an uncompensated FG changes exponentially with increasing temperature while the

compensated current bias remains close to the same value for the full temperature sweep, showing the benefit of the compensation scheme.

A synthesized comparator example is shown in Fig. 3.5(b). The FG is programmed to draw a current across a resistor to set a desired reference voltage level, V_{REF}. The uncompensated and compensated V_{REF} measurements are shown in Fig. 3.5(b) where the reference current was set to $10nA$ and the current through the resistor was programmed to be $100nA$ for a ratio of $1:10$. The temperature compensation ratio does not represent an ideal case, but greatly outperforms the uncompensated case.

3.4 Conclusion

A temperature compensation circuit was presented for FG-dense structures such as an FPAA to improve performance over a temperature-varying environment. It is able to compensate for a multitude of FG biases with various output currents, which is representative of the variable nature of FPAA usage. The compensation system has been demonstrated to work with an array of FG current sources intended for biasing components such as an oscillator or other analog blocks. Its performance has been tested as a part of the larger FPAA system and is shown to improve current bias stability.

Chapter 4

The Effect of Mobile Ionic Contaminants on Floating-Gate Transistors

The ability for FG transistors to operate as intended in realistic in-the-field applications requires two main characterizations. Temperature compensation, discussed at length in Chapter 3, is the first factor that must be addressed. An adequate temperature compensation scheme, specifically for FGs in an array, allows for FG-based FPAAs such as the RAMP to be deployed in environments with fluctuating or unknown temperatures. The other main factor for widespread deployment is a characterization of FG reliability in long-term scenarios. This specifically looks at how FGs are able to retain charge over long periods of time and what FG transistor performance looks like after a device has been off for a long duration.

The studies that have been conducted regarding FG transistor lifetimes have looked at general device performance [47] and their ability to perform offset cancellation [48]. Both of these studies are very beneficial to understanding the long-term performance of FG transistors. However, there is still room for an additional characterization study that focuses on long-term performance of FG transistors in the scope of FPAAs and other array-based biasing structures. The paper covering general long-term device performance for FG transistors was reported with respect to an end of life criterion of 70% loss in cell current. This criterion is certainly helpful and gives good perspective of utilizing an FG transistor in excess of 100 years [47]. However, current losses much smaller than 70% would have dramatic impact on an FG transistor being used to generate a precise analog bias current.

From [47, 48], the mechanisms for charge loss are broken down into two categories. The first category represents "short-term" charge loss. Short-term charge loss is seen as a current drift directly after programming. The drift is attributed to interface trap sites settling to a new equilibrium after programming. Drift is typically observed within the first few days after programming, after which, no drift is observed. The other charge loss category refers to "long-term" charge loss. Long-term charge loss is attributed to two different mechanisms which occur in different, but overlapping, time frames.

The "early" long-term charge loss is due to mobile ionic contamination (MIC). MIC refers to the presence of ionic contaminants in semiconductor devices [49]. The contaminants are made up mostly of alkali ions, but are mainly attributed to Na^+. The Na^+ contaminants are observed in the gate oxide layers of MOS transistors. The contaminants are free to move about, and their mobility is accelerated by temperature and electric field. The presence of MIC in a MOS transistor results in a mobile ionic charge. This mobile charge can cause long-term changes in the threshold voltage of the transistor. This is particularly bothersome for FG transistors where precise control of the threshold voltage is the main avenue for creating tunable current sources.

Na^+ is the most common contaminant due to its high mobility (due to its small atomic radius) and its widespread presence. MICs can come from a myriad of sources within the fabrication process, the environment, or contact with humans. Due to this, ensuring a clean fabrication process alone will not eliminate all sources of Na^+ contamination. In regards to long-term charge loss, MIC occurs "early" within the first few years of an FG transistor's lifetime and contributes around 20% of total charge loss [47].

The second "long-term" charge loss mechanism is thermionic emission (TE). TE concerns the release or expulsion of electrons from an electrode by way of its temperature, or more plainly, the release of energy from heat. The TE mechanism is a steady loss over time and accounts for around 80% of charge loss, on the scale of up to 100 years [47]. The effect of TE can be accelerated with higher temperatures, allowing for long-term charge loss to be measured within days instead of years.

Given the two mechanisms discussed, the amount of charge lost is a function of both temperature and time. Given that Q(0) is the initial charge on an FG transistor and Q(t)

is the amount of charge after some time, t, the relationship between charge before and after t is found to be:

$$\frac{Q(t)}{Q(0)} = exp[-tv * exp(\frac{-\varphi_B}{kT})] \tag{4.1}$$

where v is the relaxation frequency of electrons in polysilicon, φ_B is the Si-SiO_2 barrier potential, k is the Boltzmann constant and T is the temperature. Adding or removing charge to an FG transistor modifies the threshold voltage, V_{T0}, of the device. By measuring the change in threshold voltage right after programming, $V_{T0}(0)$, and then measuring the threshold voltage after a long period of time, t, (applied as $V_{T0}(t)$) the charge loss can be calculated from:

$$\frac{Q(t)}{Q(0)} = \frac{V_{T0}(t) - V`_{T0}}{V_{T0}(0) - V`_{T0}} \tag{4.2}$$

where $V`_{T0}$ is the threshold of a transistor with no FG charge (i.e., a non-FG device) [48]. To facilitate these measurements, the FG transistor will be exposed to high temperatures and humidity, speeding up the charge-loss mechanisms by a known amount.

The issue with measuring the effects of MIC on a FG transistor lies in the inability of MIC effects to be accelerated by increased temperature. In fact, the introduction of temperature has been shown to "bake" off MIC, removing its effects on the semiconductor, as shown in [50]. The experiment presented in this Chapter aims to speed up the process of introducing MIC, and measure its effects on a FG transistors startup transients. The problem statement is as follows: An FG transistor that has been in operation for a long period of time will be measured for startup transients. The IC will be baked to remove the MIC present and decrease startup transients. The IC will then be subjected to humidity to reintroduce MIC and its effects on the startup transients. The experiment will verify the presence and effects of MIC. Given a verifiable method for controlling the presence of MIC, it will allow FPAA designers to test for and predict the effects of mobile ionic charge.

When FG transistors are used in an FPAA, FGs are only "turned on" when the application requires their specific bias. On an FPAA, there are hundreds of FG devices being unused when considering any possible application. Not all devices are connected, and therefore, not

all FGs are needed to generate a specific bias. When not being used, the FG transistors are connected so that all terminals are at the same potential, forcing the device to be off and ensuring it does not consume any unnecessary power.

When considering the practical use of FPAA systems, they would be deployed on site, and then programmed for a specific application. The system would perform this task for a set amount of time. After the initial task is performed, the FPAA could be cycled to another task, perhaps very quickly to monitor a different metric. Or, the FPAA could perform one task for a long period of time, and then be reprogrammed wirelessly to start a different task, made from a different set of its core components.

Both of these situations present a scenario where the FPAA is using a different set of internal components for different amounts of time. The FGs on the FPAA will go from stages of powered off inactivity to stages of powered on precise bias production. If the performance of the FGs directly after being turned on is different than that of an FG that has been on for a long period, that must be characterized and taken into account. The experiment aims to verify the effects of MIC on the startup transients after a FG transistor has been "off" for some period of time. The experiment will be presented as a two-factor nested design. The variables will be discussed in section 4.1.

Related work with charge retention in FG transistors has focused on short-term charge loss or long-term TE. The "short" long-term mechanism caused by MIC has largely been left un-characterized. MIC leads to changes in the threshold voltage of an FG transistor, but more commonly manifests as a leakage current that effects the device's startup transients. When a device is powered on after being "off," there is a specific amount of time where the current fluctuates up to its steady-state value. The change in current looks like it is charging a capacitance. The presence of MIC as leakage currents lends to this view. As more MIC is present, a longer time constant manifests.

Mobile ionic charge has been shown to be directly related to exposure to the Na^+ and K^+ ions, as shown in [50]. As these elements are commonly found in water, this experiment aims to introduce the MIC via exposure to humidity, accelerating the effects of mobile ionic charge. To reiterate, the other "long" long-term effect of charge loss can be accelerated due to exposure to temperature. This experiment aims to characterize the "short" long-term effect

due to MIC without needing to wait years for an IC to experience exposure to the correct elements.

The null hypothesis for this experiment is: There is no way to reliably control, characterize, and account for the effects of MIC when operating in-the-field other than to have long startup delays in device operation. The hypothesis for this experiment is as follows: Accelerated effects of mobile ionic charge due to MIC can be introduced to an integrated FG transistor via exposure to humidity and removed via exposure to high heat. Characterization of the leakage currents caused by the MIC can be used for designing in-the-field applications of FPAAs without built-in startup delays.

4.1 Two-Factor Nested Design

There are multiple variables to consider for this experiment. Some of the variables originate from the experiment itself, the "lab" environment (which has been affected by COVID-19), and what originates from working with an IC. An IC is fabricated in a semiconductor material (specifically silicon in this case). The fabrication process is a series of unit processes that implement a circuit design onto the silicon wafer. The processes have been refined to high degrees of accuracy, but there still exist some inherent problems with IC fabrication.

General yield is the first uncontrollable variable to consider. Depending on process size, design, and the company that is fabricating the IC, yield can fluctuate quite a bit. Chips that return from fabrication must be tested to determine if the fabrication was successful. There are process design rules that can help with ensuring high yield from an IC fabrication. This experiment has been done with an IC that is already tested and in working order. But, this would need to be taken into consideration for large-scale deployment. Other fabrication variables will be discussed in section 4.3 when discussing threats to validity.

4.1.1 Independent Variables

There are three main independent variables to consider for this experiment. Each of these variables are controlled with respect to the steps that will be discussed in section 4.1.2. Due to COVID-19, some of these variables were not controlled to the degree of accuracy

that would be expected with a lab. Due to how the experiment was conducted, the levels present in the "current state of MIC in the FG transistor" could be considered their own independent variables. But due to how the data was collected, they are considered a "level" in that factor.

The breakdown of variables in this experiment is an unorthodox way of presenting the factors, but the experimenter felt it was the best way of presenting the data. No data was taken during the "bake" or "vaporize" portions of the test. All data was taken before and after the state changes. Due to this, the difference in MIC present in the FG transistor is seen as a factor being controlled by the experiment. As stated previously, this experiment is presented as a two factor nested design. For each of the three levels in factor A (current state of MIC in the FG transistor), all five levels of factor B will be tested (FG off time). The independent variables are as follows:

1. The current state of MIC in the FG transistor (The main factor, categorized by three levels)
    - This is considered factor A and has three levels:
        - Pre-bake
        - Post-bake and pre-vaporize
        - Post-vaporize
2. The time a FG transistor has been held in an "off" state before startup
    - This is considered factor B, will be nested under factor A, and has five levels:
        - FG off for 10 seconds
        - FG off for 100 seconds
        - FG off for 1000 seconds $\approx$ 17 minutes
        - FG off for 10000 seconds $\approx$ 2.8 hours
        - FG off for 100000 seconds $\approx$ 28 hours
3. The temperature of the room during startup testing

- An attempt to hold this variable constant will be made through the use of a ZTC point

The current state of MIC in the FG transistor and transistor "off" time will be the two factors being adjusted in this experiment. The other independent variable, room temperature, will be held fixed (discussed at length in section 4.3.2). Table 4.1 shows how factors A and B will be presented in a nested format. The three levels of factor A correspond to the three states of MIC in the FG transistor. The five levels of factor B correspond to the "off" time of the FG transistor, measured in seconds.

Table 4.1: Two Factor Nested Experiment Setup

Factor A														
Pre-bake					Post-bake and pre-vaporize					Post-vaporize				
Factor B					Factor B					Factor B				
10	100	1000	10000	100000	10	100	1000	10000	100000	10	100	1000	10000	100000

4.1.2 Dependent Variable

This experiment aims to affect one specific dependent variable. The dependent variable, as has been discussed throughout the paper is FG transistor startup time. The startup time refers to how long it takes a FG transistor to reach its operating current after being held "off" for a period of time. This startup time is the focus of the experiment. Without proper characterization, FPAAs are required to assume long startup transients regardless of device lifetime. This leads to an inherent latency in programming any new application into an FPAA that has been deployed in-the-field. By changing the independent variables presented in section 4.1.1, we can characterize the effects of exposure to MIC and program in the proper startup times for FG transistors, decreasing the inherent system latency.

To facilitate the experiment, the FG transistor will be exposed to both high temperatures and humidity, speeding up the removal and reintroduction of MIC. The following steps will be used to ensure an accurate measurement of the effects of MIC can be taken:

1. Program FG transistors up to a desired level, conducive to precise analog biasing

2. (Factor A, Level 1) Measure the FG transistor startup transient (for all 5 levels of Factor B)

3. Place FG transistor integrated circuits in a furnace and bake

   - Bake at 200°C for 2 hours

4. (Factor A, Level 2) Measure the FG transistor startup transient (for all 5 levels of Factor B)

5. Calculate the "removed" effect of MIC on the startup transient

6. Place FG transistor integrated circuits in humidifier

   - Humidify for 2 hours to reintroduce MIC as K^+ and Na^+

7. (Factor A, Level 3) Measure the FG transistor startup transients for different "off" times (for all 5 levels of Factor B)

8. Calculate the "reintroducted" effect of MIC on the startup transient

The purpose of this test is to verify previous work and see the effect on startup transients due to MIC. A more realistic end of life criterion can be developed for analog device operation if the startup transients are also accounted for. The steps above have shown a plan for characterization so that "long-term" charge loss is not incorrectly attributed to the effects of MIC. Each level of factor A took about 46 hours to complete. 31 hours were needed from the total "off" time required for all levels of factor B. 15 hours were needed for each test after the device had been turned on. This resulted in about 138 hours of data taking and over 32,000 data points.

4.2 Data Analysis

For this experiment, the same FPAA device was used for all stages of the testing process. The testing results have been broken down into three groups (section 4.2.1). From section 4.1.1, the three groups will be "pre-bake," "post-bake and pre-vaporize," and "post-vaporize"

states. The "pre-bake" scenario refers to FG transistors that have been in the operation for a set amount of time that they have collected enough MIC to effect startup transients. The "post-bake and pre-vaporize" scenario refers to when the FG IC has been baked at a high temperature to remove the MIC, reducing the latency in startup transients. And finally, the "post-vaporize" scenario refers to the FG IC after it has been exposed to high levels of humidity to reintroduce the MIC and by effect the latency in startup transients.

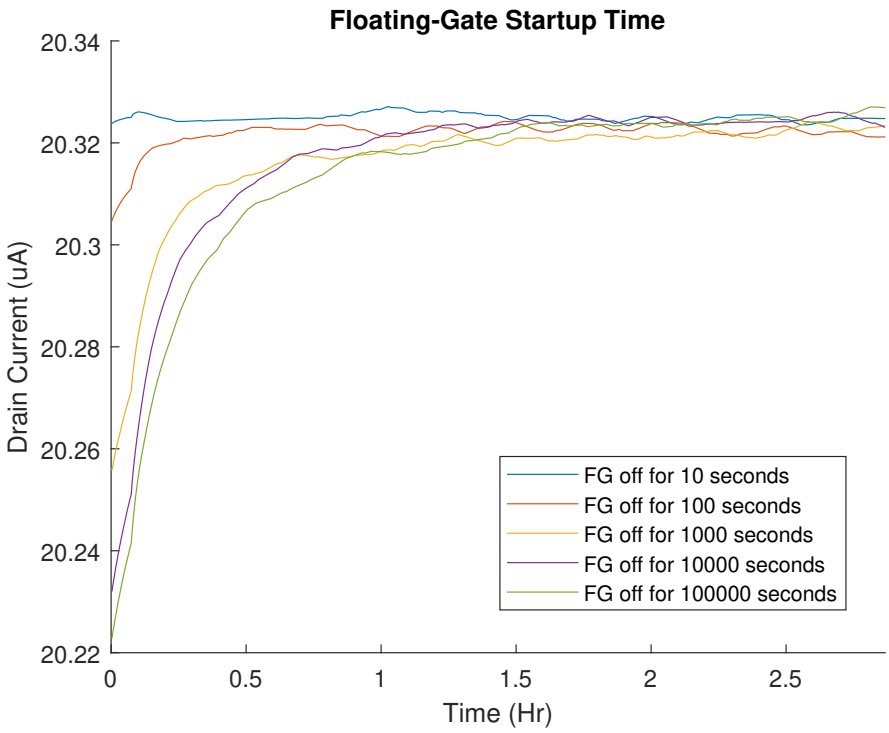


Figure 4.1: Measured data showing startup transients that are observed from an FG transistor directly after it is powered on after years of use.

For each stage of the test (the three levels of factor A), a plot will be shown for the FG IC's current output where the FGs were held in an "off" state for different periods of time (the five levels of factor B). The settling transients are then measured for each case where it can be observed that for each different period of "off" time, the FGs had a different startup transient. In fact, the longer an FG was held off, the longer it took for an FG to return to the desired bias level. Section 4.2.1 covers the three levels of factor A. For each level of factor A, all five levels of factor B will be tested. The graphs will be presented for each case and the results will be discussed in section 4.3.

4.2.1 Levels of MIC in FG Transistors

The pre-bake test was done on an FPAA that had been in use for over 4 years (the same FPAA that will be used for the entirety of the experiment). This FPAA was chosen as it had been adequately exposed to elements and it was hypothesized that it had contracted MIC, affecting FG transistor startup times. This sub-hypothesis of the experiment will be proved true or false depending on the results of the following sections.

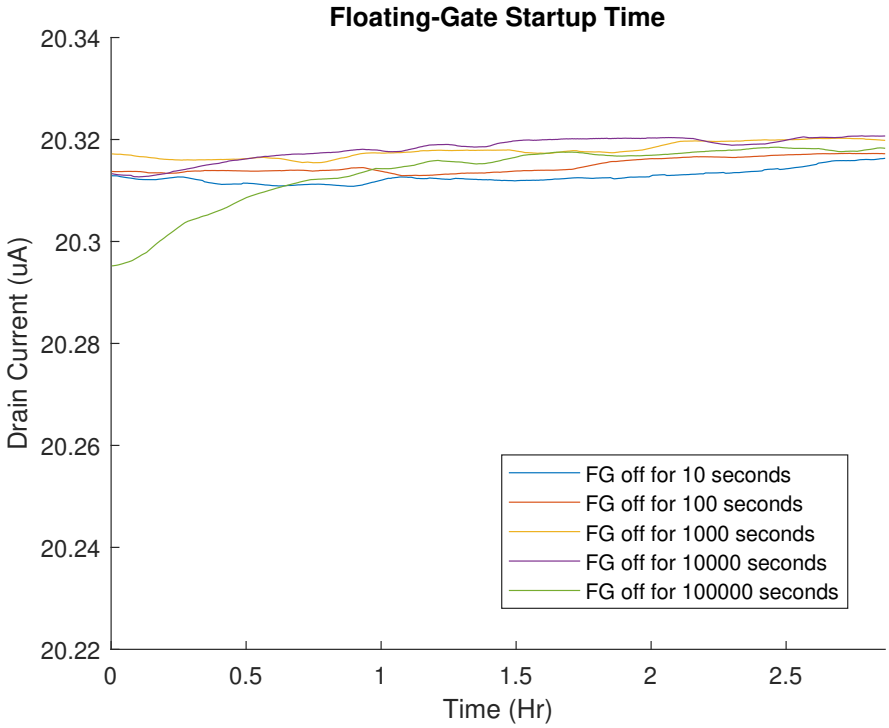


Figure 4.2: Measured data showing startup transients that are observed from an FG transistor directly after it has been baked at 200°C for 2 hours.

Figure 4.1 shows a plot of FG transistor startup times. For each case, the same FG transistor was used as the device under test (DUT). The DUT was held in an "off" state for a specific amount of time and then powered on. As soon as power was connected to the device, the current flowing through the transistor was measured every 5 seconds for 3 hours. The test was done for a bias level of around 20.3µA. The biasing level was chosen due to the transistor's zero temperature coefficient (ZTC) point. The ZTC point is a bias level for a transistor that results in temperature independence. The closer a transistor bias is to the

ZTC point, the less effect temperature will have on its performance. The biasing level of 20.3µA was chosen to negate temperature effects as much as possible.

The test shown in figure 4.1 shows a max deviation of around 100 nA. As stated previously, this work aims to look at FG reliability with respect to precise analog biasing. For analog devices, it is common to see biasing values on the tens of micro-amps down to the tens of nano-amps scale. The change shown at 20.3µA may be insignificant, but down at 100nA would easily cause a sensitive analog circuit to fall out of the correct biasing range. After the DUT was measured for all levels of factor B, the experiment moved to level two of factor A. To move the experiment to level two (of factor A), the device was baked at 200°C for 2 hours. This was done to remove the MIC that had built up in the FPAA over four years of typical use. Level two of factor A is a scenario where the minimal amount of MIC should be present in the FG transistor.

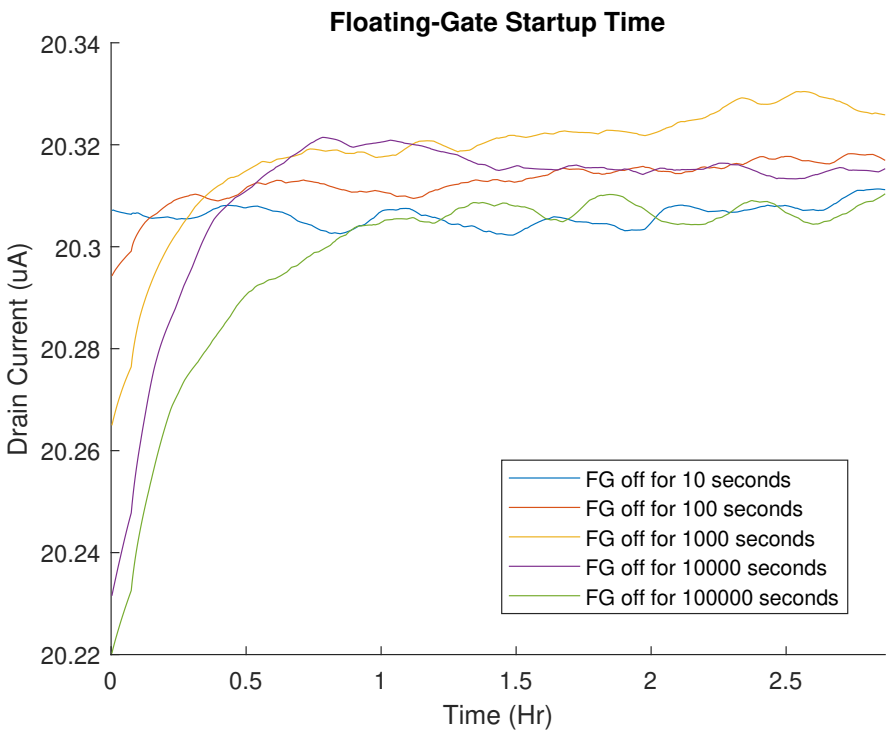


Figure 4.3: Measured data showing startup transients that are observed from an FG transistor directly after it has been exposed to humidity for 2 hours to reintroduce MIC as K^+ and Na^+.

Continuing with the proposed steps, the FPAA was exposed to humidity for 2 hours to

move the experiment into level three (of factor A). After being subjected to humidity, it was hypothesized that the presence of Na^+ and K^+ in water would reintroduce MIC into the FG transistors. If that were the case, then the startup transients would go back to a level similar to the "pre-bake" scenario. Figure 4.3 shows that from visual inspection, the startup transients seem to have reverted back to levels seen "pre-bake." If the visual inspection is correct, it would show that the MIC had been reintroduced into the FG transistors.

4.3 Results and Threats to Validity

As stated previously, from visual inspection, we can observe that the startup transients were affected as predicted by the change in MIC. Continuing to analyze the data further, we are able to measure the time constants of each startup transient. To calculate the time constant, an average of the last 500 data points is taken for each run. This average is seen as the "steady-state" of the run. The time constant is how long it takes the FG transistor to turn on to at least two-thirds of the "steady-state."

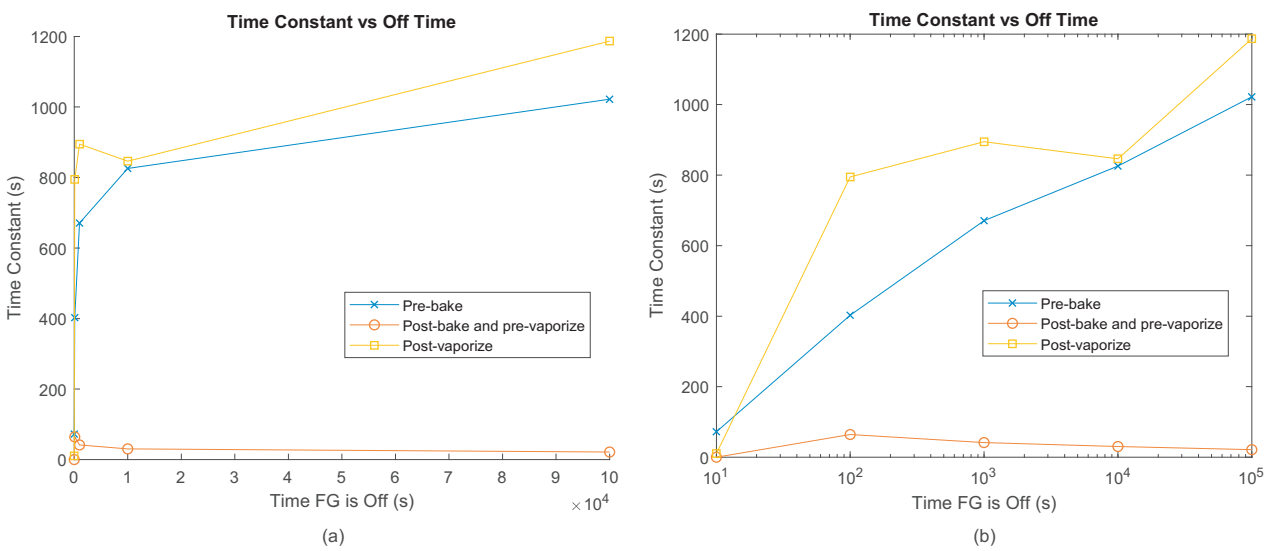


Figure 4.4: Resulting time constant for all three levels of factor A and all five levels of factor B on (a) linear scale and (b) semi-log scale.

Figure 4.4 shows the time constants for each level of both factor A and B for (a) a linear scale and (b) a semi-log scale. The three different lines refer to the three levels in factor A

(pre-bake, post-bake and pre-vaporize, and post-vaporize). The five data points on each line refer to the five levels of the nested factor B ("off" time of the FG transistor).

Table 4.2: The Resulting Time Constants for Each Nested Level of Factors A and B

	Factor A														
	Pre-bake					Post-bake and pre-vaporize					Post-vaporize				
	Factor B					Factor B					Factor B				
FG "off" Time (s)	10	100	1000	10000	100000	10	100	1000	10000	100000	10	100	1000	10000	100000
Average (steady-state) (µA)	20.324	20.317	20.300	20.294	20.290	20.313	20.315	20.318	20.317	20.310	20.305	20.309	20.305	20.287	20.278
Time-constant (s)	72.25	402.5	670.9	825.7	1022	0.103	64.2	41.39	30.45	21.26	10.32	794.7	894.7	846.3	1187

Table 4.2 shows the resulting time constants for all nested levels of each factor. This table shows empirically that the time constants are dramatically improved for the case where MIC have been removed (post-bake and pre-vaporize). Deviations in these values will be discussed further in section 4.4.

The threats to validity originate from two different categories, namely the fabrication of the IC and the testing setup. Each of these sources contribute different aspects of threats to validity and will be discussed within the following sections. The testing setup involves threats that are introduced by the environment and by the experimenter. Errors in setup or procedure can lead to erroneous results.

IC design is done in two steps. The first step is schematic design and simulation. This is done via SPICE netlisting and simulation to give designers control on how their circuits will operate. Once a circuit has been designed in a schematic view and is simulating correctly, the designer moves to layout. Layout refers to the physical dimensions of the circuit that will be fabricated into the silicon wafer. Different layout practices are implemented to ensure the finalized layout will match the schematic in both design and performance (reducing parasitic capacitances, device mismatch, etc.).

For the purposes of the execution of the experiment and handling of independent variables, we have assumed that all layout practices were done to minimize the negative effects that manifest in layout design. But, the different problems that arise from layout design and general fabrication still need to be addressed.

4.3.1 Construct

FG transistors have parasitic capacitances that lead to startup transients without the effect of MIC. A singular FG transistor will have a set amount of parasitic capacitance. This parasitic capacitance manifests as a leakage current, causing an effect on the startup transient. This effect should be minimal when looking at the effects of MIC being added or removed to the FG transistor. However, we are considering FG transistors within an FPAA. In this case, we are not just dealing with one FG transistor, but instead hundreds of transistors within an array.

When we are measuring the startup transients, we are essentially measuring the leakage current of the FG transistor. But, when the device is connected to hundreds of other devices (even though they are powered off), there exist leakage currents through the directly connected drain and also the switches used to select the currently working device. More work would need to be done to ensure the differentiation of leakage current being observed between the MIC and the array of FG transistors. Due to the magnitude of change in leakage current, we can extrapolate that the leakage current that exists after MIC have been "baked" off is due to the FG array, but some of that might still be due to the presence of unremoved MIC.

4.3.2 Internal

Continuing on the subject of FG transitor arrays within FPAAs, there exists an unknown factor when programming array-based FGs. When you program one specific FG within an array, specifically with the process of injection to add charge, you also slightly program all the other FGs within the array. As you add charge to one FG, you are also adding a small amount of charge to the other FGs. This can be negated by applying a small tunneling voltage to all of the FG transistors. While this slows down the injection process, this also helps reduce the amount of charge being added to the other FG transistors.

The issue is that without cycling through every single FG transistor and measuring its IV relationship with a gate-sweep, we can not be sure they were not slightly injected (or tunneled, if the tunneling voltage was too high). This process would add immense latency

after every programming cycle and burn more power, making it unrealistic for in-the-field scenarios. This results in an unknown amount of charge being added or removed to all the unused FG transistors within the array. This charge would modify the total leakage current seen from any individual FG transistor. This experiment ignores this small change in leakage current, but would certainly effect the absolute accuracy of the results. Characterization of each FG transistor after programming would allow for it to be accounted for and increase the resolution accuracy of the results.

The main source of threat to internal validity comes from the testing setup that was used for this experiment. This experiment was conducted during the COVID-19 pandemic. The original plan was to house the experiment within a temperature chamber, available in the CES Lab at WVU. The temperature chamber could have been used to ensure the testing occurred at a specific temperature. This is critical because FG transistors are highly dependent on fluctuations in temperature. There exist compensation techniques to reduce the effect of temperature on a FG transistor, but there is no way to completely negate its effect. The temperature chamber would have ensured no changes in temperature, removing this variable and its effects from the results.

However, when the stay-at-home order was issued, the experiment was moved to my own home office. I have access to electronics testing equipment, but the temperature chamber was not able to be moved from CES Lab. To try and negate this, two actions were taken. The first was general temperature regulation from my home central heating system. I have a digital thermostat that I was able to set an upper and lower bound to keep temperatures even. But, this range is much less accurate than a laboratory temperature chamber, and given that the sensor for the thermostat was not in the office, this could only be so accurate. The second action taken was to operate the FG transistor at what is referred to as the zero temperature coefficient (ZTC) point.

The ZTC point of a transistor (and specifically a FG transistor) is a biasing point that completely negates any temperature dependence. The issue with the ZTC point is just that, it is simply a point. It is not a region of operation or a range of values, but just one specific bias point. Due to the variable nature of reality, hitting that exact point is realistically impossible. But, the region above and below the ZTC point allow for extremely

low proportional or inversely proportional dependence on temperature. So, by operating closely to the ZTC point we can negate the majority of temperature effects. However, by limiting our experiment to only this biasing range, we have limited the conclusions that can be drawn from the results.

4.3.3 External

The results discussed in section 4.2 show that MIC can be added or removed by way of "baking" and "humidifying" the FG transistor-based FPAA IC. However, while the results can be applied directly to the FPAA designed in CES Lab at WVU, applying these results to other FPAAs becomes challenging. As already discussed in the previous threats to validity sections, the composition of the FG transistor array is an important factor in what leakage currents (and therefore startup transients) can be observed. While the method of characterizing the effects of MIC could be reproduced for alternative FPAAs, the results from this experiment could not be directly applied to another system.

FG transistor array size and composition is not the only difference in FPAAs that need to be addressed. One important factor that needs to be addressed is the chosen process node. In IC fabrication, the different available processes are typically described by the number of layers of polysilicon and smallest possible resolution. This is important when dealing with FG transistors for multiple reasons. The first is that to design a reliable FG transistor, two polysilicon layers are required. If a fabrication process only provides one polysilicon layer, the control-gate will need to be constructed with a non-linear MOSCAP (MOSFET connected as a capacitor) which will make accurate programming more difficult. The other factor of resolution comes into play when dealing with oxide thickness. Smaller process sizes usually have smaller available oxide thicknesses.

Both oxide thickness and the access to two polysilicon layers are the determining factor on whether a reliable FG transistor can be built. The FPAA designed in CES Lab at WVU was fabricated in a 0.35µm process. This process has access to two polysilicon layers and thick oxide devices. When considering how the results are applicable to other FPAAs and other styles of FG arrays, it must also be noted that these results only apply to 0.35µm

processes.

4.4 Conclusion

This experiment has shown that MIC can be introduced and removed from a FG transistor via high temperatures and exposure to elevated levels of humidity. The results detailed in section 4.2 show very clearly that the startup transients due to MIC could be removed and then reintroduced. The hypothesis that the long-term effects of MIC could be accelerated for characterization in FPAAs was validated. However, this experiment was only the first step. This experiment showed that MIC could be controlled and that its effects were as predicted by earlier studies.

Due to the extensive nature of the testing, only one full run was done for all levels of factor A. To continue with this work, the experiment should be redone multiple times with no changes to other independent variables. This would allow all the data points in the nested levels of factor B to be averaged. Some error in the data is assumed, especially when only one run has taken place. By increasing the number of runs, the characterization data could be more valuable to FPAA designers working in a 0.35µm process.

Beyond the improvement stated above, getting this experiment back into an actual lab setting would allow for more precise control of other independent variables. Access to a temperature chamber would enable the FG transistor to be biased at other current levels (instead of just the ZTC point). This would increase the scope of characterization that could be concluded.

Chapter 5

The RAMP 1.1

A new version of the Reconfigurable Analog and Mixed-Signal Platform (RAMP) [39] field-programmable analog array has been designed and fabricated. This version, being an evolution from the first version of the RAMP is named the RAMP 1.1. In this second iteration, improvements were made to the programming infrastructure necessary for FG transistor operation, as well as additions and improvements to its computational analog blocks (CABs). The CABs on the RAMP 1.1 hold all the components that are used to make up the pool of synthesizable circuits. In addition to modifications to preexisting stages, an extra stage was added.

The RAMP's CABs include specific stages for spectral-analysis, transconductors, sensor interfacing, mixed-signal-processing, digital computation, and general transistors. Each of the stages are replicated to create eight independently-reconfigurable channels. The RAMP 1.1 (hereafter referred to as just the RAMP) has an added ninth analog stage and increased die space for housing the improved spectral analysis stage. A new type of filter has been implemented in the spectral analysis stage, aiming to improve filter response when used in low and high pass situations.

The spectral-analysis CABs contain bandpass filters (BPFs), peak detectors, adaptive-time-constant filters, and operational transconductance amplifiers (OTAs). The transconductance stages contain current sources, current sinks, connections to FGs, current mirrors and multipliers. There are device stages which include general purpose MOSFETs, BJTs, differential pairs, capacitors, and resistors. The mixed-signal CABs contain comparators,

sample and holds, pulse generators, timers, Schmitt triggers, and starved inverters. The digital stages contain LUTs and JK flip-flops.

Signal routing in the RAMP is achieved via two programmable switch domains: "switch boxes" and "connection boxes" (as shown in Figure 5.1). These programmable switches are implemented using T-gates (nMOS and pMOS transistors connected in parallel to create a rail-to-rail switch). The "connection boxes" provide a crossbar switch matrix comprised of T-gates for intra-CAB routing. The "switch boxes" utilize the T-gates in a 4-way "diamond-switch," allowing for inter-CAB connections. A desired switch configuration can be loaded into the RAMP using an on-chip serial-peripheral interface (SPI). 20,380 switches in total are included within the RAMP [39].

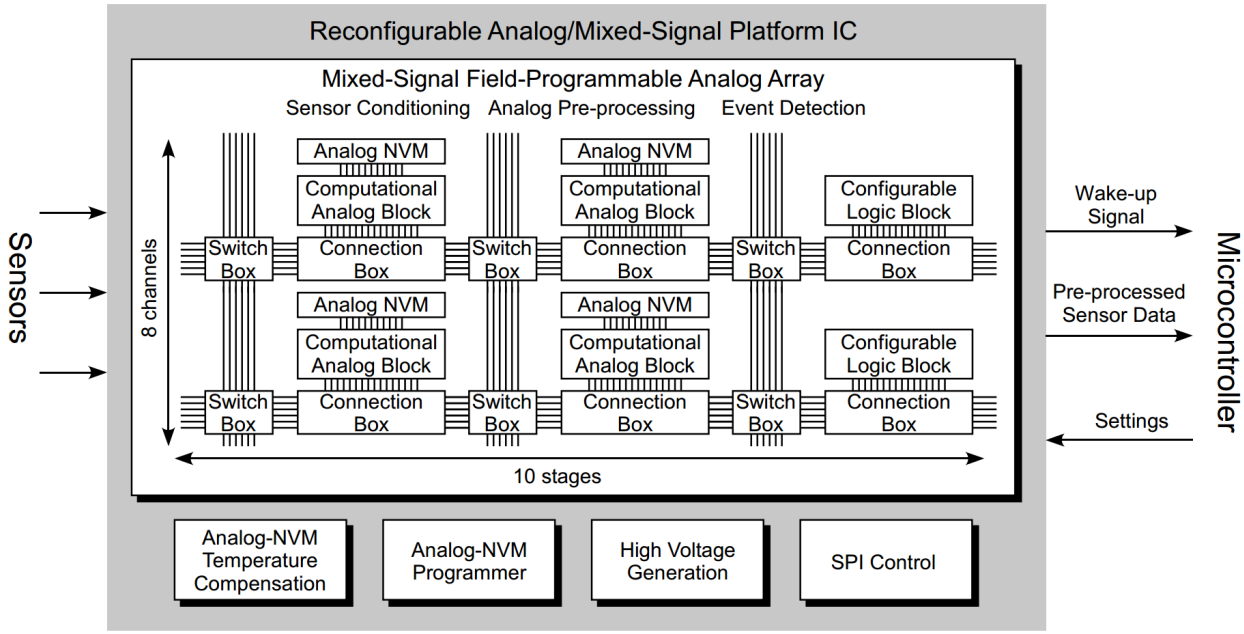


Figure 5.1: RAMP block diagram.

CABs are connected to tunable FG current biases, allowing for the performance of analog circuits with programmable parameters (eg. corner frequencies and gain) to be modified to fit a user-specified design constraint. As discussed in section 2.3, two different processes are used to apply or remove charge from an FG. Hot-carrier injection is used to add charge to the FG, and Fowler-Nordheim tunneling is used to remove charge from the FG [32]. Due to the high voltages required for tunneling, the RAMP only utilizes tunneling for global erasure

of FGs.

To perform fast, linear injection on the RAMP, FGs are individually placed into a programming structure. This structure is a continuous-time, OTA-based, negative feedback controller which compares the voltage on the control gate of the FG being injected to a target voltage. The injection target voltage corresponds to a characterized bias current that is desired from the FG [32]. After programming, FGs are connected to CABs to bias analog circuits. The programming circuits are now integrated on the same die as the RAMP (1.1). The RAMP 1.0 required programming circuitry to be housed on its printed circuit board (PCB).

Configuration of the RAMP is expedited by a few layers of abstraction. The required programming process is currently a work in progress. The functionality of the RAMP is limited to its capability to be programmed and re-programmed with minimal effort. The more streamlined the process is, the more likely the RAMP will be adopted by electronics designers as an alternative to front end signal processing. The following is a plan for the RAMP 1.1 infrastructure, based on the RAMP 1.0.

To synthesize a circuit on the RAMP, a user must first provide the RAMP software with the corresponding netlist. The RAMP software will then estimate injection target voltages for the necessary FGs and use simulated annealing [51] to find the switch activation pattern resulting in the shortest mean path among the components specified in the netlist. FG and switch domain configurations are then compressed and sent to the MCU on the RAMP PCB. The MCU then communicates with the RAMP through its SPI to place each required FG into the on-chip programming structure and inject them to their respective targets. Switch domains are configured after all FG programming is complete [39]. The RAMP can be reconfigured in-the-field, but any user-specified configurations will not be operational for the duration of programming. Once the relevant FGs and switches have been programmed, the RAMP will be ready to perform user-specified operations.

5.1 The Unique CAB

The new, ninth stage added to the RAMP 1.1, does not follow the convention of the preexisting CABs. Instead of the stage having eight repeating channels, each of the channels includes a different circuit, hence being named the "unique CAB." The unique CAB is used to implement larger scale circuits that are found in the other CABs. The unique CAB allows for circuits like a voltage-controlled oscillating-based ADC (VCO-ADC) (to be described in Chapter 7) to be synthesized from larger CAB components without taking over all the routing space on the RAMP. For a circuit to be synthesized on the RAMP system, it requires a series of connections to be made through the inter and intra CAB routing matrix. This is a finite system, and as such, reducing the amount of connections needed for a given component opens the door for more complex designs. By including a circuit directly in the unique CAB, it can be synthesized while still leaving room for addition synthesis.

5.1.1 Spectral Analysis

The spectral analysis stage has been increased in die space, allowing for biquadratic (Biquad) filters to be implemented. The Biquad filter is an upgrade and replacement to the C4 Bandpass filter [52] previously housed in the spectral analysis stage. The C4 filter operated as a bandpass filter, with operation being based on two bias currents, which set the upper and lower cutoff frequencies. If the lower cutoff frequency bias was set by a low enough current, the C4 could perform similarly to a low-pass filter. In contrast, if the upper cutoff frequency was set with a high enough current, the C4 could perform akin to a high-pass filter. The C4 requires very accurate capacitor matching to achieve uniform gain for similar quality factors while in a bandpass configuration. Also, in practical application, the C4 does not see uniform gain in its low-pass and high-pass configurations. Utilization of the Biquad filter aims to remove the issues with capacitor layout mismatch while also improving on the low-pass and high-pass performance.

Two versions of the Biquad filter (Figure 5.2) have been implemented on the RAMP 1.1. The two versions vary with capacitor sizing. The Biquad with smaller capacitor sizes was used to replace the C4 filter. The smaller capacitor sizing was needed to fit with the

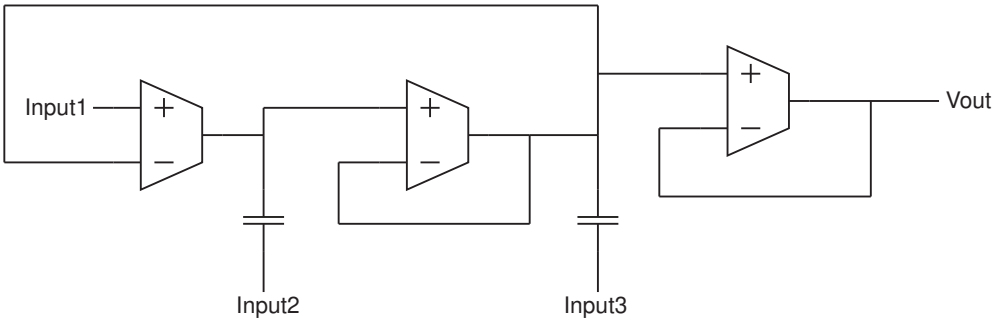


Figure 5.2: Schematic of Biquad Filter.

other components already in the spectral analysis stage (even with an increased stage size). Utilizing the full space of a CAB, a Biquad with larger capacitor sizes was added to the unique CAB. Without needing to share die space with other components, the unique CAB Biquad filter was able to utilize larger capacitor sizes, which we predict will increase the accuracy of cutoff frequencies.

The Biquad filter operation is based on more than just one input. There are three points for input into the filter. By choosing a specific input, and setting the other two inputs to an analog ground, the Biquad is able to operate as either a low-pass, high-pass, or bandpass filter. With each of the three inputs connected to the CAB lines, when the circuit is connected on the RAMP, it is forced into a specific configuration, only allowing for the desired filter operation. By removing the need to use extremely low (subject to noise) or extremely high (power hungry) bias currents, the Biquad aims to fill the space left by the C4 filter with metrics more promising to low-power ASP tasks.

5.1.2 Sensor Interfacing

The RAMP 1.1 has been designed to have better sensor interfacing capabilities over the 1.0. The most practical way to increase interfacing capabilities was what led to adding the unique CAB. Certain comprehensive circuits bring processing power without the need for synthesis. A reduction in synthesis routing is not the only benefit of pre-connected circuits. While a circuit like the Wheatstone bridge could be synthesized from the existing components on the RAMP 1.0, when connecting via the CAB lines, additional parasitic capacitances and

resistances affect its sensitive performance.

While parasitics can never be fully avoided, allowing the circuit to be connected fully within one CAB helps decrease such negative effects. Due to this consideration, one of the circuits added to the unique CAB was a pre-built Wheatstone bridge with its two inputs connected directly to the pads on the IC, for minimizing parasitics and providing accurate resistance-to-voltage measurements.

5.1.3 Oscillators

To increase the range of applications available to the RAMP 1.1, two different clock generators were added with the inclusion of the unique CAB. A Ring Oscillator (RO) was added to perform fifty percent duty cycles and an Integrate and Fire Neuron (IFN) was added for a variable pulse period generator.

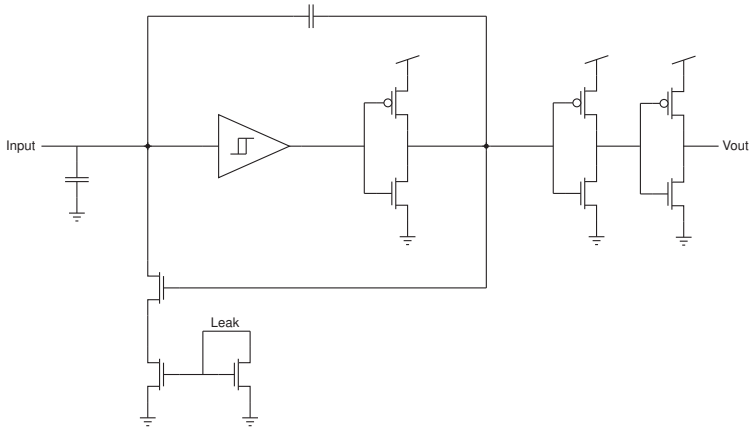


Figure 5.3: Schematic of Integrate and Fire Neuron.

The IFN (shown in Figure 5.4) mimics the behavior of a biological neuron [53]. To begin firing, a current is dumped on to the input pin in a manner similar to how a neuron is contacted within the brain. A leakage current is used to control the pulse width of the clock cycle and will be generated from the floating-gate transistors. A Schmitt trigger is utilized to set up hysteresis within the neuron, allowing for a duty cycle controlled by the leakage current [54]. A simulated output of the neuron is shown in Figure 5.4.

The RO added to the unique CAB is the same topology of VCO as described in section 7 for the VCO-ADC and can be seen in Figure 7.14. The RO's current starved inverters are

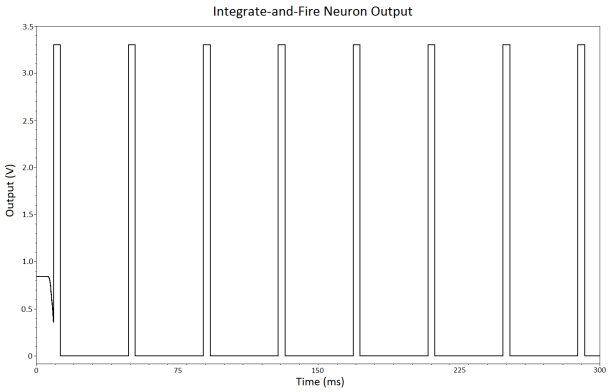


Figure 5.4: Simulated Output of Integrate-and-Fire Neuron.

connected directly to an FG transistor for input. The FG transistor, as a part of the RAMP 1.1's programing infrastructure is used to convert a voltage value to a current, which sets the frequency of the RO. The linearization techniques described in section 7 can be leveraged to improve the VCOs linear range. The RO was added so that the VCO-ADC and other systems that require local oscillators can be synthesized on the RAMP 1.1.

5.2 On-Chip Floating-Gate Programming

The RAMP 1.0 required off-chip circuits for FG transistor programming. A continuous time programmer was integrated on the RAMP 1.0, but the charge pumps required for injection and tunneling were not present, requiring off-chip circuits housed on the PCB. For the RAMP 1.1, the charge pumps were integrated directly on the die, allowing for less over-head on the PCB and a smaller package.

The programming process involves moving FG transistors into the programming structure and then back out to connect to the necessary circuits. The following will describe FG programming for the RAMP 1.1. FG transistors are placed into the continuous time programming structure shown in Figure 5.5. The multiplexer S_1 is used to connect the control gate of the FG transistor to the programmer circuit for injection or to the temperature compensation circuit during normal operation. The multiplexer S_2 is used to connect the source of the FG transistor to ground when it is not used, V_{dd} when it is connected to a

circuit, and the programmer for injection. Multiplexer S_3 is used to connect the drain of the FG transistor to V_{dd} when it is not used, ground for injection, and to the circuit during normal operation.

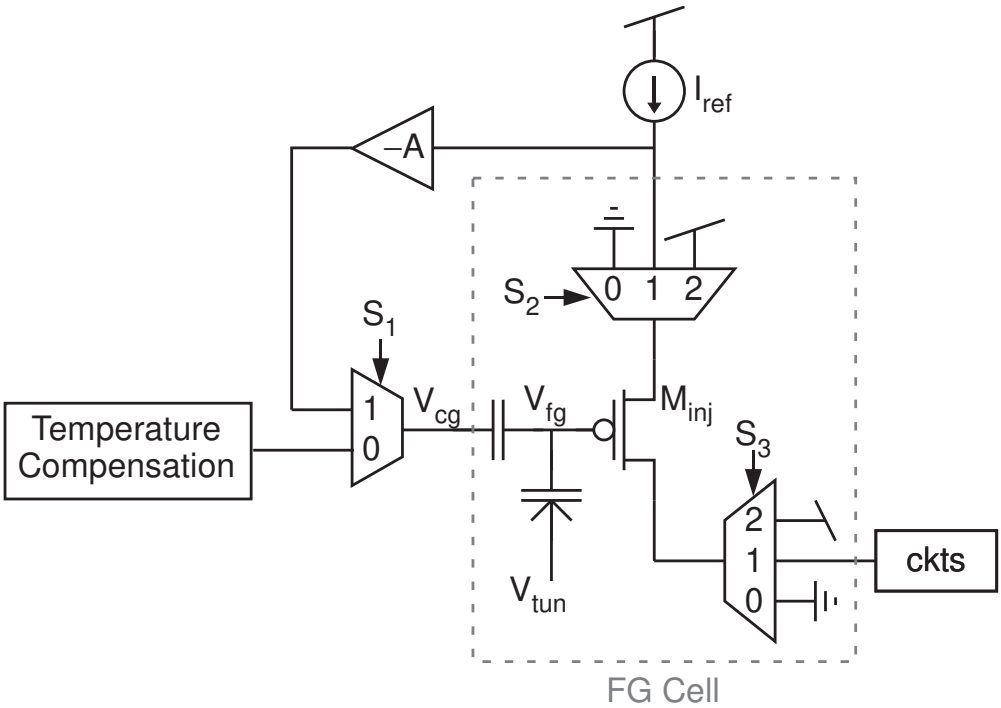


Figure 5.5: The programming circuit utilized by both the RAMP 1.0 and 1.1.

The two mechanisms for programming FG transistors discussed in Chapter 2.3 require larger voltages than the normal supply rails for a given process. The need for these high voltages requires the use of charge pumps. For the RAMP 1.1, fabricated in a 0.35µm process, tunneling requires a voltage applied to V_{tun} of at least 10V (but closer to 14V is used to speed up the process). 4.5V is required to initiate injection (but 6.5V is used to speed up the process). Charge pumps are required to produce these voltages on-chip. By integrating the charge pumps directly on the RAMP 1.1, required PCB space is decreased. The power consumption of the RAMP naturally increases by including the charge pumps, but with an integrated, non off-the-shelf circuit, the power consumption of the entire system is decreased.

5.3 A Modular MCU Printed Circuit Board

A PCB has been designed and fabricated for the RAMP 1.1. The new PCB was able to save space by removing ICs that were previously used to generate voltages for tunneling and injection. The PCB layout is shown in figure 5.6.

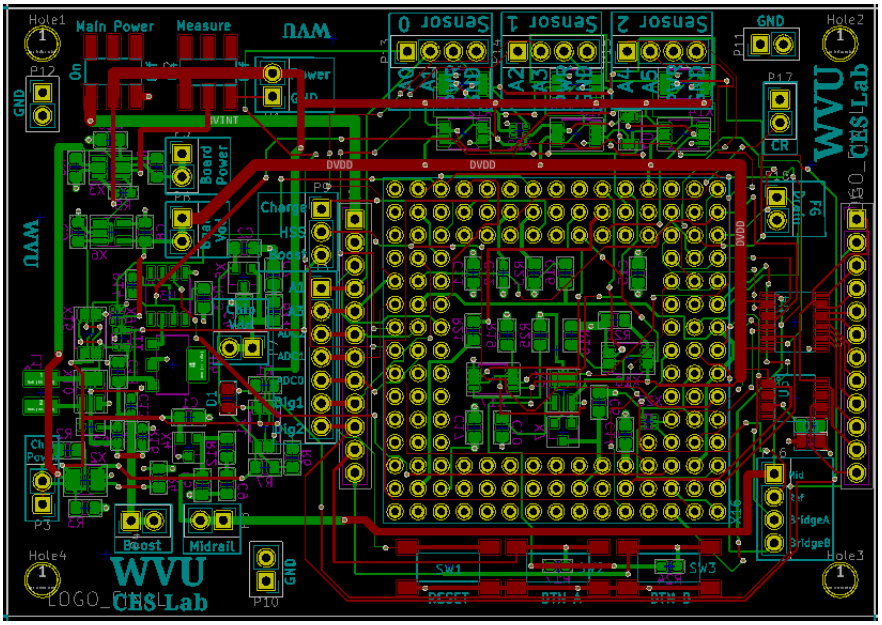


Figure 5.6: The RAMP 1.1 printed circuit board layout.

One of the improvements that comes with the new PCB is the ability to use any MCU. On the PCB designed for the RAMP 1.0, the MCU was soldered directly on the bottom of the board. But, with updates and changes being made to MCU units with current state-of-the-art speeds, the ability to upgrade the MCU without needing to fully re-fabricate and repopulate the PCB is a large boon. Currently, a small MCU PCB or "surfboard" is used for the MCU to be connected to the RAMP.

Surfboards have been designed for the current version of the Particle Xenon [55], Arduino Nano [56], and Arduino Nano IoT 33 [57]. While it may seem unwieldy that a second PCB is needed for attaching a new MCU, the benefits of allowing for upgrades to the MCU outweigh the added work. To illustrate, the Particle Xenon was chosen due to its similar benefits to the previously used PanStamp. Both PanStamp and Particle Xenon included radio directly on the package. However, like the PanStamp, the Particle Xenon has now

been discontinued. While this does not completely negate use of these MCUs, it is still desirable to use a state-of-the-art package, with all the benefits of regular updates to device and interfacing software.

The surfboard for the Arduino Nano requires the addition of a radio transceiver. The Arduino Nano does not have a built in radio, but was chosen as a stepping stone for the plethora of open source projects that have been developed for the Arduino platform. To enable radio functionality, allowing for remote programming, the Digi Xbee3 radio transceiver [58] was added to the MCU surfboard. This simple radio enables programming information to be sent from a base station.

The most recent version of the MCU surfboard houses the Arduino Nano IoT 33 (NIoT33). This MCU comes with built-in low-energy Bluetooth (BLE) communications. BLE allows for low-power wireless communication. This is leveraged to send connections lists from a base-station to the RAMP device. The PCB layout for the NIoT33 is shown in figure 5.7.

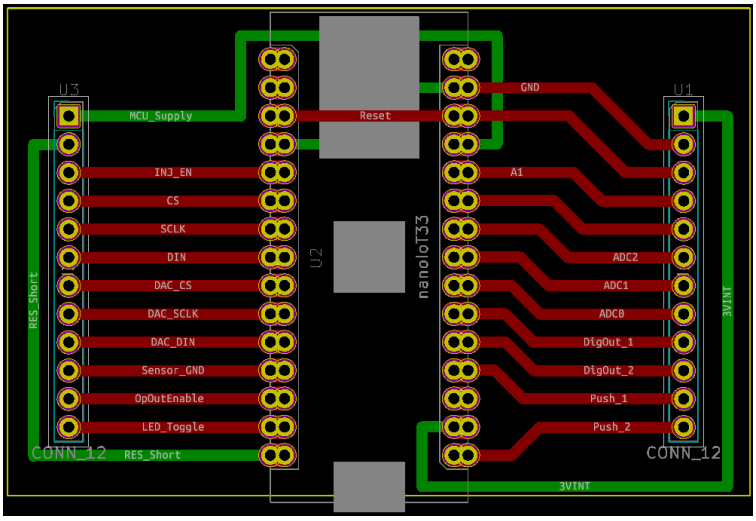


Figure 5.7: PCB layout for the Arduino Nano IoT 33 with BLE (and castelated edges for a slimmer package).

The MCU surfboard connects to the bottom of the RAMP board and can be seen attached in Figure 5.8. The complete system allows for connections lists to be sent to the microcontroller platform via BLE communications. The MCU interprets the packets being sent via BLE and communicates with the RAMP via a serial peripheral interface.

Figure 5.8: The RAMP 1.1 printed circuit board with attached modular MCU surfboard (below).

5.4 Wireless Programming with Low-Energy Bluetooth

The RAMP switch matrix, detailed previously in this Chapter and shown in figure 5.1, allows for the reconfigurable nature of the FPAA system. Programming the RAMP starts with a high-level netlist. The netlist is a representation of the circuits that will be connected within the RAMP. The following example will be used to demonstrate programming a desired circuit onto the RAMP. This example implements a peak-detector. The peak-detector tracks rising and falling voltage signals differently based on the supplied bias currents. The netlist is as follows:

```
include boardDefsNewBoard.net
TempComp 1
begin PK In Out Channel HiBias LoBias
    PkDe_S1C<Channel> <In> <Out>
    FG_S1C<Channel> PkDe_Atk Itar=<HiBias>
    FG_S1C<Channel> PkDe_Dec Itar=<LoBias>
end
PK Vin Vout 0 8e-08 1.5e-08
A5 Vin
ADC0 Vout
```

This netlist makes use of “subcircuit” notation. The “begin” and “end” statements wrap

Table 5.1: Connections list for implementing the peak detector on the RAMP 1.1.

	Switch Matrix Bits	Switch Address	CAB Channel	CAB Stage	Reset	SB or CB	LUT	Circuit	VO
Global Reset	111111111111	XXXX	XXX	XXXX	X	X	X	X	X
Enable Temperature Compensation	XXXXXXXXX0XX	XXXX	XXX	1111	0	X	X	X	X
Connect output of peak detector to Connection Block SB0	111110111111	1000	000	1000	0	1	0	0	0
Connect input of peak detector to Connection Block SB1	111111011111	0000	000	1000	0	1	0	0	0
Connect output to RAMP pin ADC0	011111111111	1000	000	1000	0	0	0	0	0
Connect input to RAMP pin A5	110111111111	0000	000	1000	0	0	0	0	0
Connect output from channel 0 to 1	011111111111	0100	000	1000	0	0	0	0	0
Connect output from channel 1 to 0 and 2	011111111111	0100	100	1000	0	0	0	0	0
Connect output from channel 2 to 1 and 3	011111111111	0100	010	1000	0	0	0	0	0
Connect output from channel 3 to 2 and 4	011111111111	0100	110	1000	0	0	0	0	0
Connect output from channel 4 to 3 and 5	011111111111	0100	001	1000	0	0	0	0	0
Connect output from channel 5 to 4 and 6	011111111111	0100	101	1000	0	0	0	0	0
Connect output from channel 6 to 5 and 7	011111111111	0100	011	1000	0	0	0	0	0
Set the FG to program for the DEC peak detector bias (falling)	000010000000	0000	000	1000	0	0	1	0	1
Connect the FG to the circuit	000010000000	0000	000	1000	0	0	0	1	0
Set the FG to program for the ATK peak detector bias (rising)	000000000010	0000	000	1000	0	0	1	0	1
Connect the FG to the circuit	000000000010	0000	000	1000	0	0	0	1	0

the subciruit "PK." When PK is then called on the subsequent line, it passes the parameters into the subcircuit. The nets "Vin" and "Vout" are the input and output of the circuit respectively. The pin "A5," a general analog pin is used as the input and is connected to Vin. The pin "ADC0," which is connected to an ADC on the NIoT33 is used for Vout. This pin is chosen so the output of the circuit can be read directly by the NIoT33 MCU.

Once a netlist has been written for programming into the RAMP, a connections list is generated. The connections list is used to determine which switches will be set within the conneciton and switch matrix to synthesize the desired circuit. The connections list is a series of binary strings that can be passed into the RAMP via an SPI. The connections list holds commands for resetting the RAMP setup (including tunneling FG transistors), enabling the correct switches, and programming the needed FG transistors for biasing. Table 5.1 shows how the connections list appears and what each section of the binary string corresponds to.

The RAMP 1.1 connects inputs on the "right" side of the chip next to channel 7 and

outputs on the "left" side of the chip next to channel 0. The peak detector being used is housed in stage 1 channel 0. Due to this, the output can be piped directly to pin ADC0. The input, however, must connect from channel 0 all the way to channel 7 to allow for connection to A5.

To facilitate wireless programming, the NIoT33's BLE communications are used to pass information between a base-station and the wireless node. The base-station is an NIoT33 plugged into the PC. The wireless node is an NIoT33 connected directly to the RAMP 1.1 via the MCU surfboard. The programs installed on each NIoT33 can be found in Appendix C.

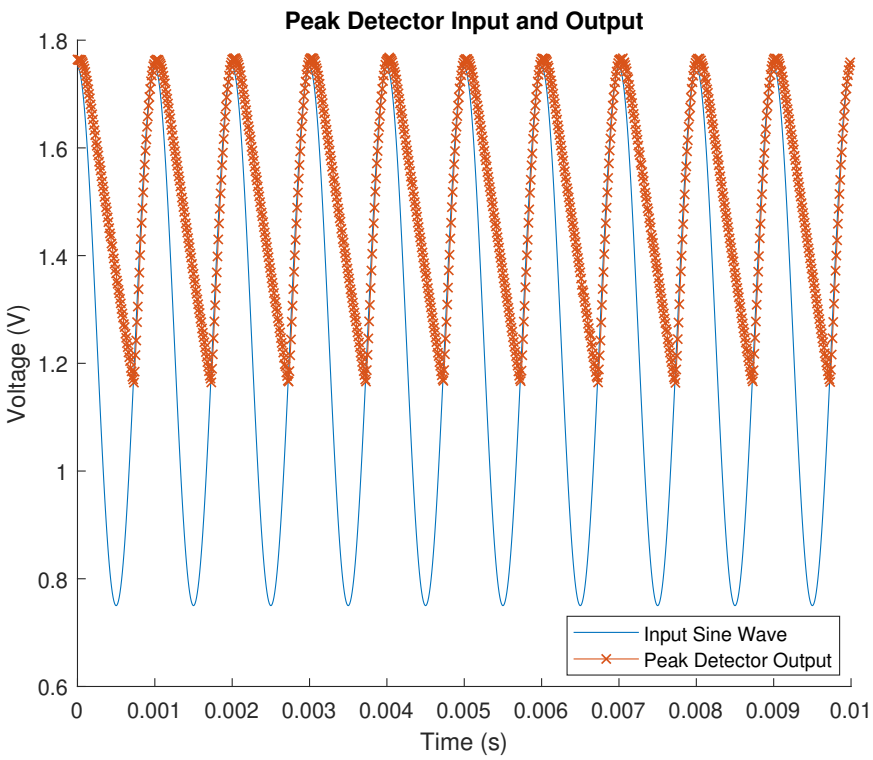


Figure 5.9: Peak detector output with a sine wave input.

FG transistors 3 and 9 are connected to the "ATK" and "DEC" biases of the peak detector, respectively. The connections list sets the FG transistors to be injected and connected to the circuit for biasing. The ATK bias is used to determine how closely the peak detector will track a rising signal. The DEC bias is used to set how closely the peak detector will track a

falling signal. A sine wave input was used to test the peak detector. The input and output of the peak detector can be seen in figure 5.9.

By setting the ATK bias to 8e-8, the output of the peak detector closely tracks a rising signal. With the DEC bias set to 1.5e-8 , the peak detector tracks a falling signal at about 1/2 of the slope. This type of peak detector setup is useful in spectral analysis. Using an array of bandpass filters, an analog signal can be decomposed into different frequency bands. Peak detectors can then be connected to the output of each bandpass filter. By connecting the output of the peak detector biased in the way described above with a low-pass filter, the RMS voltage of the signal for that frequency band can be extracted. This is a powerful tool for spectral analysis and event detection.

5.5 Conclusion

The RAMP 1.1 makes considerable improvements over the RAMP 1.0. The inclusion of all programming structures helps to reduce PCB sizing while also improving overall system power consumption. The upgrade to bandpass filters helps to improve the RAMPs performance as a front end ASP. By ensuring accurate filtering, ASP tasks can be performed with more reliability.

The additional stage of CABs allows the RAMP 1.1 to implement more complex designs while still conserving routing space, leading to less performance effecting parasitics that can have negative impact on precise ASP applications. The implementation of BLE programming from the modular MCU board allows for the RAMP to be reprogrammed wirelessly with low energy communication.

Chapter 6

An Acoustic Vehicle Detector and Classifier Using a Reconfigurable Analog-Mixed-Signal Platform

**This Chapter was published in the MDPI Journal of Low Power Electronics and Applications, Volume 10, Issue 1, March 2020. It has been re-purposed to fit in this dissertation by removing redundant information that has already been discussed in previous Chapters. To read the original, unedited, published paper, see citation [59].

This Chapter aims to highlight the ability of a field-programmable analog array (FPAA), more specifically, the Reconfigurable Analog and Mixed-Signal Platform (RAMP) as a reconfigurable tool for analog pre-processing. General analog signal processing (ASP) has typically been limited to custom design. A specific problem space exists, and an application-specific integrated circuit (ASIC) must be designed and fabricated as a solution. With advancements in FPAA technologies such as the RAMP, custom solutions no longer require fabrication of a new IC. The RAMP allows for tailored design solutions to be synthesized directly on chip. While the application space is of course not infinite, the RAMP does an adequate job of covering a wide array of analog pre-processing areas. This Chapter will showcase an example of using the RAMP to solve an acoustic detection and classification problem previously only done on ASICs.

The wireless sensor nodes used in a growing number of remote sensing applications are

deployed in inaccessible locations or are subjected to severe energy constraints. Audio-based sensing offers flexibility in node placement and is popular in low-power schemes. Thus, in this Chapter, a node architecture with low power consumption and in-the-field reconfigurability is evaluated in the context of an acoustic vehicle detection and classification (hereafter "AVDC") scenario. The proposed architecture utilizes an always-on FPAA as a low-power event detector to selectively wake a microcontroller unit (MCU) when a significant event is detected. When awoken, the MCU verifies the vehicle class asserted by the FPAA and transmits the relevant information. The AVDC system is trained by solving a classification problem using a lexicographic, nonlinear programming algorithm. On a testing dataset comprising of data from ten cars, ten trucks, and 40 seconds of wind noise, the AVDC system has a detection accuracy of 100%, a classification accuracy of 95%, and no false alarms. The mean power draw of the FPAA is 43 μW and the mean power consumption of the MCU and radio during its validation and wireless transmission process is 40.9 mW. Overall, this Chapter demonstrates that the utilization of an FPAA-based signal preprocessor can greatly improve the flexibility and power consumption of wireless sensor nodes.

6.1 WSNs and Acoustic Techniques

Over the past decade, the rapid rise of the Internet of Things (IoT) has facilitated the advancement of remote sensing by simplifying the design and expanding the scale of wireless sensor networks (WSNs) [60–64]. Yet, WSN expansion has faced bottlenecks due to the energy constraints and accessibility restrictions imposed by the remote placement of sensor nodes [60]. Thus, there is a growing demand for reliable devices with ultra-low power consumption and high wireless reprogrammability.

To achieve low power consumption, minimally preprocessed data is often transmitted to a base station for further analysis. However, frequent wireless communication can be a leading contributor to energy cost in WSNs. Another school of thought is to locally process data and only transmit relevant information. In general, there exists a tradeoff between computation and communication overhead in WSNs [1]. Nevertheless, local processing may be a more viable choice in scenarios where sensor data are sparse in relevant information

content or where the necessary computational tasks can be performed at low energy cost.

Audio data are often sparse in relevant information, which makes audio sensor nodes good candidates for the local-processing approach. Audio capture and analysis inherently consumes less energy than its video counterpart, making acoustic techniques good for low-power sensing schemes [65]. AVDC has previously been used as a benchmark for the validation of low-power node architectures [1]. AVDC has applications in traffic monitoring and military surveillance, especially in scenarios where sensors can not be placed in the line of sight of the classification target [61].

Vehicles, like many other objects with rotary or oscillatory components, can be identified from the frequency content of their acoustic signature. Hence, wavelet analysis, which classifies objects based on temporal frequency content, has been used in AVDC-like applications [66]. However, wavelet analysis is energy-intensive due to involved digital computations.

Amplitude analysis is another oft-used audio processing method. It is typically executed through a thresholding scheme like the structural integrity monitor presented in [67]. Adaptive thresholding was proposed for vehicle detection in [68]. The approach taken in [69] uses an envelope detector and comparator to construct an event detector for a low-power cargo monitoring tag. Amplitude analysis affords simple, energy-efficient solutions to a multitude of problems but causes a high false alarm rate due to its low event-oriented specificity. For the AVDC task, a combination of spectral analysis and amplitude analysis could achieve good accuracy while maintaining low power consumption and simplicity.

The ability to implement training with the node out-of-the-loop would help hasten node deployment and facilitate remote reconfigurability. With the node out-of-the-loop, training can be done offline with the same reconfigurable architectures being used in the field. With this approach, system parameters can be re-adjusted in the field if more data become available, and specific adjustments can be made for individual node performance. Deployed nodes can also be repurposed for different remote-sensing tasks without the need for retrieval. Minimizing the number of programming cycles undergone by the sensor node potentially saves time and energy while extending the lifespan of the node.

6.1.1 Proposal, Novelty, & Overview

Section 2.2 makes the case for ASP as an advantageous alternative to digital signal processing (DSP). However, analog circuits can be susceptible to electrical noise and environmental changes. Analog electronics are also ill-suited for certain tasks (such as sequential logic that is more readily handled by a DSP). It may be more synergistic to use a combination of analog and digital electronics for a general-purpose node architecture. Hence, this Chapter proposes a hybrid node architecture comprising of an always-on, mixed-signal processor (MSP) and an MCU that is only awoken when a significant event is detected by the MSP. In this way, the intrinsic device physics of analog components are leveraged to perform computations that would inherently require more power on fully digital platforms while still retaining the abilities to wirelessly network and perform sophisticated digital computation. [1, 60, 70] have previously shown that this type of configuration can facilitate significant power savings.

Although this Chapter expands on the work described in [1], it differs in a few aspects from previous works that offered solutions to the AVDC problem. Related works utilized ASIC solutions [1, 60, 67, 68, 70]. ASICs offer superb performance, ultra-low power consumption (especially with an integrated MSP), and good robustness. However, ASIC development is time-consuming, and ASIC-based solutions can be limited in their ability to handle future changes to procedures or overall tasks. These issues are mitigated in this Chapter by using a low-power FPAA. Compared to an ASIC-based solution, an FPAA-based solution can leverage mixed-signal computation and a reconfigurable architecture for a more expansive scope.

The RAMP introduced in [39] is the base for the preprocessing circuitry in this Chapter. By utilizing the RAMP, which contains a custom-designed FPAA at its heart, a sensor node with immense post-deployment reconfigurability, flexibility, and versatility can be achieved. As a consequence of using the RAMP, the event-detection stage of the AVDC system proposed in this work is different from that used in [1], which was based on an analog ASIC. In this work, floating-gate transistors (FGs) are used for comparator threshold generation instead of digital potentiometers, a lookup table (LUT) is used for template matching instead

of a complex programmable logic device, and a PanStamp MCU is used for transmission of the final decision instead of a TelosB digital platform. The subcircuits used in the spectral-analysis stage of the MSP also have different topologies than their counterparts in [1], and a digital debouncing circuit is introduced to the event-detection stage. A comparison between the results of this work and [1] highlight the improvements in power consumption and accuracy that are possible by using a reconfigurable system rather than an ASIC.

To take full advantage of the increased precision afforded by generating comparator thresholds with FG-transistor-based circuits, a new training algorithm is proposed in this Chapter. The training algorithm used in [1] used brute enumeration of comparator thresholds, keeping the MSP in-the-loop thoughout the entire training process. This methodology is only practical if threshold increments are coarse; hence, the methodology proposed in this Chapter uses a two-step, continuous-optimization procedure to determine comparator thresholds and requires significantly less communication with the MSP.

In summary, this Chapter evaluates the performance of an FPAA (specifically the RAMP) within the context of a vehicle detection and classification application. The remainder of the Chapter is as follows: The topology of the AVDC circuit is explained in Section 6.2. Section 6.3 discusses the process of training the AVDC system. Section 6.4 demonstrates the performance of the AVDC circuit in comparison to historical work. Finally, Section 6.5 offers concluding remarks.

6.2 Vehicle Detector Configuration

As mentioned in Section 6.1.1, the pivotal use of an FPAA-based MSP and the implications of FPAA usage distinguish the work proposed in this Chapter from that on which it is based [1, 60, 70]. The "always-on" audio MSP constructed on the RAMP performs a combination of spectral and amplitude analyses; vehicles are identified by their instantaneous frequency content by first decomposing audio data into several frequency channels and then comparing the signal power in the frequency channels to predetermined templates. A match to the spectral template for a vehicle will wake the MCU from its low-power sleep mode. The AVDC system leverages the following signal-processing steps:

Spectral-Analysis Stage:

1. *Spectral decomposition* using a filterbank of bandpass filters (BPFs)
2. *RMS envelope estimation* using a bank of root-mean-square (RMS) detectors cascaded with a bank of ripple-smoothing, adaptive-time-constant (adaptive-τ) lowpass filters

Event-Detection Stage:

1. *Digitization* using a bank of comparators
2. *Digital "debouncing"* using a starved inverter
3. *Template matching* using a LUT
4. *Final decision transmission* using a PanStamp MCU

Figure 6.1 shows a block diagram of the AVDC system, and the various signal-conditioning steps of the MSP are discussed in more detail in the subsequent subsections.

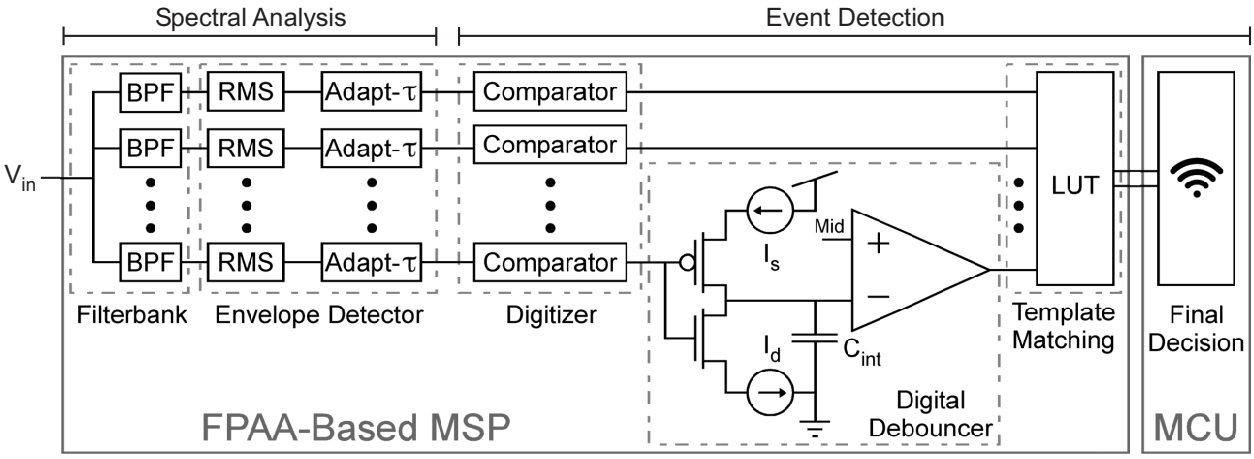
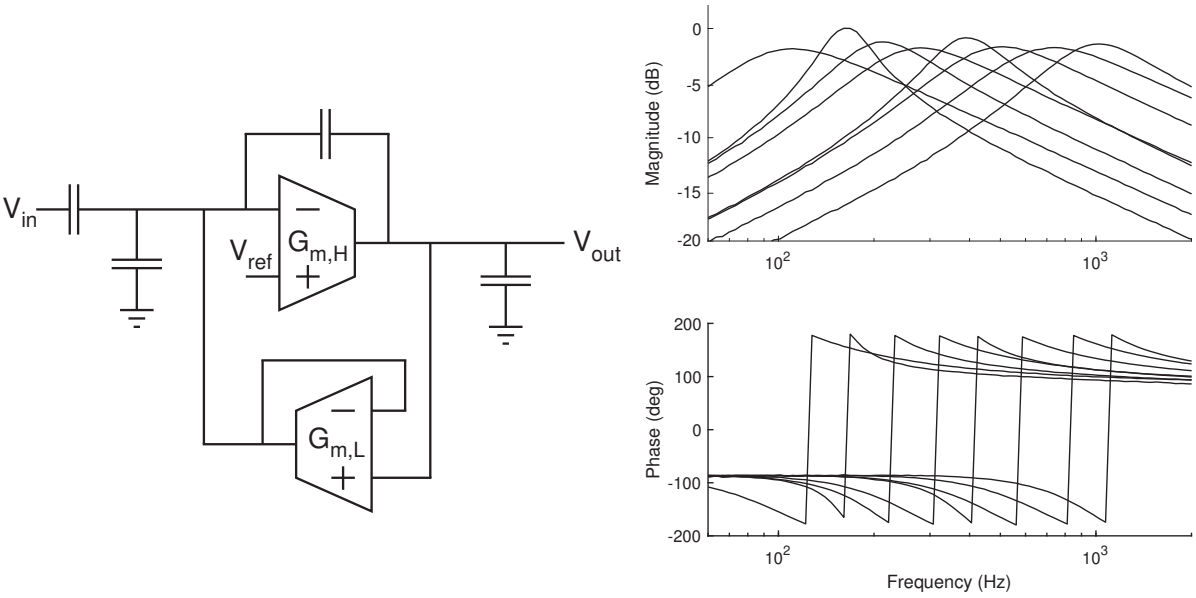


Figure 6.1: Block diagram of the AVDC circuit.

6.2.1 Spectral Decomposition

Operational transconductance amplifier (OTA)-based BPFs, shown in Figure 6.2 (a), are utilized in a filterbank to decompose the input audio. The BPFs used in the RAMP are

based on a design demonstrated in [71], and they have the benefit of independently-adjustable corner frequencies. The corner frequencies of a BPF is set by tuning the transconductances of its constituent OTAs via FG current biases.



(a) (b)

Figure 6.2: (a) Schematic of the BPFs used and (b) frequency response of the filterbank.

The filterbank is initially constructed with eight BPFs configured in a half-octave spacing from 77 Hz to 1113 Hz. Although the spacing between filters in adjacent channels and the exact corner frequencies can be tuned precisely [71], this level of precision is found to be of low importance for the AVDC proposed in this Chapter—the training algorithm proposed in Section 6.3 (b) finds that a few channels contain redundant information. These redundant channels are removed from the spectral-analysis stage of the final AVDC system. Figure 6.2 (b) exhibits the frequency response of the initial filterbank. To minimize the output distortion of BPFs, the range of the input to V_{in} should be properly scaled with the mean at midrail, and V_{ref} should be referenced to midrail.

6.2.2 RMS Envelope Estimation

After obtaining the frequency decomposition, it is desirable to estimate the RMS envelopes of the BPF outputs to assess the instantaneous signal power in each frequency band. The peak detector described in [72] is biased as an RMS detector to perform this task. The RMS detector, depicted in Figure 6.3 (a), has two independently adjustable parameters, attack rate ($G_{m,A}$) and decay rate ($G_{m,D}$), which correspond to the transconductances of the constituent OTAs. Adjustment of OTA transconductances also changes the output ripple of the RMS detector. The RMS detector operates like an asymmetric integrator whose output is given by:

$$\frac{dV_{RMS}}{dt} \approx \begin{cases} G_{m,A}(V_{in} - V_{RMS}), & V_{in} > V_{RMS} \\ G_{m,D}(V_{in} - V_{RMS}), & V_{in} \leq V_{RMS} \end{cases} \tag{6.1}$$

where V_{in} and V_{RMS} denote the input and output of the RMS detector, respectively.

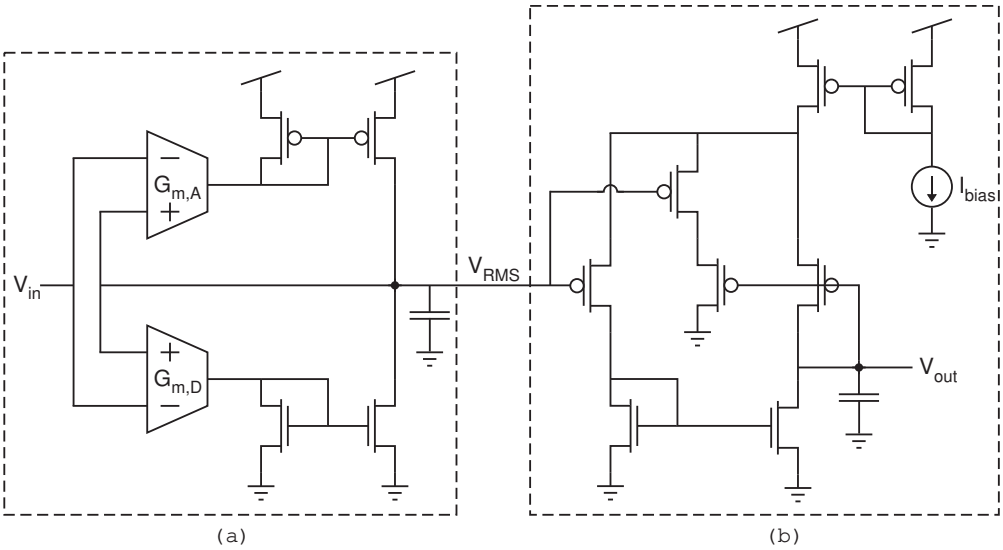


Figure 6.3: Schematics for the (a) RMS detector and (b) Adaptive-τ filter.

Despite careful tuning, the RMS detector output will invariably contain some ripple, especially if an accurate envelope estimate is desired. Thus, a bank of adaptive-τ lowpass filters based on the topology presented in [72] is implemented for ripple rejection. At its core, the adaptive-τ, shown in Figure 6.3b, is an OTA-based lowpass filter. However, the

OTA used in an adaptive-τ filter is designed to have a transconductance that increases with increasing input amplitude [72]. Hence, the time constant of the adaptive-τ filter decreases with increasing input amplitude.

6.2.3 Digitization

The smoothed RMS envelopes are digitized using a bank of comparators. Comparators on the RAMP have built-in hysteresis, which aids in producing a steady LUT output despite residual noise from the spectral analysis stage. Comparator reference voltages are generated by using FGs to source or sink current through resistors. The comparator architecture in the MSP (depicted in Figure 6.4) consists of a differential amplifier cascaded with an inverter and an "edgifier" circuit. The edgifier circuit was proposed in [73] and uses starved inverters to accelerate the rising and falling edges of its input signal, making it useful for the digitization of slow analog signals.

6.2.4 Digital Debouncing

For certain channels (notably the highest frequency channel), the hysteresis inherent to a RAMP comparator is insufficient to generate a steady digital input to the LUT. Continual oscillations in the LUT input may cause excessive querying and lead to unwanted power consumption. A digital debouncing circuit (shown in Figure 6.1) is used to mitigate this issue. To incite a state change in the output, this debouncing circuit requires that the input signal maintain its digital state for a minimum amount of time. This minimum time requirement for the debouncer is analogous to the "setup time" requirement of a D-latch. The digital debouncing circuit is constructed by cascading a time-voltage conversion circuit with a comparator referenced to midrail. More particularly, the time-voltage conversion circuit is an asymmetrically-starved inverter with the output connected to a capacitor. This implementation allows for the "setup times" necessary for the "high" or "low" state to be independently adjusted. If the voltage on the capacitor is near either supply rail prior to an input state change, the "setup time" of the debouncer can be approximated as follows:

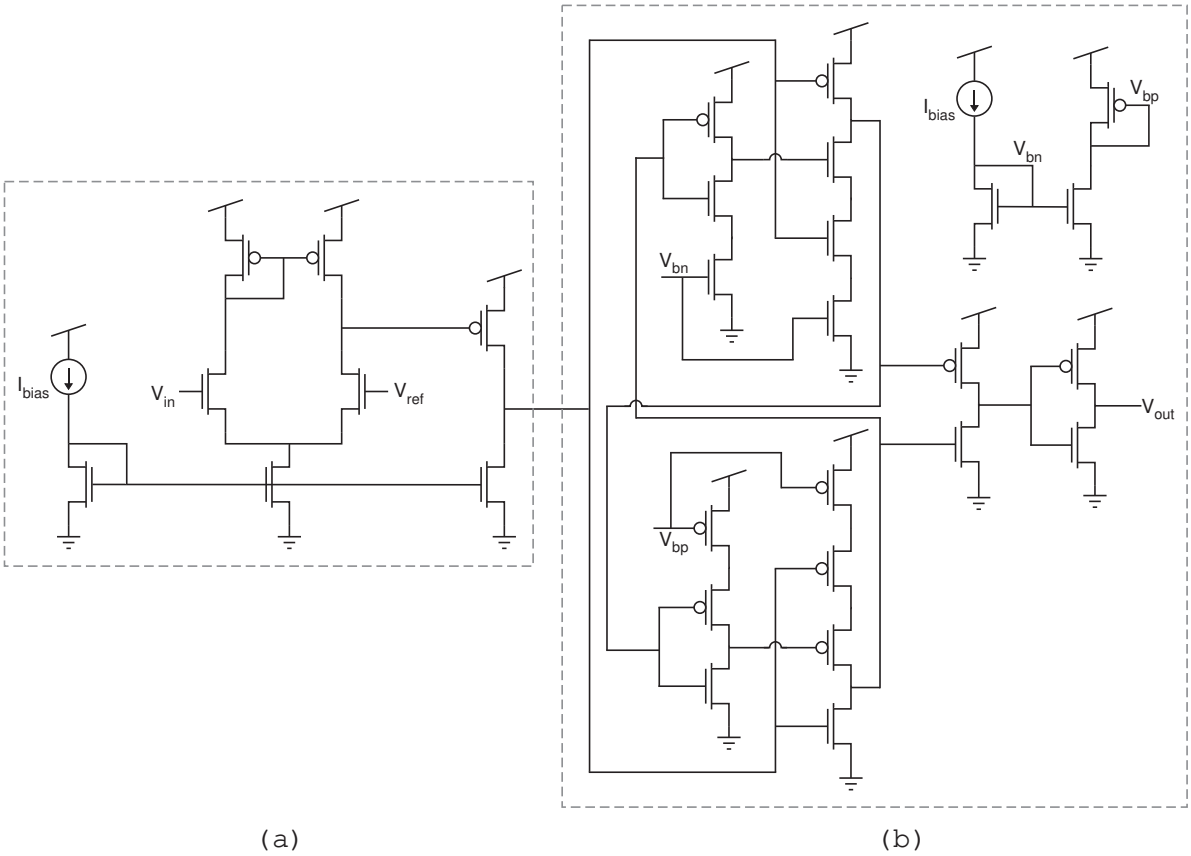


Figure 6.4: Schematics for the (a) comparator with attached (b) edgifier.

$$t_{setup} \approx \begin{cases} \frac{Mid \cdot C_{int}}{I_s}, & \text{when input transitions from } 1 \rightarrow 0 \\ \frac{Mid \cdot C_{int}}{I_d}, & \text{when input transitions from } 0 \rightarrow 1 \end{cases} \tag{6.2}$$

The constants used in Equation 6.2 are labeled in Figure 6.1.

6.2.5 Template Matching

Once stable comparator outputs are generated, an LUT is used to match comparator activation patterns to one of the three vehicle classes considered in this Chapter: "noise/no car," "car," and "truck." Due to chip real-estate considerations, the LUT fabricated on the RAMP is a digital framework with six inputs and two outputs that checks compliance with Boolean expressions; hence, the LUT is configured to use one output to assert vehicle presence and another output to assert vehicle type (i.e. if a truck is present). To configure a

LUT, the RAMP software decompiles the necessary Boolean expressions into logical tables that are then stored in the two SRAM arrays comprising the LUT. When the LUT receives an input, it asynchronously queries the SRAM cells with the corresponding addresses to determine the outputs. Figure 6.5 presents a block diagram of the LUT architecture.

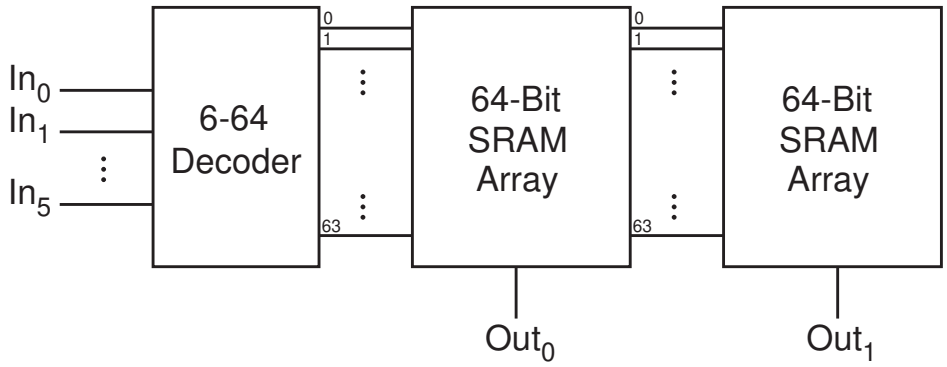


Figure 6.5: LUT block diagram.

6.2.6 Final Decision Transmission

The preliminary decision made by the MSP is susceptible to errors due to wind noise and differences between approaching, present, and receding vehicles. Differences in vehicle size and proximity can lead to scenarios where the LUT incorrectly identifies the vehicle class. Thus, an MCU is used to verify the decision of vehicle presence by implementing the decision-accumulation scheme demonstrated in [1]. In the decision-accumulation scheme, the LUT output that asserts the presence of a vehicle is used as an interrupt or "wake-up" pin for the MCU. Once the LUT has generated an interrupt on the "vehicle presence" pin, the MCU wakes up and records both outputs from the LUT. The MCU continues sampling the LUT output until the LUT's "vehicle presence" pin remains low for 100 ms. This additional pause helps confirm that the vehicle that triggered the interrupt has moved out of the range of the audio sensor.

The MCU then decides the vehicle class. If the LUT's "vehicle presence" pin has been asserted for a minimum of 50 ms, the MCU decides that a vehicle is present. Otherwise, the MCU decides that the LUT has false triggered, and it returns to its low-power sleep mode.

If a vehicle is present, the MCU must activate its radio transmission module to send the final decision results to a base station. However, before radio transmission, the MCU determines the vehicle type by comparing the number of samples for which the LUT's "vehicle type" pin has been asserted to a threshold. If this threshold (which is estimated to be 5000 samples using basic training data statistics) is exceeded, the vehicle is classified as a truck; otherwise the vehicle is classified as a car. The MCU decision-making process is summarized in Figure 6.6.

6.3 Classifier Training

6.3.1 Data Preparation

Audio data preparation is the first step in the AVDC training process. The audio data used in this Chapter were collected previously in [21]; the data are ten-second audio clips (sampled at 4 kHz) from 20 cars and 20 trucks as well as 80 seconds of wind noise. Truth vectors indicating the presence and class of each vehicle were constructed via human inspection. The normalized audio and their respective truth vectors were randomly concatenated into a training and testing dataset with the same size and class composition. The training dataset was passed through the spectral-analysis stage of the AVDC system. The output stream from the adaptive-τ filter in each channel was recorded via a National Instruments 6259 data acquisition card and then imported into MATLAB® to be used in the threshold estimation procedure, as described in the next two subsections. A spectrogram demonstrating the output of the spectral-analysis stage in response to the testing dataset is shown in Figure 6.7 for reference.

6.3.2 Comparator Threshold Optimization

Once the data have been prepared, training the AVDC system entails configuring the event-detection stage of the MSP. The first step in the configuration of the event-detection stage, and the focus of this subsection, is the selection of comparator reference voltages (hereafter "comparator thresholds"). On the RAMP, comparator threshold voltages are generated

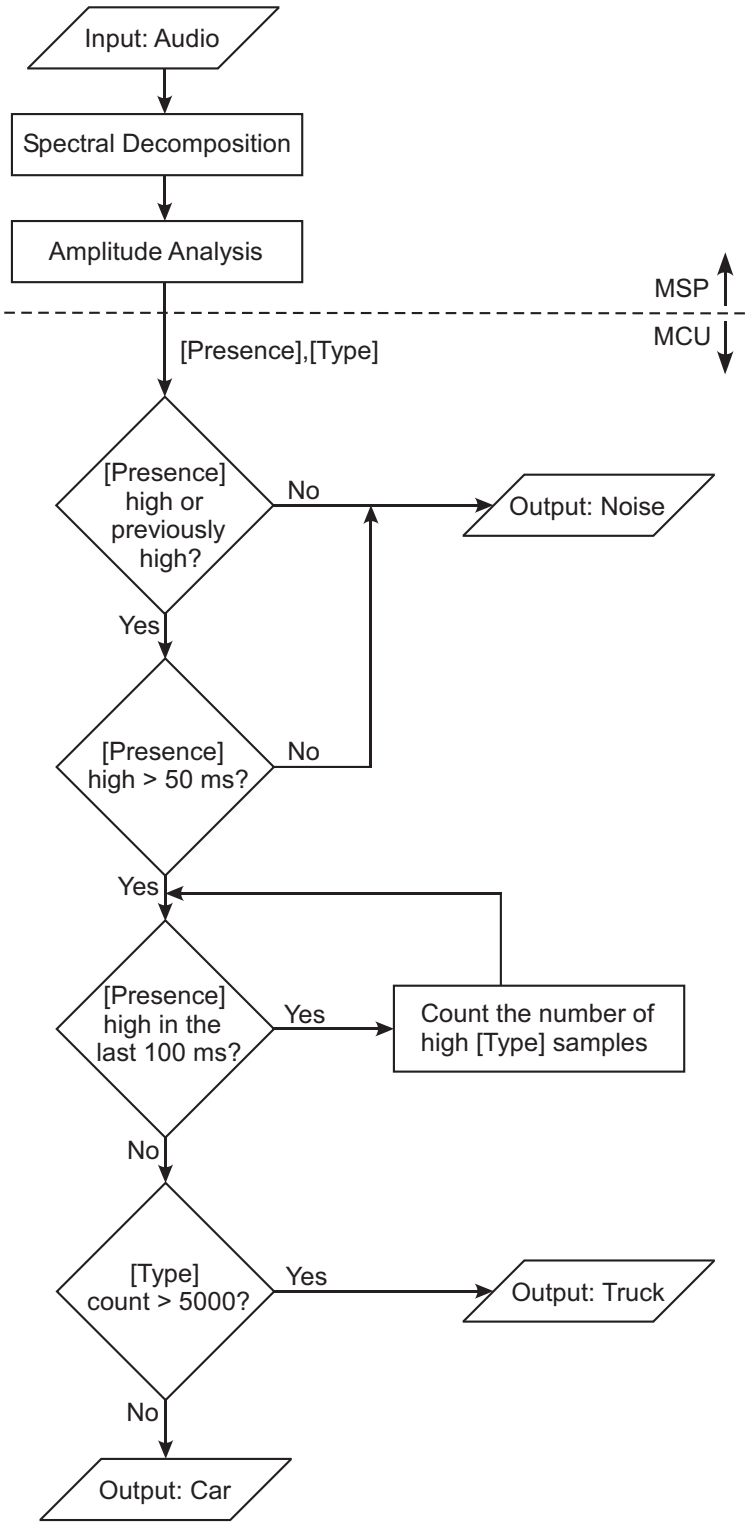


Figure 6.6: Process used by the MCU to make the final detection and classification decision. Section 6.3 discusses methods for normalizing the input audio and configuring the amplitude analysis circuits (i.e. the AVDC event-detection stage) in detail.

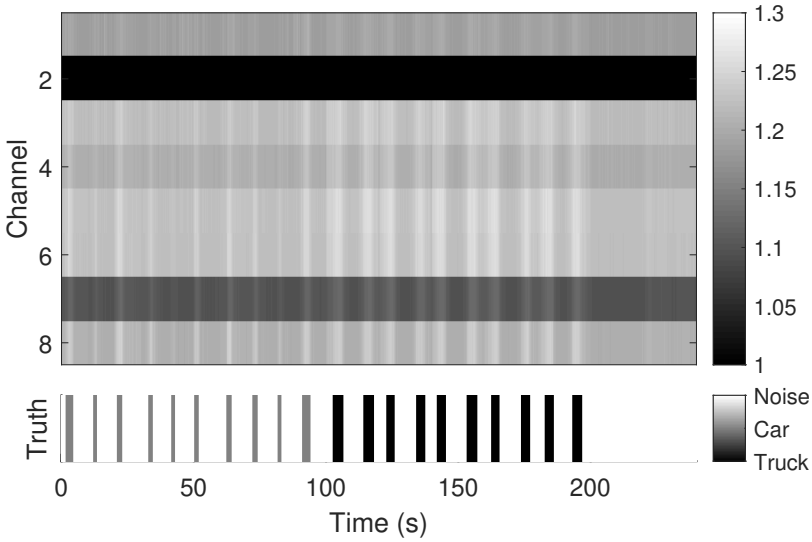


Figure 6.7: Spectrogram showing the output voltage (V) of each channel of the spectral-analysis stage to the testing dataset. Channels with higher indices have higher center frequencies.

by using FGs to source or sink current through resistors. These threshold voltages should be selected such that comparator activation patterns (hereafter "codewords") are indicative of a specific vehicle class: (1) noise, (2) car, or (3) truck. Hamming distance, which is the minimum number of bit replacements required to match two binary strings, is a good measure of the similarity between two codewords. Hence, comparator thresholds (denoted by τ) are selected to maximize the mean Hamming distance between codewords triggered by different vehicle classes (denoted by M_1) and to minimize the mean Hamming distance between codewords triggered by the same vehicle class (denoted by M_2). These criteria can be phrased as a multiobjective nonlinear programming problem. In this Chapter, maximization of M_1 is prioritized to reduce false alarms and misclassification, which cause unnecessary energy expenditure.

As noted in Section 6.1.1, it is desirable to keep the MSP out-of-the-loop for most of the training process to improve the training rate, reduce the degradation of FGs, and facilitate post-deployment adjustments. The event-detection circuits in the MSP are well characterized, so empirical measurements of spectral-analysis stage outputs can be utilized in software simulations to estimate configuration parameters for the event-detection stage of the MSP.

These parameter estimates can be easily "ported" back to the RAMP for implementation in hardware. Hence, this optimization problem is solved with a two-step, lexicographic approach [74] using the MATLAB® *fmincon* solver and the *interior-point* algorithm.

Lexicographic approaches are multiobjective optimization strategies that sequentially optimize a series of objectives in order of decreasing priority. Lexicographic approaches can be used to obtain Pareto-optimal solutions [75]. The optimization problem proposed in this Chapter, has two objectives, M_1 and M_2. In the first solution step, M_1 is maximized while M_2 is constrained by the relaxed boundary condition $M_{2_{Tar}}$:

$$M_{1_{Max}} = \max\left(M_1(\tau)\right) \quad s.t. \begin{cases} \tau_{lb} \leq \tau \leq \tau_{ub} \\ M_2(\tau) \leq M_{2_{Tar}} \end{cases} \tag{6.3}$$

In Equation 6.3, τ_{lb} and τ_{ub} denote the bounds on the comparator thresholds and are set to be the minimum and maximum output voltage from each channel of the spectral-analysis stage, respectively.

In the second solution step, the "argument" τ of the "minimum" of M_2 is sought:

$$\arg\min_{\tau}\left(M_2(\tau)\right) \quad s.t. \begin{cases} \tau_{lb} \leq \tau \leq \tau_{ub} \\ M_1(\tau) \geq (1-\epsilon)M_{1_{Max}} \end{cases} \tag{6.4}$$

where the optimum from the first problem ($M_{1_{Max}}$) is relaxed by a factor of ϵ (typically around 10%) to act as a constraint on the value of M_1 for acceptable solutions. Objective functions M_1 and M_2 are computed by weighting the mean Hamming distance between each pair of codewords triggered by the relevant comparison classes by a factor that is indicative of the specificity of the codeword. Defining the objectives in this manner rewards codewords for their specificity to a particular vehicle class and for their distinctness from codewords triggered by other vehicle classes. The underlying formula for computing M_1 and M_2 can be written as M_k for $k \in \{1, 2\}$:

$$M_k(\tau) = \frac{1}{3}\sum_{\psi=1}^{3}\left[\frac{G(P_{\psi,1,k}\,,\,P_{\psi,2,k}) \circ H}{e^T\,G(P_{\psi,1,k}\,,\,P_{\psi,2,k})\,e}\right] \tag{6.5}$$

$$\text{Where} \quad P_{:,:,1} = \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 1 \end{bmatrix} \quad \text{and} \quad P_{:,:,2} = \begin{bmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 3 \end{bmatrix} \tag{6.6}$$

In Equation 6.5, $\circ$ denotes the element-wise (Hadamard) matrix product, e denotes the 256x1 all-ones vector, H denotes the 256x256 matrix indicating the Hamming distance between pairs of binary codewords, and P denotes the three-dimensional matrix representing the required class comparisons. Figure 6.8 visually illustrates the class comparisons represented in P and the indexing scheme used in Equation 6.5. G represents the sparse matrix containing the occurrence frequency of each codeword comparison, which in turn is indicative of the class-specificity of the code words. G is computed using the following expression:

$$G(P_{\psi,1,k}, P_{\psi,2,k}) = F^T(P_{\psi,1,k}, *)\, F(P_{\psi,2,k}, *) \tag{6.7}$$

where $F(m, *)$ denotes the mth row of F, a 3x256 matrix whose first, second, and third rows indicate the number of times each codeword (ordered by decimal value) is expected to occur when noise, cars, or trucks, respectively are present. Codewords are in turn estimated from τ and the output from the spectral-analysis stage of the MSP. The thresholds found through the optimization procedure proposed in this section are mapped to current biases using the characterization models of the relevant FG current sources and sinks.

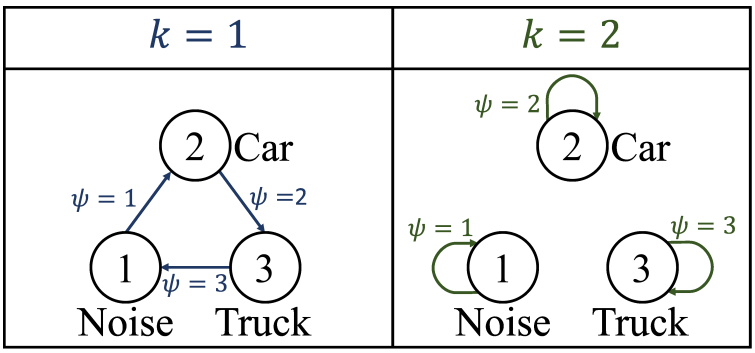


Figure 6.8: Graphs of the class comparisons represented in P. The comparisons for computing M_1 are in the leftmost column, and the comparisons for computing M_2 are in the rightmost column.

6.3.3 Lookup Table Configuration

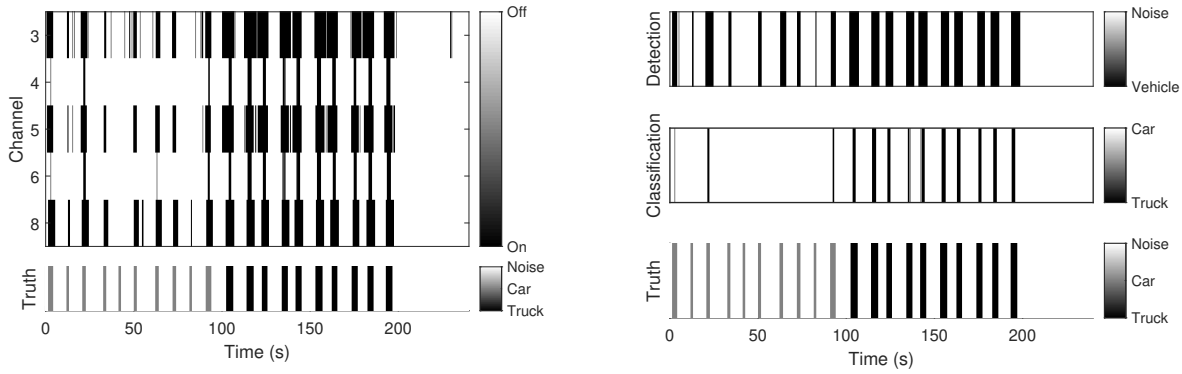
After threshold estimation, a LUT is configured such that the first output indicates vehicle presence and the second output indicates vehicle type. The RAMP contains 16 LUTs, each with six inputs and two outputs, that are available for usage in the AVDC. The flexible signal routing granted by the RAMP enables LUTs to be arrayed to obtain a variety of input-output combinations. However, we found that the LUTs did not need to be arrayed in this work; a few frequency channels contained redundant information, and the threshold selection algorithm placed the comparator thresholds for these channels near the lower or upper bounds (τ_{lb} or τ_{ub}). As a result, redundant channels had a mostly stationary comparator activation pattern and were pruned to preserve resources. The final AVDC system in this Chapter uses five frequency channels.

In this work, the number of nonredundant frequency channels is tractable, and comparator activation patterns are defined (where certain comparators generally activate when a vehicle is present, and other comparators generally activate when a truck is present). Hence, after the comparator threshold values have been determined by the optimization procedure, LUT Boolean expressions can be readily found via observation of the resulting codewords. Table 6.1 shows the percentage of codewords that belong to each vehicle class that also satisfy the Boolean templates used to configure the LUT. Figure 6.9 (a) shows the codewords in response to the testing dataset, and Figure 6.9 (b) demonstrates the output of the final configuration of the LUT to the testing dataset. Table 6.1 and Figure 6.9 both indicate that the performance of the LUT alone is satisfactory for vehicle detection yet unsatisfactory for vehicle classification, thus motivating the use of an MCU to interpret the output of the LUT and bridge lapses in LUT performance.

Table 6.1: Percentage of Codewords Satisfying LUT Templates in Response to the Testing Dataset (C_n denotes the Boolean state of the channel with index n) for AVDC system

		Vehicle Class		
LUT Output	**Codeword Template**	Noise	Car	Truck
Vehicle Presence	$C_3 \wedge C_5 \wedge C_8$	4.1%	68%	97%
Vehicle Type	$C_3 \wedge C_4 \wedge C_5 \wedge C_6 \wedge C_8$	0.0%	11%	50%

If there are a greater number of nonredundant channels or if the data contain a greater



(a) (b)

Figure 6.9: (a) Codewords in response to the testing dataset. Channels with higher indices have higher center frequencies. (b) LUT output in response to the testing dataset.

number of vehicle classes, more elaborate techniques may be necessitated for LUT template selection. In such a scenario, it may be possible to leverage multiclass classification algorithms [76] to aggregate codewords into their respective vehicle classes and then use logic minimization methods to resolve the final LUT templates.

6.4 Results

After the training procedure, no changes are made to the AVDC system. Figure 6.10 shows the process of making a vehicle detection, starting from the raw audio signal from the sensor. The label for vehicle presence is shown below the input audio signal to give an idea of when the vehicle is near the microphone. "Detection Interrupt" indicates that a vehicle has been detected by the MSP. The "Classification" output from the LUT is used to implement the decision-accumulation scheme described in [1]. Once the MCU makes the final decision on the vehicle type, a radio transmission is sent (which is registered as a spike in power draw).

The MCU, which consists of the microcontroller and its peripheral devices (eg. the radio transmitter), can operate in three distinct states. Each state's power consumption is listed in Table 6.2. The first state is "sleep mode." Sleep mode is the low-power state of the MCU when it is monitoring a digital interrupt pin for a vehicle detection from the LUT.

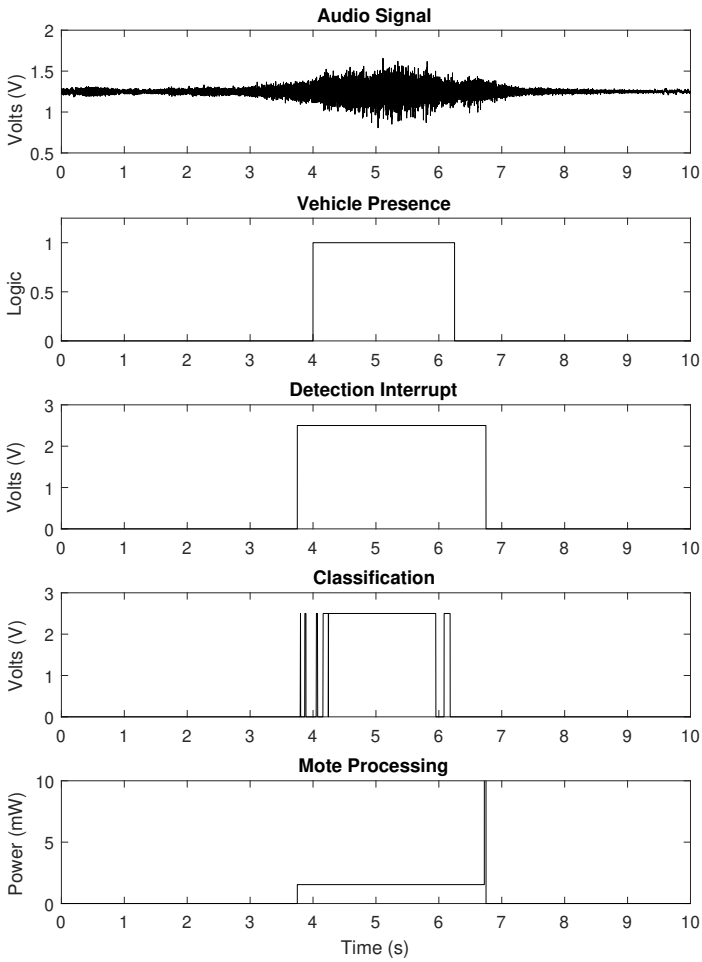


Figure 6.10: Process of AVDC detecting a vehicle from the raw audio sensor input to the final classification and radio broadcast.

When an interrupt has been sent from the LUT, the MCU enters the second state: "wake mode." In wake mode, the MCU powers up to make the final decision pertaining to vehicle presence and type. If the MCU decides that a vehicle is present, it enters the third state: "radio mode." Otherwise, it returns to sleep mode. Radio mode is the state in which the the MCU's radio transmitter is enabled to broadcast vehicle detections and classifications. Of the three modes, radio mode is the highest in power consumption, so the MCU immediately returns to sleep mode after the radio transmission is complete.

The testing data described in Section 6.3 were utilized to evaluate the performance of

Table 6.2: Power Consumption of AVDC system

	Device	Power
MSP Circuitry (Always On)	RAMP	43.0 μW
"Sleep Mode" LUT Monitoring	MCU	32.6 μW
	Total	75.6 μW
"Wake Mode" LUT Monitoring	MCU	1.49 mW
	Total	1.54 mW
"Radio Mode" MCU Broadcast	MCU	40.9 mW
	Total	41.0 mW

the trained AVDC system. Like the training data set, the testing data set consists of ten car samples, ten truck samples, and 40 seconds of wind noise. The detection and classification results of the trained AVDC for the testing dataset are shown in Table 6.3. The system proposed in this Chapter achieved a detection accuracy of 100% and a classification accuracy of 95% (one car was misidentified as a truck). The system had no false alarms during the 40 seconds of wind noise. False alarms cause the MCU to enter wake mode and possibly radio mode. Hence, each false positive contributes to energy consumption, which decreases the overall lifetime of a resource-constrained sensor node.

Table 6.3: Vehicle Detection and Classification Results of AVDC system

		Ground Truth		
		Car (10 Samples)	Truck (10 Samples)	Noise (40 seconds)
Rumberg [1]	Car	80%	0%	2 false alarms
	Truck	0%	100%	2 false alarms
	Noise	20%	0%	
This Work	Car	90%	0%	0 false alarms
	Truck	10%	100%	0 false alarms
	Noise	0%	0%	

Table 6.3 also includes comparisons to [1], which presented results from a system with a similar analog/mixed-signal processing chain but built as an application-specific device. Since the audio data and class labels used in both this Chapter and that paper are identical, and the signal processing circuits share some similarities (similarities are summarized in Section 6.1.1), comparisons to the work presented in [1] can provide a fair testimony to the merits of employing a reconfigurable system over an ASIC. In comparison to [1], which reported a classification accuracy of 80% for cars and four false positives, the AVDC presented

in this Chapter has a higher classification accuracy for cars and no false positives.

To demonstrate the power savings of the AVDC system, Figure 6.11 compares battery lifetime in three scenarios—(1) this work, (2) previous work utilizing an ASIC-based MSP [1], and (3) a system with a purely digital implementation. Figure 6.11 shows the advancements that were made by [1] compared to an all-digital implementation, increasing the estimated lifetime from 4 months to 2.4 years. Figure 6.11 also shows the improvements of this AVDC system over [1], increasing the estimated lifetime from 2.4 years to 7.5 years. Additionally, Figure 6.11 predicts each case for both radio transmission (w/ TX, for real-time systems that need to transmit on each event occurrence) and no radio transmission (w/o TX, for data-logging systems that do not need to transmit on each event occurrence).

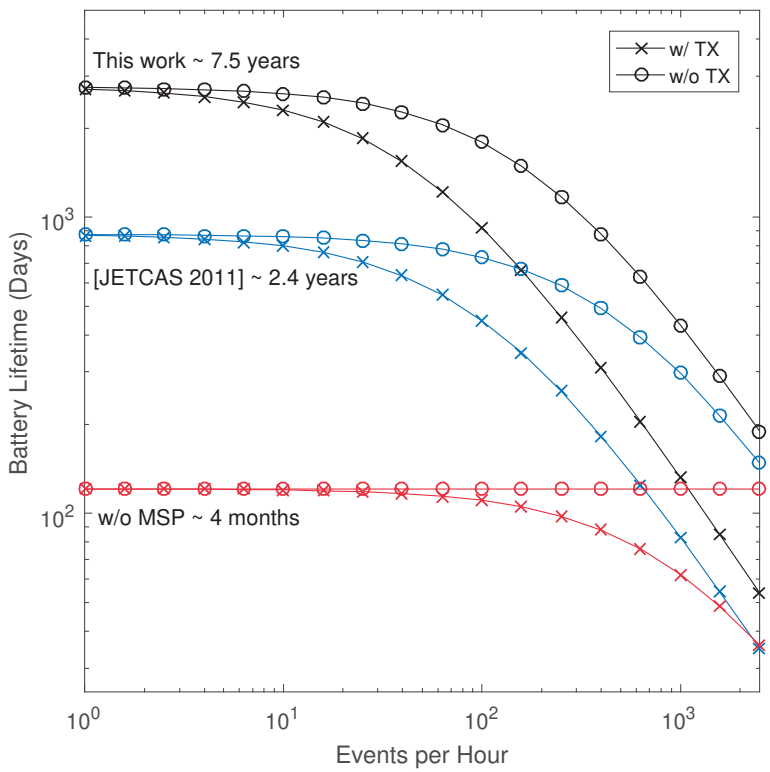


Figure 6.11: Battery lifetime for the AVDC system based on the number of events per hour. The plot shows results for this work, previous work ([1]), and a system with a purely digital implementation (w/o mixed-signal processing).

Other recent advances in acoustic classification systems have shown high accuracy with low power consumption. While different in application, the voice activity detectors described

in [77–79] provide useful comparisons in terms of acoustic processing. The voice activity detector presented in [77] reports an accuracy of 89% while consuming 6 μW. Another voice activity detector constructed from an analog filterbank feeding into a neural network reports approximately 85% accuracy while consuming only 1 μW [78]. A voice activity detector utilizing a deep neural network is presented in [79]; the neural network has a power consumption of 22 μW and the total system can consume up to 7.78 mW.

Acoustic sensing systems have also been applied to applications that are more similar to the AVDC. The system described in [80] reports a very-low power consumption of 12.2 nW and 95% classification accuracy when applied to vehicle detection (however, a different data set was used, so direct comparison is difficult). The vehicle detection system uses a custom micro-electromechanical system sensor and a custom digital-processing scheme for classification but is limited by a bandwidth of 500 Hz. The stereo-audio sensing application presented in [81] utilizes on-chip feature extraction for high accuracy and reports a power consumption of 55 μW, which is comparable to the 43 μW power consumption of the RAMP in this Chapter. The underwater acoustic monitoring system presented in [82] focuses on a different application space, but performs spectral analysis like many other sensing systems highlighted in this section. The underwater system has a reported power consumption of 62 μW for its MSP and was constructed primarily from commercially-available components.

The systems described in [77–82] were all fabricated in 65 nm, 90 nm, or 180 nm processes. Smaller process sizes naturally bring lower supply voltages and power consumption, particularly for digital systems. In contrast, the custom-designed RAMP IC was fabricated in a standard 0.35 μm CMOS process and leveraged a non-optimized, commercially-available MCU for the digital processing. Integration of custom digital-processing circuits on the RAMP IC would also significantly reduce the power consumption of the MCU, particularly in wake mode and radio mode.

While the acoustic monitoring systems in [1, 77–82] perform their respective tasks at low power levels, their application scope is constrained. In contrast, the AVDC implementation proposed in this Chapter was built on a highly reconfigurable platform, which opens up many more signal-processing possibilities; AVDC represents only a single application of the RAMP system, and many more low-power applications can be developed on the RAMP.

In summary, the RAMP system is able to provide a classification accuracy and low power consumption that is comparable to other ultra-low-power ASIC systems while also providing the flexibility to be reprogrammed for a wide variety of applications beyond AVDC.

6.5 Conclusion

A low-power, reconfigurable WSN node architecture centered on an FPAA-based mixed-signal preprocessor and a PanStamp MCU was presented in this Chapter. The proposed node architecture was evaluated in context of a resource-constrained AVDC scenario in order to evaluate the feasibility of its use in remote sensing applications. Using a mixed-signal audio processor and selectively waking the MCU for significant events, indeed leads to considerable power savings compared to an exclusively digital approach. Additionally, test results demonstrate that the AVDC system had a 100% detection accuracy, correctly classified 95% of the vehicles detected, and had no false alarms during 40 seconds of wind noise. The RAMP architecture has the potential to let sensor nodes adapt to changes in detection conditions and changes in mission directives without the need for physical recovery, making it indispensable for long-term remote-sensing applications.

Chapter 7

Linearization of Voltage-Controlled Oscillator-Based ADCs Using Floating-Gate Transistors

**This Chapter was published in the IEEE Journal Transactions on Circuits and Systems II: Express Briefs, ISSN 1549-7747, Early Access. It has been re-purposed to fit in this dissertation by removing redundant information that has already been discussed in previous Chapters and has additional information that did not fit in the published document. To read the original, unedited, published paper, see citation [83].

This Chapter presents a linearization method for the voltage-to-current (V-to-I) stage of voltage-controlled oscillators (VCOs). A floating-gate transistor is utilized at the VCO's input stage as a low-overhead method for V-to-I linearization. The resulting VCO is a ring oscillator with an extended linear voltage-to-frequency range. The ring oscillator VCO (RO-VCO) has also been included in a VCO-based analog-to-digital converter (VCO-ADC) to showcase how the linearization technique can be leveraged to improve system performance. The measured results from the linearized RO-VCO are compared to current state-of-the-art designs, and the VCO-ADC is compared to a standard architecture. The circuits were fabricated in a standard 0.35μm complementary metal-oxide-semiconductor (CMOS) process.

This Chapter emphasizes the design of the VCO-ADC circuit as a stand-alone device, highlighting the usefulness of FG transistors as a tool for input linearization. That being said,

this design was also a prototype for implementation in the new version of the Reconfigurable Analog and Mixed-Signal Platform (RAMP). The RAMP 1.1 and its peripheral circuitry were discussed in section 5, including how the VCO-ADC from the present section is implemented on the platform.

7.1 Ring-Oscillator-Based VCOs

Electronic oscillators are important circuit building blocks with widespread use in filters, modulators, power converters, data converters, and more. The VCO is designed so that it produces an oscillation frequency based on an input voltage. The ability to control an oscillation frequency is important for applications such as radio frequency modulation, phase-locked loops, and VCO-ADCs. For each application, it is necessary to have a wide frequency tuning range and a linear voltage-to-frequency (V-to-f) relationship for optimal and predictable circuit performance.

Ring-oscillator-based VCOs (RO-VCOs) offer the highest output frequency range of modern VCO designs [84]. However, they suffer from non-linearities existing between the input voltage and the output frequency. Recent work has demonstrated various methods for linearizing the V-to-f relationship in RO-VCOs [85–89]. However, a direct trade off exists between linearity and frequency range. Implementation of these linearization methods results in a limited frequency range.

As technology continues to advance towards smaller, wearable devices, the ability to convert data between the analog and digital domain reliably and with low power consumption becomes increasingly more important in electronics design. Among the many topologies of ADCs, the VCO-ADC allows for a completely digital design. The digital design allows the ADC to operate at the same low voltages required by most modern digital electronics and benefit from the power scaling described by Moore's Law. However, the VCO-ADC does not come without its pitfalls, and in its base form has a lack of linear relationship between the input voltage and oscillation frequency.

The nonlinearity discussed most often manifests in the input stages of the VCO-ADC, and this is where our design will focus. A VCO-ADC operates based on a relationship between

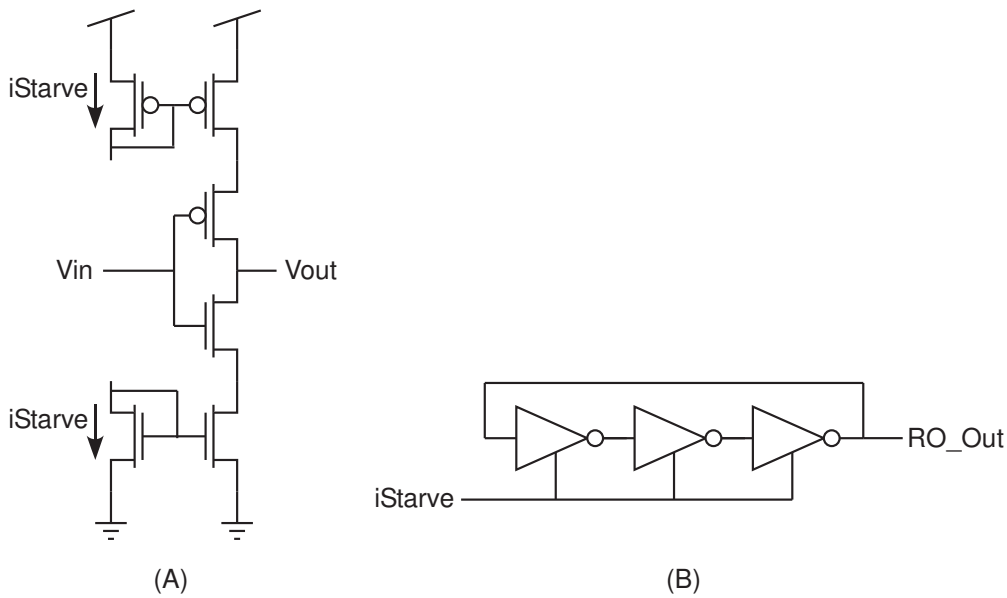


Figure 7.1: (a) Current-starved inverter schematic with iStarve current used to set delay. (b) Ring oscillator utilizing odd number of inverters and starvation current iStarve to generate output Vout.

input voltage and oscillator frequency. Without a reliably linear relationship between these components, the VCO-ADC will suffer from poor performance, especially in regards to its linear range.

This Chapter presents a method that leverages a floating-gate (FG) transistor to linearize the V-to-f conversion of an RO-VCO while simultaneously increasing its linear frequency range. The FG transistor is implemented at the input stage with low overhead and does not change the traditional RO-VCO architecture. The FG transistor also provides the added benefit of easily allowing for rail-to-rail input voltages. This design technique can be implemented with any VCO that must convert voltage to current for oscillation. Even a system that does not require precise linearity can still benefit from an increased frequency range.

This Chapter describes the design principles that are being implemented to improve the RO-VCO's performance and also reports measured results from an integrated circuit fabricated in a standard 0.35μm CMOS process. Measured data from the fabricated RO-VCO are also compared to similar state-of-the-art linear RO-VCO designs. Finally, the RO-VCO is demonstrated in an ADC application to highlight the benefits of the wide linear frequency range.

Section 7.2 will cover the proposed method of linearizing the V-to-f relationship with the RO-VCO. Section 7.2.3 will report on the measured results from the RO-VCO itself, showing how the linearization method improves the V-to-f range of the oscillator. Section 7.3 will cover an overview of the additional components needed to construct a VCO-ADC and report how the linearized RO-VCO is used to improve ADC linearity metrics. Measured results will be compared to similar state-of-the-art designs and includes a discussion on combining the proposed linearization method with other advanced forms of RO-VCO architectures for even better system performance.

7.2 Input Linearization of RO-VCOs

A VCO is a circuit that produces output voltage oscillations based on a voltage input. VCOs are categorized by their output waveform. The waveform will be seen as a sinusoidal waveform in VCOs such as Harmonic Oscillators, LC Oscillators [90], and Crystal Oscillators [91]. Other output signals such as sawtooth, triangular, or square waveforms are generally seen as the other VCO category, which are usually generated by Relaxation Oscillators. Relaxation Oscillators can further be broken down into Differential [92], Grounded-Capacitor [93], Emiiter-coupled [94], and delay-based ring VCOs [95]. This section focuses on the RO-VCO as it consistently performs with the widest output frequency range [96], a metric that is desirable for VCO-ADC performance.

An RO-VCO makes use of an odd number of inverters to force oscillation in a closed loop as shown in Figure 7.1. To allow for variable output frequency, an RO-VCO can take advantage of current starved inverters, rather than the standard digital inverter. A current starved inverter includes a current mirror above and below the standard n-type and p-type transistors. The current mirrors are then used to set a specific current that can flow through the inverter, which is especially important during inversion. The starved inverters stop a current dump from occurring between the supply and ground every time there is an inversion [97]. The mirrors ensure that only a desired amount of current will be flowing through the inverters at any time.

This current, iStarve, flowing through the inverter is also what sets the inverter's delay.

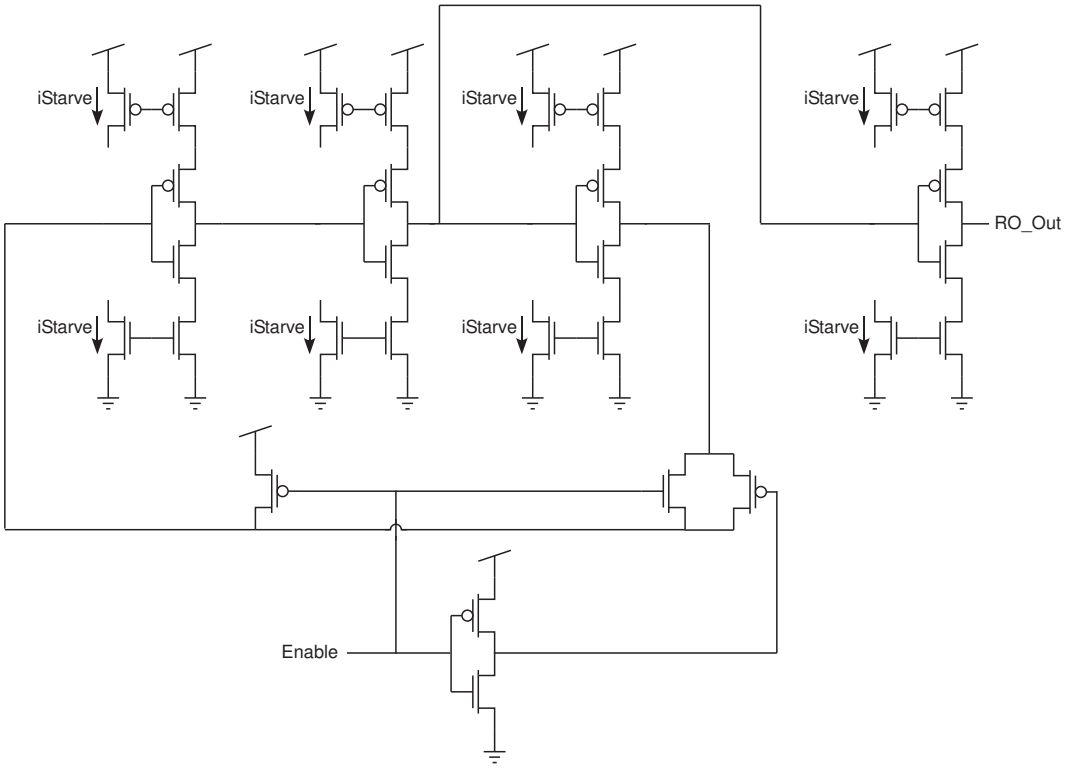


Figure 7.2: Schematic of ring oscillator clock generator with current starved inverters, fan-out, and start-up circuitry.

By setting the same current through each inverter, the frequency of the RO-VCO is controlled [98]. By changing the current and then by effect, changing the delay, the frequency of the RO-VCO can be modified. RO-VCOs are a good choice for a design which requires a wide range of frequency operation [99]. While other design topologies such as the standard grounded-capacitor relaxation oscillator can yield low noise, low jitter outputs [92], the RO-VCO offers the highest frequency range outputs.

Figure 7.2 shows the full RO-VCO, implemented with start-up circuitry and a fan-out inverter. The startup circuit is connected to the Enable pin. An RO-VCO implemented without the startup circuit has potential to get caught in a locked state where all terminals between inverters are held at mid-rail. To avoid this, the startup circuit holds the loop up at top rail when a “0” is on the Enable pin. When the Enable pin moves to “1”, the loop is released, causing the RO-VCO to begin oscillations. The fan-out inverter seen on the output is implemented with large transistor sizes, allowing it to drive capacitive loads that may attenuate the signal being generated by the smaller inverters inside the loop. Fig.

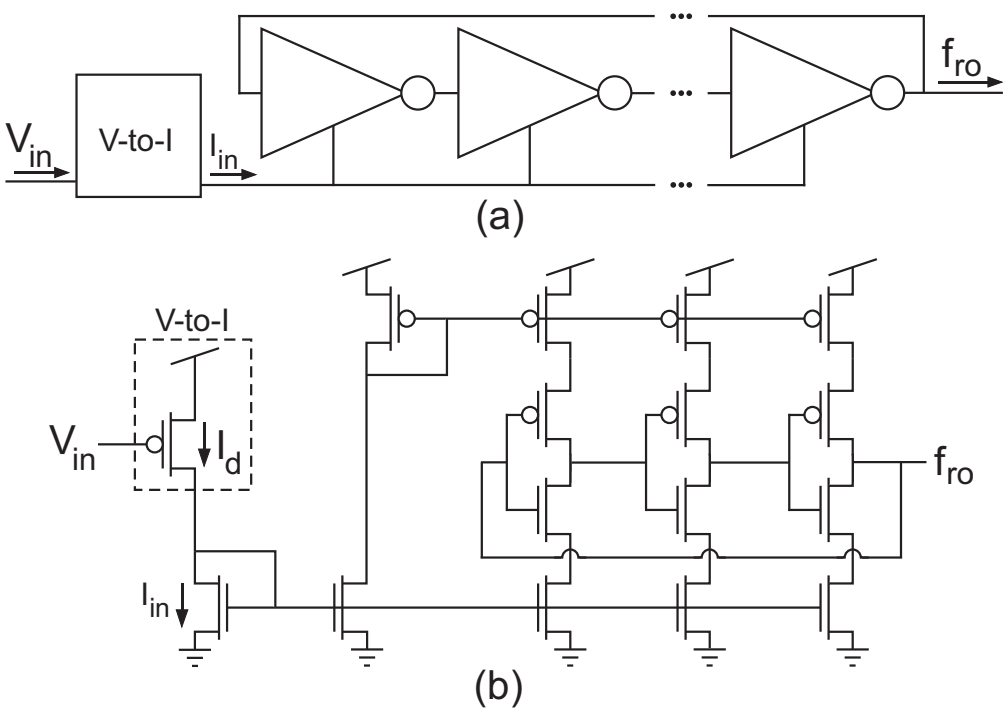


Figure 7.3: (a) Block diagram of a voltage-controlled ring oscillator. (b) Transistor-level schematic of a voltage-controlled ring oscillator with a pFET input stage.

7.3 (a) shows the block diagram of a generic RO-VCO. The process of oscillation begins by taking an input voltage (V_{in}) and converting it to a current (I_{in}) to bias the ring oscillator's current-starved inverters. An odd number of inverter stages are connected in a loop to force oscillation. The input current sets the delay within the loop of the RO-VCO and, accordingly, sets the output oscillation frequency (f_{ro}).

Any non-linearity between the input voltage and the generated current will be coupled into the RO-VCO frequency. This Chapter aims to significantly reduce the non-linearity within the input V-to-I relationship. Non-linearities also exist in the current-to-frequency (I-to-f) conversion; however, the I-to-f non-linearities are often small compared to those contributed by the V-to-I stage and will be dependent upon the specific VCO structure being used. Based upon simulations, the I-to-f non-linearities were $< 1\%$ of the full-scale range for the 3-stage RO-VCO used in this work and shown in Fig. 7.3 (b). Our FG linearization technique significantly reduced the V-to-I non-linearites such that the I-to-f stage dominated overall linearity for the circuit.

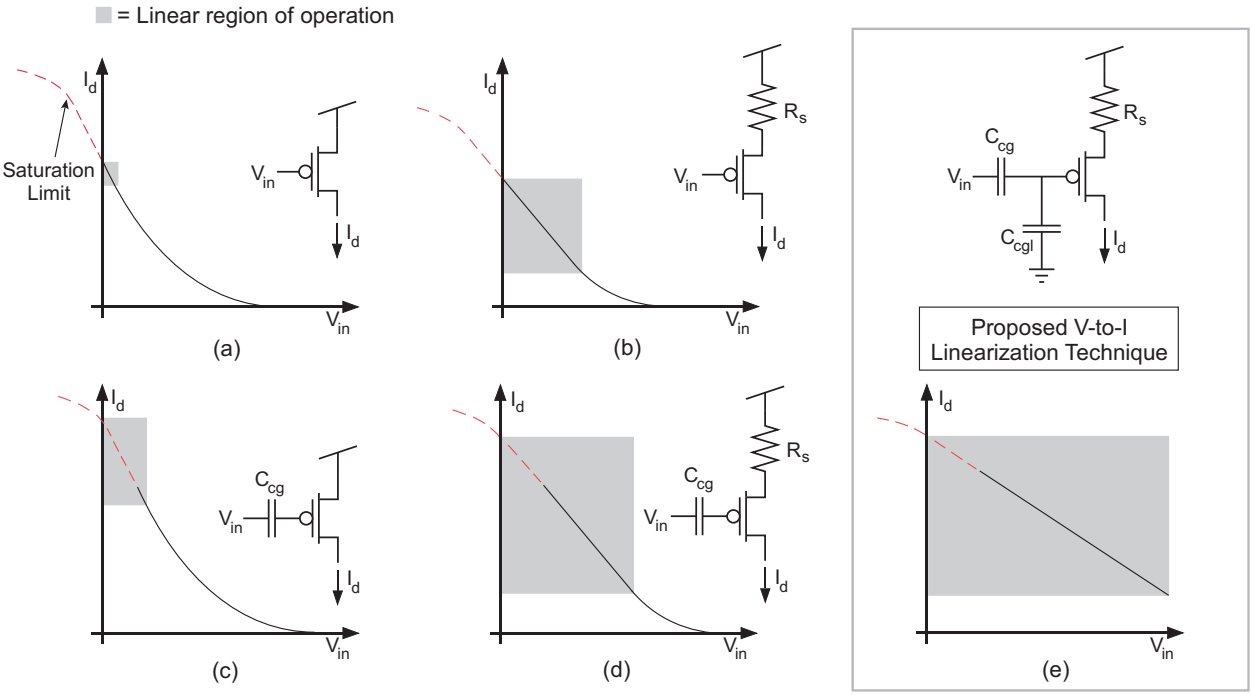


Figure 7.4: Demonstration of the linearization technique. The graphs show the relationship between the input voltage, V_{in}, and the drain current, I_d, for an input stage consisting of (a) a pFET with no linearization techniques, (b) a pFET with source degeneration, (c) an FG pFET, (d) an FG pFET with source degeneration, and (e) an FG pFET with both capacitive division and source degeneration.

7.2.1 FG Transistor Implementation

For the RO-VCO structure of Fig. 7.3 (b), a pFET is used as the input stage to convert the input voltage to a current. A pFET's V-to-I characteristic curve can be seen in Fig. 7.4 (a), and its limited linear frequency range is illustrated by the highlighted grey area—typically the upper limit is determined by the maximum current of the V-to-I stage since high frequencies are often desired. A common way to extend the linear region of the V-to-I relationship is accomplished by adding a source-degeneration resistor at the input stage, as shown in Fig. 7.4 (b). The additional resistor linearizes the V-to-I relationship of the pFET by placing it in a source-follower (i.e. common-drain) configuration. A change in voltage at V_{in} results in a linearly proportional change in voltage at the source of the transistor which, in turn, is reflected in a linear change in current through resistor R_s. This linearized current, I_d, is then supplied to the ring oscillator.

Utilizing a linear, source-degenerated input stage on the RO-VCO allows for a linear

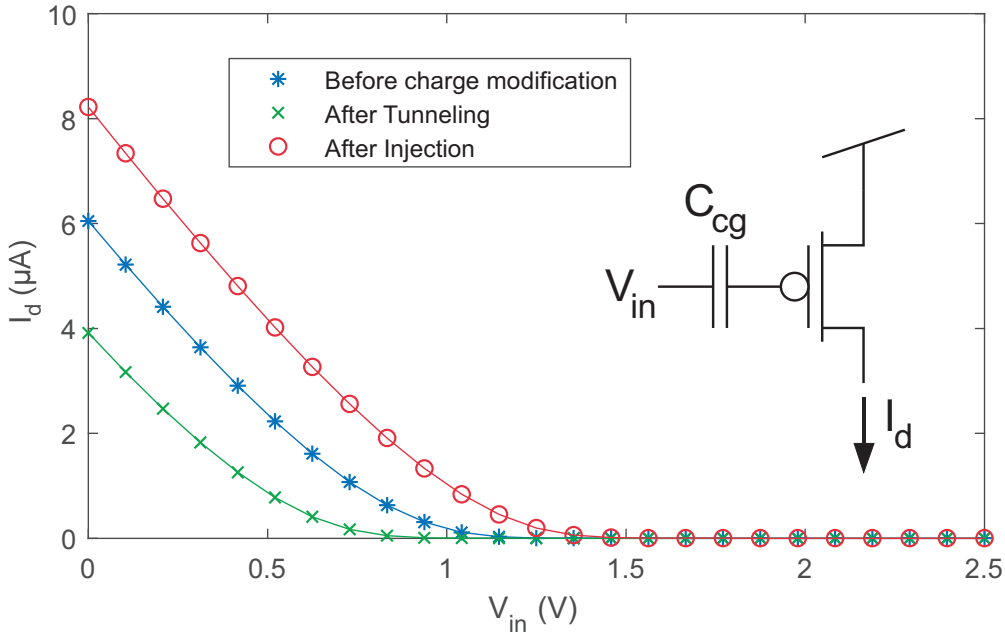


Figure 7.5: FG transistor with measured currents illustrating its programmable threshold characteristics.

V-to-I relationship. However, from Fig. 7.4 (b), it can be seen that the linear region would be extended to the saturation limit of the transistor if V_{in} were allowed to go below ground and take on negative voltages. Therefore, to increase the linear range of the system further, the additional linear region below ground must be accessed. If we were somehow able to shift the threshold voltage, V_T, of the pFET towards the top voltage rail, we could move the saturation limit closer to the y-intercept, allowing for access to the unused linear region. This shift in threshold voltage can be accomplished by using a floating-gate transistor.

The basic structure of an FG transistor is a standard MOSFET with a capacitor connected in series with the gate, as shown in Fig. 7.5 and with a similar cross section as shown in [100]. The capacitor causes the gate of the transistor to be "floating," with no DC path to ground. Fig. 7.5 shows that charge can be added to or removed from the FG, resulting in a memory cell—or in the case of analog applications, a programmable threshold-voltage device [101,102]. Two different processes are used to add or remove precise amounts of charge from an FG. Hot-electron injection is used to add electrons, and Fowler-Nordheim tunneling is used to remove electrons [103,104]. FG transistors have previously been used in RO-VCOs as a memory element to set a programmable low-frequency (around 1Hz) oscillator [105], but

here we use an FG transistor for extending the linear frequency range of a VCO. FG transistors have also been used to linearize circuits such as OTA-based filters and variable-gain amplifiers [100, 106]. To program the FG up to the desired voltage V_{fg}, the injection process is implemented. Additional information on FG devices and the programming process can be found in Chapter 2.3.

Fig. 7.4 (c) shows the effects of using an FG transistor as the input stage (without source degeneration). The threshold voltage of the transistor is shifted towards the top voltage rail; accordingly, both the available frequency range and the linear region can be expanded to the saturation limit by accessing the operating region that was previously unavailable. When this FG method is applied to an input stage along with the linearizing effects of source degeneration, as shown in Fig. 7.4 (d), the maximum linear current range is achieved. The FG allows for the full linear range (part of which was previously below ground) to be accessed after programming.

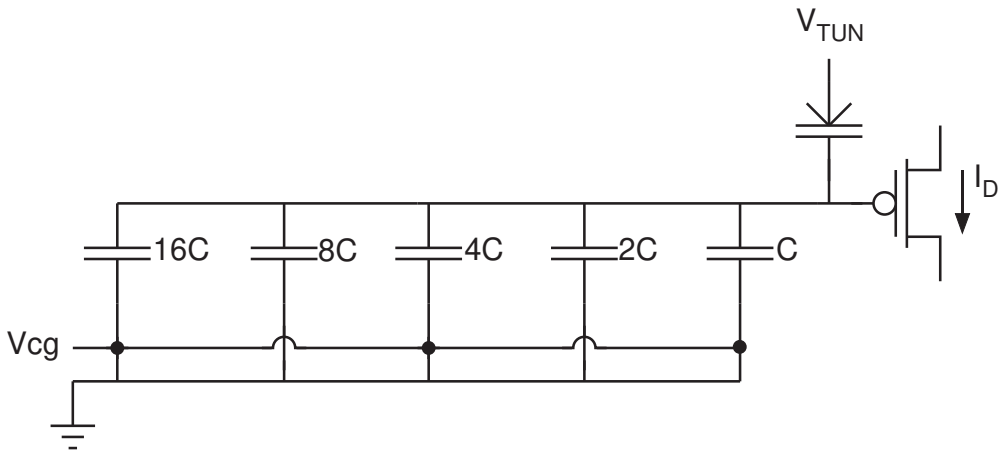


Figure 7.6: Schematic of the capacitive divider that is used in place of standard control gate. The design is shown in a "10101" binary configuration.

7.2.2 Capacitive Control-Gate Division

Additionally, the FG transistor provides a simple method to achieve rail-to-rail input range by using a capacitive divider from V_{in} to V_{fg}. The segmented control-gate capacitor, as shown in Fig. 7.4 (e), allows the linear input range to be spread out over the full input range of the system. The capacitors C_{cg} and C_{cgl} are sized based on the input to the RO-VCO

to match the input signal with the voltage rails of the system.

In most FG devices, it is assumed that the control gate capacitance C_{cg} is much larger than the parasitic capacitances seen on V_{fg} and will dominate the relationship. A capacitive divider can be implemented, segmenting the FG capacitor. The segmented control gate capacitor shown in Figure 7.14 allows the linear input range (already increased with source degeneration and FG programming) to be expanded for the full input range of the system as shown in Figure 7.4 (f). The capacitor C_{cgl} is sized based on the input to the ADC and matches the input with the voltage rails of the system. The capacitive divider, along with FG programming, allows for above or below rail input ranges.

A capacitive divider utilizes the same methods as a typical resistor based voltage divider. A voltage divider produces an output voltage that is a desired fraction of the input voltage. This is a result of distributing the input voltage among the devices in the divider. For the capacitive divider implemented in this architecture, a binary weighted scheme has been used. The Control Gate capacitor is fragmented into five different capacitors. The top plate of each capacitor is connected to the floating node of the FG transistor. The bottom plate of each capacitor can be connected to either the voltage input or to ground.

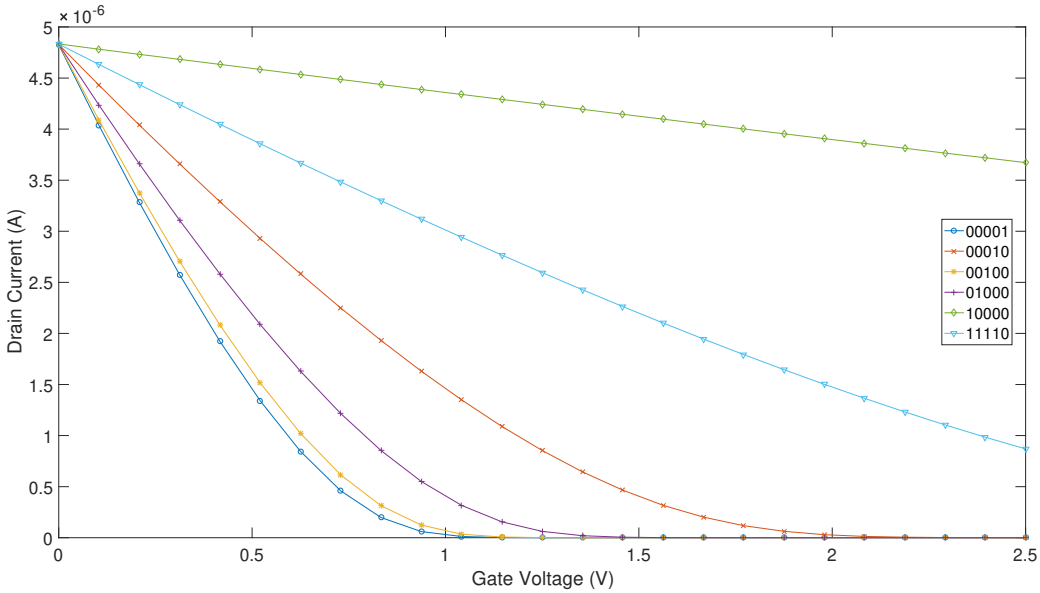


Figure 7.7: Gate sweep of the floating-gate detailing a sample of possible configurations for the capacitively divided variable control gate.

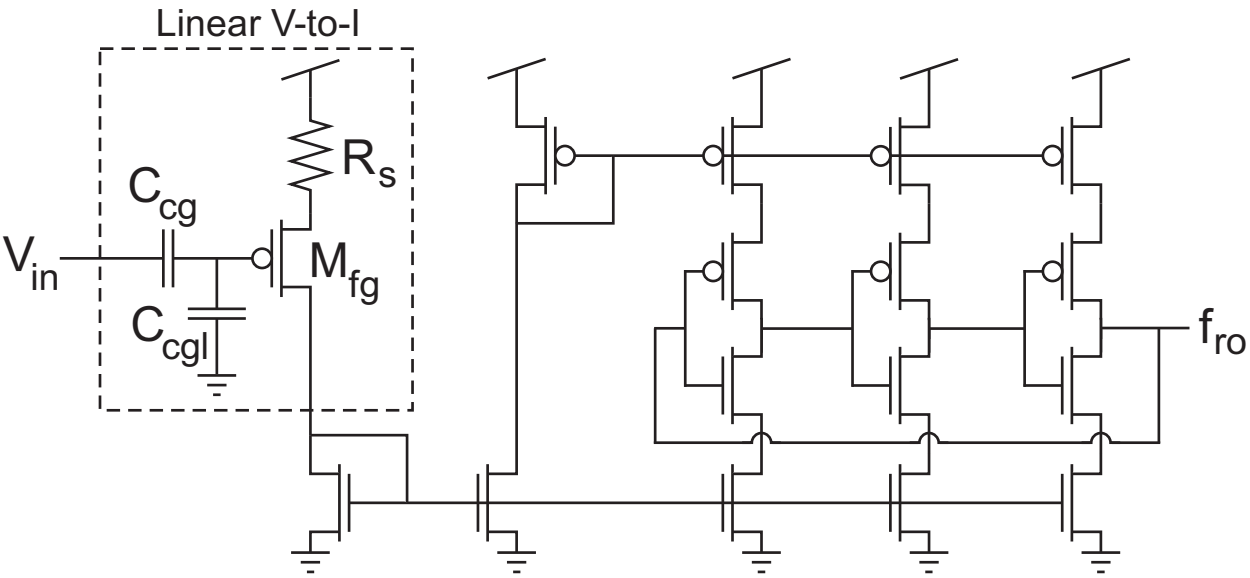


Figure 7.8: Schematic of the proposed RO-VCO with FG linearization.

Figure 7.6 shows the capacitive divider in a "10101" configuration. The bottom plate of each capacitor is connected to a separate pin. Figure 7.7 shows a set of example ranges possible with the capacitive divider. 31 different configurations are possible (based on the 5-bit capacitive divider) to achieve the desired expansion. A minimum of one capacitor must be connected to the input to act as the control gate (hence why there are 31 configurations and not 32). Additional capacitors connected to the gate add their capacitance proportionally as the capacitors connected are seen in parallel. The capacitors that are not connected to the control gate are connected to ground to establish the capacitive divider.

Table 7.1: FG RO-VCO Device Sizes

Device	**Size**
FG Transistor, M_{fg}	$2\mu m \times 1\mu m$
Current Mirror Transistors	$2\mu m \times 1.5\mu m$
Inverter - nFET and pFET	$700nm \times 2\mu m$ and $2\mu m \times 2\mu m$
Source Degeneration Resistor, R_s	$20k\Omega$
Control Gate Capacitor, C_{cg}	900fF
Control Gate Divider, C_{cgl}	2.2pF

The full RO-VCO with FG linearization is shown in Fig. 7.8. It is based on a standard ring-oscillator topology. In the implementation presented in this brief, 3 inverter stages are

used. The input transistor has been modified for source degeneration, FG programming, and capacitive division.

7.2.3 RO-VCO Performance

The FG RO-VCO described above was fabricated in a standard 0.35μm CMOS process that did not have any process add-ons. Device sizes are given in Table 7.1. The FG RO-VCO was fabricated so that it could be operated as a standalone circuit or connected to a digital counter to act as a VCO-ADC, as will be discussed in the next section. A die photograph of the FG RO-VCO and VCO-ADC system can be seen in Fig. 7.10. The FG RO-VCO occupies an area of 120μm $\times$ 130μm, and all modules needed to implement the VCO-ADC occupy an area of 960μm $\times$ 150μm. All results have been measured directly from the fabricated integrated circuit.

To facilitate the V_T shift of the FG transistor, pulse-based injection similar to [107] was used to precisely program the FG charge. Prior to programming, the FG transistor was isolated by using multiplexers at the source and drain. The FG transistor was then programmed such that the edge of saturation occurs when V_{cg}=0V, as illustrated in Fig. 7.4 (c), which is equivalent to having V_{fg} be one V_T below ground when V_{cg}=0V. When reconnected to the circuit, the source degeneration linearizes the V-to-I relationship of the FG transistor. The capacitive divider ratio was chosen to map the resulting linear V-to-I range to rail-to-rail operation. In our exploratory design, we employed a digitally adjustable capacitive divider where the total size of C_{cgl} was adjustable via multiplexing various capacitor sizes to allow us to adjust the capacitive divider at run-time; future versions would have this ratio set at design time.

The relationship between the input voltage and the output frequency is the main factor in system performance. Fig. 7.9 (a) shows the output frequency for two RO-VCOs using source degeneration. Additionally, Fig. 7.9 (b) shows the frequency error measured with respect to an ideal linear V-to-f relationship. The "Standard RO-VCO" case refers to an RO-VCO with only source degeneration applied to the input transistor (i.e. using Fig. 7.4 (b) as the input stage). The "FG RO-VCO" case refers to the FG RO-VCO with source

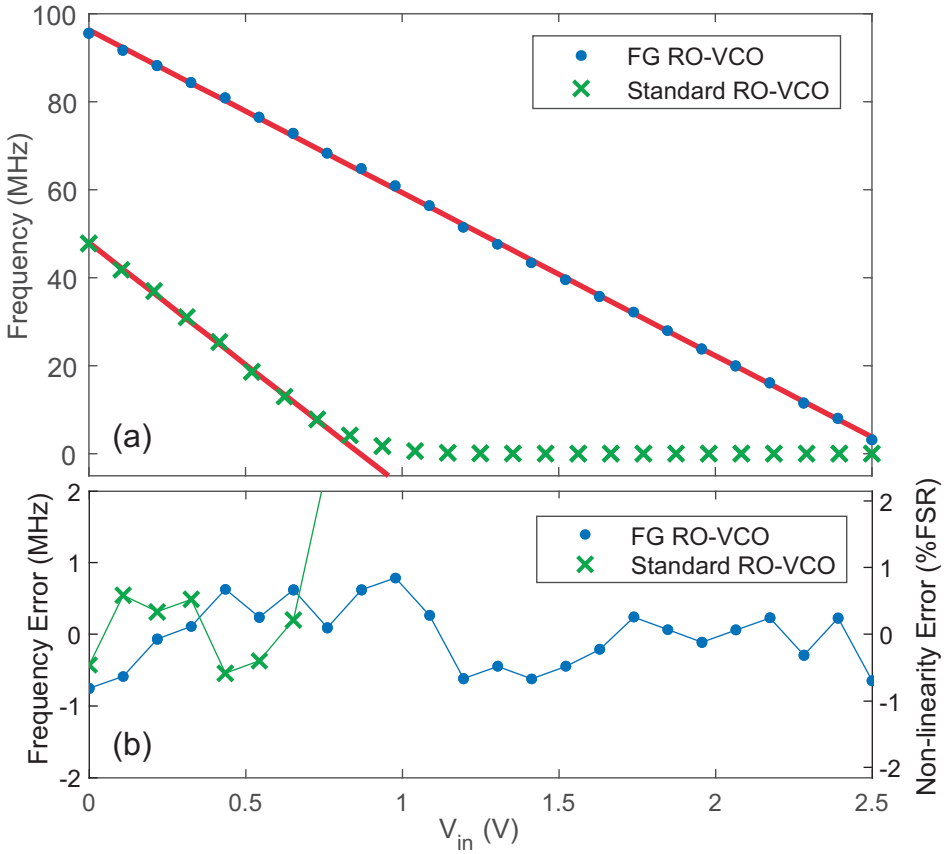


Figure 7.9: (a) Measured output frequency of the VCO, comparing the standard RO-VCO to the FG-based version. (b) Measured frequency error.

degeneration, FG programming, and capacitive division (i.e. Fig. 7.8).

As can be seen from Fig. 7.9, implementation of the FG significantly extends the linear frequency range by increasing the range of currents available from the input stage. Both RO-VCOs have a minimum frequency of 4MHz, and FG linearization extends the linear frequency range to 97MHz, which is 2.33 times the standard case that only reaches 44MHz. In addition to extending the frequency range, the FG linearization also ensures that the V-to-f relationship stays linear, with non-linearity error staying $< 1\%$. Furthermore, the capacitive division of FG linearization provides an easy method to map the output linear frequency range to the full rail-to-rail input range (increased from an input range of 0V–0.7V for the standard case), as is demonstrated in Fig. 7.9.

Table 7.2 shows a comparison between this work and other highly linear RO-VCOs. Our

Table 7.2: Comparison to State-of-the-Art RO-VCOs

	This Work	**[85]**	**[86]**	**[88]**	**[89]**
Technology (nm)	350	180	65	180	65
Power Supply (V)	2.5	1.8	1	1.8	1
Number of Stages	3	7	18	5	8
Input Voltage Range (V)	Rail-to-Rail	0 – 1	Rail-to-Rail	Rail-to-Rail	Rail-to-Rail
Linear Frequency Range (MHz)	4 – 97	48 – 143	100 – 500	573 – 852.8	4000 – 5000
Non-linearity Error (% of FSR)	0.84	0.12	0.6	0.09	0.5
Percent Change in Frequency (%)	2325	197.9	400	48.8	25
Figure of Merit (FoM)	$3.6\text{x}10^{-4}$	$6.06\text{x}10^{-4}$	$1.5\text{x}10^{-3}$	$1.84\text{x}10^{-3}$	$2\text{x}10^{-2}$

design's benefits are apparent when considering the proposed figure of merit (FoM):

$$FoM = \frac{Non\text{-}linearity\ Error\ (\%\ of\ FSR)}{Percent\ Change\ in\ Frequency\ (\%)} \tag{7.1}$$

This FoM highlights the balance between non-linearity error (i.e. deviation from an ideal straight line) and frequency tuning range, with a smaller number indicating better performance. While the other highly linear RO-VCOs report better linearity, our design has the unique benefit of increasing frequency range after linearization. Due to the increased frequency range, our design results in the largest percent change of frequency and the best FoM. The other designs are from 180nm or smaller technology nodes, which allow for higher frequencies and lower power consumption for RO-VCOs. Our design could be translated to a smaller technology node given the availability of I/O devices with oxide thicknesses $\geq$5nm for FG transistor implementation [108]; such 2.5V and/or 3.3V I/O devices are common in many more-advanced technology nodes. Table 7.2 shows that our FG RO-VCO and linearization technique is competitive with the state-of-the-art, and, given a smaller process, could potentially improve on existing designs to produce highly linear RO-VCOs with wide frequency ranges.

FG transistors have identical noise and PVT (process, voltage, temperature) characteris-

tics as standard MOSFETs [109]. Since both the Standard and FG RO-VCOs of Fig. 7.9 are comprised of identical circuitry, with the only difference being the FG transistor replacing a standard transistor, both VCOs will have identical phase noise and PVT considerations [110]. One advantage, however, of using an FG transistor is that it can be used to calibrate out the effects of process variation after fabrication—akin to analog trimming [111]. Additionally, we have previously demonstrated a technique using FG transistors to help mitigate the effects of temperature variation on a ring oscillator [112], and that technique could potentially be combined with this linearizing technique for further improved performance. One additional consideration with regard to phase noise is the size of the FG transistor—care should be taken to ensure that the FG device is long enough that the effects of velocity saturation (and increases in phase noise due to operation in that region) are minimized. Because the effective V_T of the FG transistor is shifted to access higher currents where velocity saturation can be more prominent, unforeseen consequences of velocity saturation can occur if not carefully considered at design time. In our design, we chose a device length to minimize velocity saturation while still maintaining a relatively short channel length.

The VCO was operated at a reduced supply voltage of V_{dd}=2.5V instead of the rated 3.3V in order to prevent unwanted injection on the FG which would change the programmed charge [113]. Because of the reduced V_{dd}, even when V_{fg} goes negative due to capacitive coupling from V_{in} to V_{fg}, V_{fg} will not be low enough to cause any long-term device degradation—no oxide or junction voltage ever exceeds the rated values for these devices. Additionally, once programmed, FG transistors have been shown to have high levels of charge retention, exhibiting drift in the microvolt range or less over a 10-year period [111,114], so there should be no appreciable drift in the performance of the circuit due to charge loss.

7.3 The VCO-ADC

To show the benefits of linearity coupled with an increase in frequency range, the FG RO-VCO was demonstrated in an ADC application as a VCO-ADC.

The main benefit of a VCO-ADC derives from its digital design. With a highly digital design, the system will scale well with power consumption as well as resolution. The ability

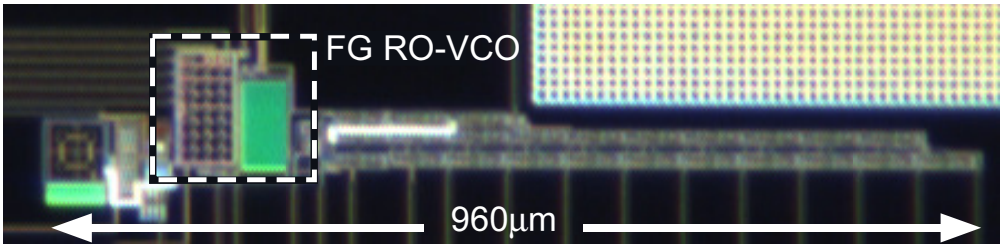


Figure 7.10: Die photograph of the FG RO-VCO and VCO-ADC system fabricated in a standard 0.35μm CMOS process.

for this ADC system to scale with power supplies earns it a mention when discussing ADCs, not just as stand alone devices but also as the quantization stage in a Sigma Delta ADC [115, 116].

To create the most basic form of a VCO-ADC, an RO-VCO output can be connected directly to a digital counter. The output digital word of the counter is proportional to the input voltage if the counter is reset at a regular interval by a global clock. In such a VCO-ADC, which is shown in Fig. 7.10, the dynamic range is directly related to the range of frequencies of the VCO. The global clock should be approximately equal to the minimum frequency of the VCO—just enough time for the least-significant bit of the counter to flip once. Assuming an arbitrary word length for the counter, the maximum frequency of the VCO sets the dynamic range of the VCO-ADC by the maximum digital word that can be counted during one clock cycle. Accordingly, the larger the range of frequencies that can be accommodated by the VCO, the higher the number that can be counted, resulting in a larger dynamic range. However, the V-to-f range must be linear so that the overall ADC's linearity will not suffer. Linearity can be measured through metrics such as its Spurious-Free-Dynamic-Range (SFDR) and Signal-to-Noise-and-Distortion-Ratio (SINAD).

Figure 7.11 (a) shows a general block diagram for a VCO-ADC. Operation begins with the RO-VCO converting the input seen at V_{in} to a square wave of proportional frequency. The produced oscillation is then fed into a digital counter. The output of the counter is then proportional to the input voltage if the counter is periodically reset. The global clock used to reset the counter is independent of the RO-VCO output oscillations and is what determines the VCO-ADC's sampling rate. The global clock sets a consistent time window for the counter to measure oscillations from the RO-VCO, resulting in a digital word that

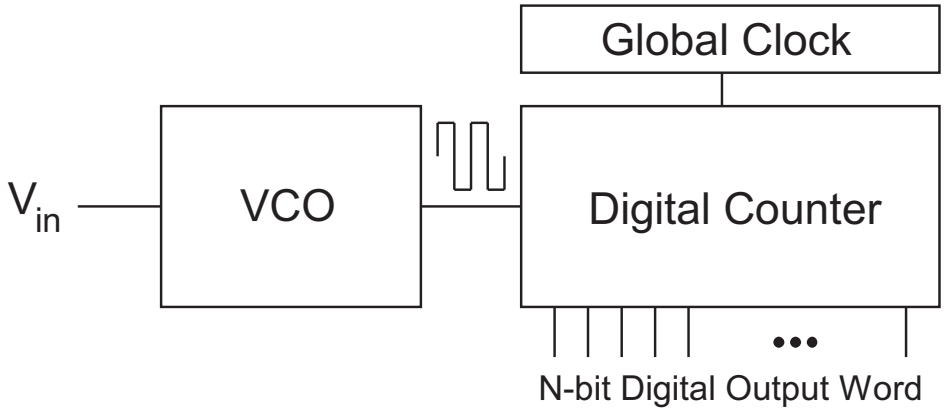


Figure 7.11: Block diagram of standard VCO-ADCs.

can be correlated back to the input voltage. The VCO-ADC is designed to scale with resolution. To increase the resolution, all that needs to be added is additional cells to the end of the counter, allowing for a larger digital word to be converted [117]. However, without making any changes to the VCO, adding additional cells to the counter will only increase the resolution if the sampling rate is decreased.

7.3.1 Architecture

A digital counter is an array of memory elements, usually Flip-Flops (FF), where the output of each FF is connected to the data input of the next FF in the chain. The digital counter implemented in this design acts as a frequency divider and the main component of the data conversion. A global clock is used to reset the counter, allowing it to begin a new conversion. In this case, the output needs to be read in between the global clock pulses to ensure no data is lost.

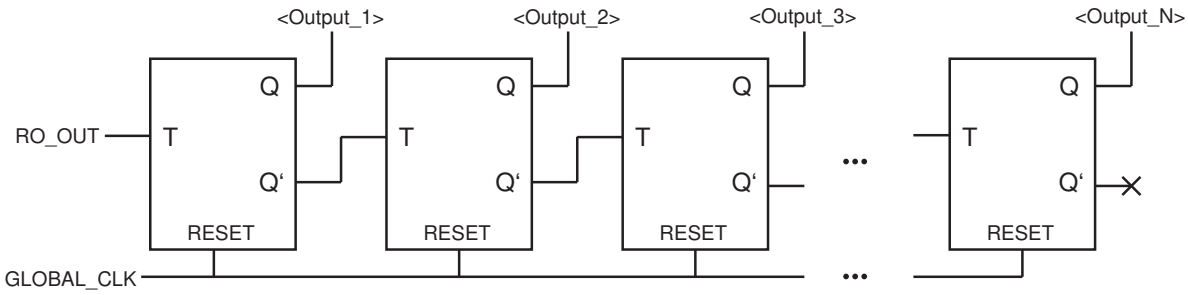


Figure 7.12: Digital counter utilizing T-Flip-Flops

A counter typically operates using "T" type flip flops as the latching element within the chain. Figure 7.12 shows a counter acting as a frequency divider that utilizes T-Flip-Flops (TFF) as the memory elements. The schematic is shown with reference to the RO-VCO output (input of the counter) and a global clocked reset. The number "N" of FFs in the chain sets the resolution of the output word for the VCO-ADC. Adding a FF to the end of the chain increases the number of bits of the output word by one for each FF added.

The inverted output of each TFF is used as the clock signal for the next TFF in the chain to create a frequency divider. This results in each stage of the counter clocking at a frequency that is one half of the clocking frequency of the previous stage. The first TFF is clocked by the output of the RO-VCO and the final TFF terminates in a no-connect (NC). Counting begins by resetting all of the TFFs in the chain to a state of a low output using the global clock. For this design, the global clock is supplied externally by an Agilent 33220A Waveform Generator. The waveform generator supplies a pulse signal with a 0.1% duty cycle. After resetting on the rising edge of the global clock, the counter begins its frequency division, starting from the RO-VCO output on the first TFF. From this configuration, the resulting digital word increases from Least Significant Bit (LSB) to Most Significant Bit (MSB) from the first TFF after the RO-VCO to the TFF that is farthest down the counter.

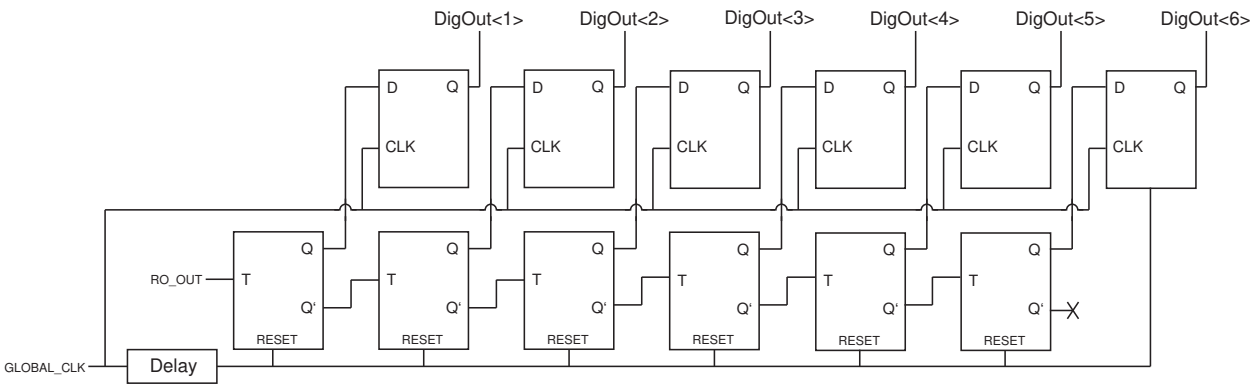


Figure 7.13: Digital counter and output memory buffer schematic.

With regards to the period of the global clock, the final count is acquired at the end of the period, right as the global clock goes high to start the next period. This creates a problem as the digital output of the counter needs to be read right as it is reset. To fix this issue, a second set of memory elements were added utilizing D-Flip-Flops (DFF). This

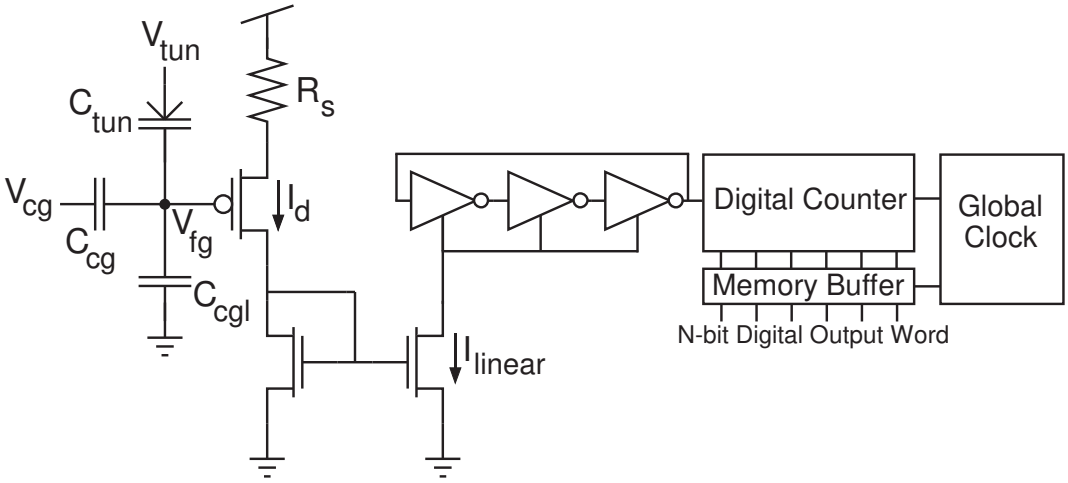


Figure 7.14: Schematic of the full VCO-ADC presented in this work.

second set of flip-flops act as a memory buffer to hold the digital word for reading while the counter is reset. This is accomplished as shown in Figure 7.13. The non-inverted, and previously unused output of each TFF in the counter is connected to the input of one of the DFFs in the memory buffer.

To ensure the memory buffer receives the TDC output before it is reset, the clock terminal of each DFF is connected directly to the global clock and are rising-edge triggered. The global clock connects to a delay element of TFFs before connecting to the RESET pin of the TFFs in the counter. This ensures that the memory buffer will take the final TDC measurement before the counter is reset. The memory buffer will hold this value for one period of the global clock before being set to the counted value for the next period. This operation sets the sampling rate of the entire VCO-ADC. The memory buffer must be read at least once during the period of the global clock to ensure no conversions are lost. By reading the ADC on the falling edge of the global reset, it is ensured that each conversion is measured. This shows that the global clock acts as the sampling frequency of this ADC.

7.3.2 Quantization Error

The VCO-ADC described above was fabricated in a 0.35µm CMOS process. The system schematic can be seen in Figure 7.14. A die-photo of the VCO-ADC can be seen in Figure 7.10. All modules from the design occupy an area of 960µm x 150µm. All results have been

measured directly from the fabricated integrated circuit.

The first metric that is used to measure the performance of an ADC begins with generating a transfer curve based on the digital output word. A slow rail-to-rail ramp voltage is applied to the input of the ADC and the resulting digital words are recorded for each conversion sample. The ideal transfer function appears as a step response, with each "step" corresponding to a one LSB change from the previous step. Figure 7.15 shows the reconstructed signal from the digital word output for two cases. (a) shows the quantized signal without any linearization methods implemented. (b) shows the quantized signal after the FG is programmed, placed in a source-follower configuration, and the capacitive divider is implemented.

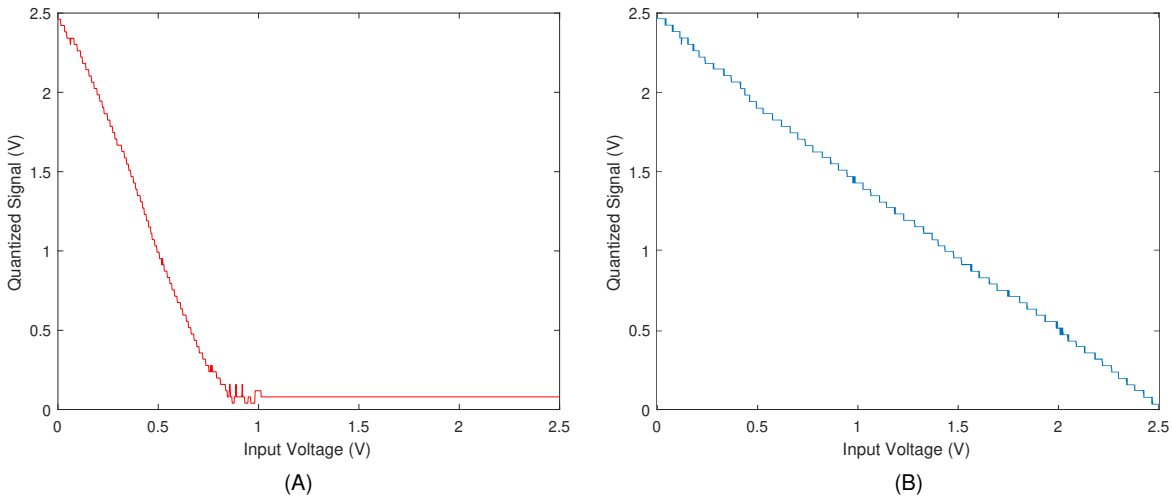


Figure 7.15: VCOADC output showing full signal quantization. Logic analyzer used to reconstruct the original signal from the digital memory buffer. (a) Quantized signal without any linearization techniques implemented. (b) After linearizing the voltage-to-frequency relationship.

The logic analyzer on a Digilent Analog Discovery 2 was utilized to generate the quantized signal from the memory buffer output. The quantized signal has been left in the form measured by the logic analyzer. However, if used in a practical setting, the logic would be "flipped" to account for the inverse relationship of the pFET device. The resulting signal in 7.15 (b) still shows signs of offset error, integral noninearity (INL), and differential nonlinearity (DNL). These issues can be attributed to the non-ideal workings of a fabricated

circuit. However, when compared to the nonlinearized case in 7.15 (a), the offset error, INL, and DNL are greatly improved. Given that the nonlinearized case has such poor performance, the following metrics will only be shown for the linearized case. Otherwise, each metric for the linearized case would look "ideal" when compared. To further illustrate the INL of the quantized signal, the reconstructed signal is compared to the actual input to the ADC. By subtracting the difference between the quantized signal and the input signal, the quantization error can be found.

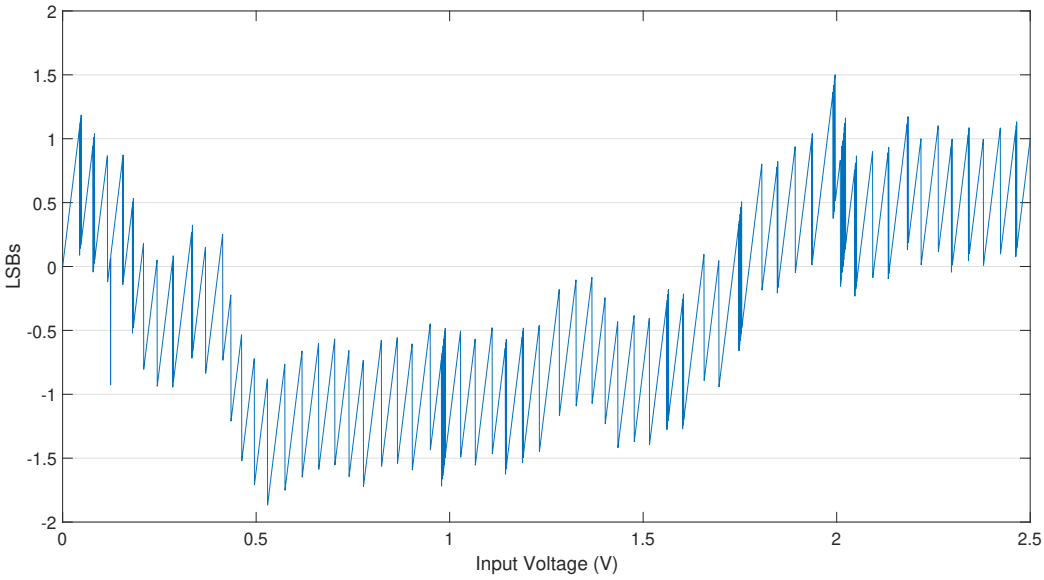


Figure 7.16: Quantization error of the reconstructed signal. The integral nonlinearity is reported as the maximum INL for the full input range. The INL is derived from this plot to be -0.87 LSBs.

The ideal quantization error should oscillate evenly between -0.5 and 0.5 LSBs. Figure 7.16 shows where the ADC's output deviates from this standard. Any value above 0.5 or below -0.5 LSBs results in an INL. In the case of an ideal ADC, the step width of each LSB change will be uniform for the entire transfer curve. Any deviation from the ideal step width is seen as a DNL. As the quantization error shown is derived by subtracting the quantized signal from the input, any error higher than 0.5 or lower than -0.5 shows that a particular step does not match the ideal case. The integral nonlinearity of the ADC is seen as the maximum DNL reported by the quantization error. The INL for this design and current

setup is reported to be -0.86 LSBs.

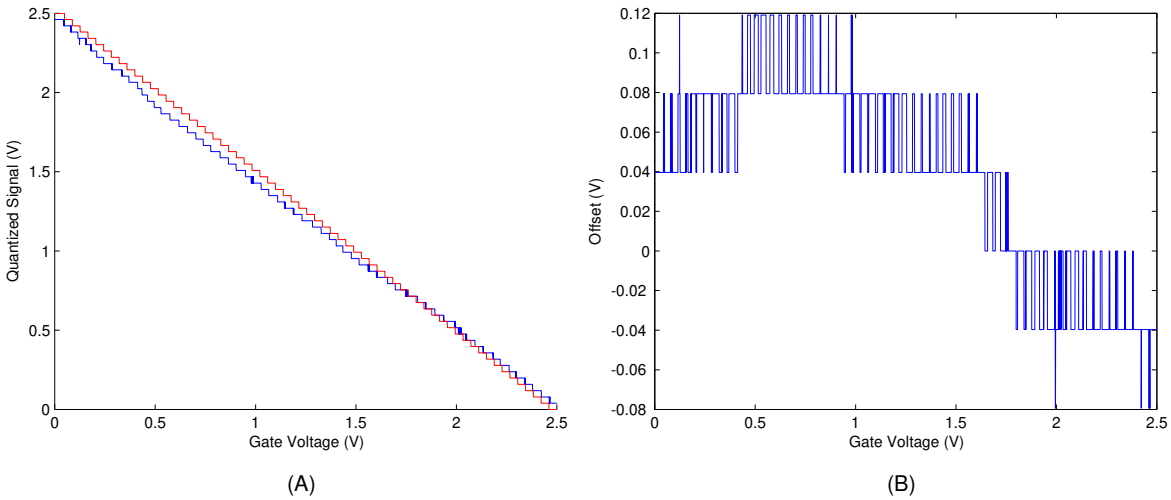


Figure 7.17: Comparing the quantized output of the designed VCOADC to an ideal ADC output.

While the DNL and INL show the deviations from an ideal ADC, it is also important to note the offset error. For offset, the quantized signal is compared to an ideal quantized signal instead of the actual input. Figure 7.17 details how the VCO-ADCs output deviates from an ideal ADC output.

7.3.3 Frequency Response

The relationship between input voltage (in our case, the control gate voltage) and output frequency is the main factor in system performance. As discussed in Section 7.2, our linearization method improves upon the basic VCO-ADC archtitecture by adding an FG and capacitive divider to the input transistor. Figure 7.9 shows the frequency tuning curves for both cases of the RO-VCO topology. The "nonlinearized" case refers to a VCO-ADC with only source degeneration applied to the input transistor, to show a base case for comparison. The "Linearized" case referes to the VCO-ADC with FG programming and capactive division implemented.

The frequency response of an ADC is vital in describing its performance with regard to linearity, and will be shown with respect to an 8-bit configuration. An ADCs frequency

response is generated with an input sine wave and shows the ratio of the signal power to the total noise and harmonic power at the output of the system. The sine wave input should be around one hundred times slower than the sampling frequency of the ADC (for this case, the sampling rate is 4MHz) to allow for adequate conversion during the period of the wave. It is crucial that the sine wave is not an integer multiple of the sampling frequency, as it will cause additional harmonics within the system.

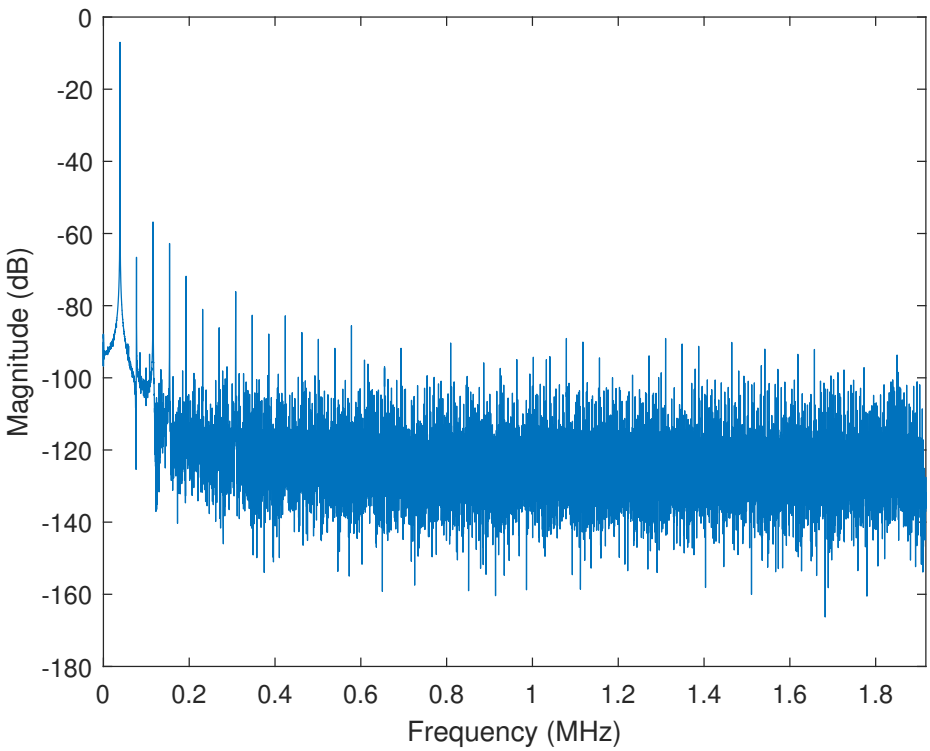


Figure 7.18: Frequency response of the VCO-ADC with a 40kHz sine wave input. The Spurious-Free-Dynamic-Range is found from this plot and calculated to be 52.67dB

The output of the system is recorded for multiple cycles of the input sine wave. Figure 7.18 shows the Fourier transform of the system output. From Figure 7.18, the Spurious-Free-Dynamic-Range (SFDR) can be calculated. The SFDR is seen as the difference in decibels between the amplitude of the fundamental and the largest harmonic spike. The SFDR for our system was calculated to be 52.67dB (compared to 19.86 dB before FG linearization). Using the other measurable harmonic spikes, we are able to calculate the Signal-to-Noise-and-Distortion-Ratio (SINAD) to be 46.4dB (compared to 17.31dB before FG linearization).

Table 7.3: Improvements Made by FG Linearization to a VCO-ADC

	Std RO-VCO	**FG RO-VCO**	**Improved**
Input Range (V)	0 to 0.7	Rail-to-Rail	1.8
Freq. Range (MHz)	4–44	4–97	53
ENOB (bits)	2.6	7.74	5.48
SFDR (dB)	19.86	52.67	32.81
SINAD (dB)	17.31	48.4	31.09

Finally, we are able to calculate the ENOB for our VCO-ADC. The ENOB measurement is the best representation of an ADC's practical performance as it highlights the systems dynamic range. Utilizing the SINAD value, we are able to calculate the ENOB of the ADC directly. This results in an ENOB of 7.74 bits (compared to 2.6 before FG linearization).

Table 7.3 shows the measured improvement in performance by using the FG linearization technique. The "Std" and "FG" versions are the two VCOs of Fig. 7.9 with the same biasing conditions. To ensure a fair comparison, both VCO-ADCs have the same sampling frequency of 4 MHz, as well as identically sized transistors and resistances.

The comparison of Table 7.3 shows that our FG linearization technique increases all metrics of the VCO-ADC. The dynamic range, as measured by the effective number of bits (ENOB), increases by nearly 5.5 bits. The linearity metrics of SFDR and SINAD have also both increased >30dB due to both the increased frequency range and the linearizing effects of the V-to-I FG stage. By combining our linearization technique with more advanced VCO-ADC techniques such as multi-phase readout, differential ring oscillators, or other new RO-VCO topologies (e.g. [86, 118]), a highly linear VCO-ADC could be achieved. Additionally, our linearization technique could be combined with other VCO linearization techniques for even further improvements.

7.4 Conclusion

A linearization method for voltage-controlled ring oscillators was presented. We have successfully shown that a floating-gate transistor can be used along with capacitive division and source degeneration to design an FG RO-VCO with a large linear frequency range.

The FG RO-VCO was demonstrated in a VCO-ADC to show how its increased linear range could be leveraged to improve ADC performance. The VCO-ADC not only operates with reasonable linearity (SFDR and SINAD), but also without sacrificing its ENOB. While there are other linearization techniques that create a highly linear region of operation, our design does so while increasing rather than decreasing the frequency range.

Chapter 8

Conclusions

Analog signal processing (ASP) has been shown in industry and this dissertation to be a viable and preferable alternative to always-on digital signal processing. By keeping high power digital components asleep while a front-end ASP monitors for events, resource-constrained systems can increase battery lifetime without sacrificing data integrity. However, as has been discussed in this work, front end ASP has typically required expensive, time-consuming design of application-specific integrated circuit (ASIC) systems to adequately monitor event detection and classification.

The floating-gate (FG) transistor with its use as nonvolatile analog memory and biasing, mixed with the field-programmable analog array (FPAA) has revolutionized reconfigurable analog circuitry. Tasks that previously required ASICs can now be approached and solved through reconfigurable analog systems such as the Reconfigurable Analog and Mixed-Signal Platform (RAMP).

The RAMP and other FPAAs have changed the way we can approach the ASP front end. Pairing the computational power of current FPAAs like the RAMP 1.1 with the ability to account for environmental effects has been presented to show that these devices are quickly evolving out of the laboratory settings. The FG transistor acting as the core element of the FPAA can be protected and adjusted so that external effects can be negated, allowing for precise in-the-field operation.

8.1 Summary of Results

With the implementation of a temperature-compensation system as described in Chapter 3, biasing from FG transistors can be kept constant across a range of environments. With the precise biasing required by sensitive analog circuits, the proposed scheme allows for the FG transistors to perform as expected without suffering from the effects of fluctuating temperature.

Accounting for the effects of mobile ionic charge as described in Chapter 4 helps push these devices out of laboratory environments while maintaining high accuracy from the system's core FG transistors. When deployed in the field, FPAAs will be exposed to not only temperature fluctuations, but also increased levels of humidity not found in the laboratory. Given that the effects of humidity have been shown in this work to directly contribute to the presence of MIC, being able to predict its effect on the FG transistor becomes increasingly important. By adjusting latency within FG startup transients, the FPAA can be reprogrammed accordingly to make sure MIC does not cause measurable drift on the FG transistors. This compensation, paired with the temperature compensation discussed previously allows for reliable operation.

The work described in Chapter 6 shows that a reconfigurable ASP front end can handle the task that required an ASIC as recently as 2011. A task that required a custom IC to be designed and fabricated can be handled by the RAMP or other adequate FPAAs by applying the correct connections list. The inherent computational power of an FPAA system, with appropriate compensation techniques for temperature and humidity has been shown to be a true competitor to ASIC systems within the ASP domain. The presented work has furthered the accessibility and capability of reconfigurable systems, and this significant improvement will pave the way for other ASIC solutions to be handled by reconfigurable FPAAs such as voice activity and other advanced event detection.

To continue improving the ASP chain, an analog-to-digital converter (ADC) was designed that is intended to leverage the capabilities of the RAMP system. The ADC (presented in Chapter 7) implements a voltage-controlled oscillating ADC (VCO-ADC) using parts commonly found on an FPAA. The VCO-ADC implements a novel technique for linearizing and

increasing the voltage-to-frequency relationship within the oscillator via an FG transistor. The VCO-ADC was designed with the ASP front end in mind, making asynchronous conversions possible when necessary. The ADC has practical use as a stand-alone ADC, as well as for implementation within the new version of the RAMP system.

The RAMP 1.1 (the newest version of the RAMP at WVU) will continue to evolve the way front-end ASP is performed, and what is capable from a reconfigurable system. The new micro-controller (MCU) interface for the RAMP 1.1 implements low-energy Bluetooth (BLE) for programming netlists wirelessly and with low power consumption. Custom printed circuit boards have been designed for the RAMP 1.1 and BLE MCU with future-proofing in mind to allow for new MCUs to be utilized as they are released to market. The RAMP 1.1 is an evolving system. With the advancements made from this work, the RAMP will continue to be a power-house in the ASP realm and replace more ASIC solutions with its reconfigurable architecture.

8.2 Future Work

Continuing this work will involve further development of the RAMP 1.1 system and characterization of additional ADCs that work well within the ASP domain. A working programming infrastructure has been presented for the RAMP 1.1. To further the RAMP's signal-processing capabilities, more efficient auto-routing methods need to be developed, and each CAB component needs to be characterized. Without an efficient auto-routing method, the best routing for a synthesized circuit needs to be checked by hand. An auto-routing scheme that ensures the shortest path between CAB elements will help ensure the least amount of parasitic capacitances are introduced from circuit synthesis. If a path between CAB elements takes an unnecessarily long path, additional parasitics will be introduced from the extra switches and from the metal paths. These extra parasitics can degrade circuit performance, especially for sensitive nodes.

The circuits within each CAB need to be characterized for optimal performance. The majority of analog components within the RAMP require the precise biasing of the FG transistors. Accurate modeling and characterization of the FG transistors will lead to better

overall circuit performance. To operate the analog circuits within the CABs, the FG transistors are programmed to a desired level, and a specific current is then produced for biasing. Currently, the FGs are programmed via a voltage value. This voltage value corresponds to the target being used by the programmer. While this method is reliable way for programming the FG transistors, it does not directly correspond to circuit performance. To ensure the RAMP 1.1 is a usable system in realistic scenarios, these voltage targets need to be mapped directly to circuit performance.

Each analog circuit will have some parameter that needs to be mapped to the programming voltage target. Taking the biquad filter as an example (from Chapter 5.1.1), it uses two FG transistors to set bias currents for its upper and lower corner frequencies. Utilizing a voltage target for each corner frequency makes the end-user experience difficult. To make this more accessible, the voltage targets need to be mapped to the corner frequencies. This way, the end-user could supply the desired corner frequencies and the programmer would use the mapping to generate the required voltage targets for the FGs. Another example would be the oscillator in the new unique CAB. The bias is used to set the oscillator output frequency. Being able to map the output frequency to a voltage target would allow the end user to specify a desired frequency needed, and the programmer would generate the required target voltage.

To ensure that the RAMP 1.1's low-power signal-processing capabilities are not wasted, additional development with ADCs needs to be done. There is no getting around using the ADC to digitize data to be read by the MCU. If we want to use the RAMP as an event detector, at some point, the event itself will need to be passed from the analog to the digital domain. If the ADC that is required for this task has high latency or consumes too much power, the savings afforded by the RAMP 1.1 become less desirable over that of a pure digital implementation. Additionally, the insight gained in Chapter 4 will be used to develop an algorithm for reprogramming the RAMP once the startup transients due to MIC have settled, allowing for long-term accurate performance.

The VCO-ADC presented in Chapter 7 makes use of the components on the RAMP and commonly found on other FPAAs. This ADC is desirable for its scaling capabilities and low overhead. However, an asynchronous ADC, such as the one presented in Appendix A, would

give additional capabilities to the RAMP system. The ability to asynchronously convert data with low latency would make extrema sampling (common in event detection) a lower-power-consuming task. The ADC described in the appendix also makes use of components found on the RAMP. By synthesizing and chracterizing the ADC from Appendix A on the RAMP 1.1, it would add an additional strength to the RAMP's already impressive ASP toolkit.

References

[1] B. Rumberg, D. Graham, V. Kulathumani, and R. Fernandez, "Hibernets: Energy-efficient sensor networks using analog signal processing," *IEEE Journal of Emerging and Selected Topics in Circuits and Systems*, vol. 1, no. 3, pp. 321–334, Sept. 2011.

[2] M. Mathew, "The perennial role of analog electronics," February 2020. [Online]. Available: https://www.eletimes.com/the-perennial-role-of-analog-electronics/ "

[3] S. D. Kamath and D. V. Anderson, "Signal processing in digital and floating-gate analog circuits; design trade-offs," in *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*, vol. 2, 2003, pp. 1483–1487 Vol.2.

[4] Z. Charbiwala, P. Martin, and M. B. Srivastava, "Capmux: A scalable analog front end for low power compressed sensing," in *International Green Computing Conference*, 2012, pp. 1–10.

[5] C. Twigg and P. Hasler, "A large-scale reconfigurable analog signal processor (RASP) IC," in *IEEE Custom Integrated Circuits Conference*, Sept 2006, pp. 5–8.

[6] B. Rumberg, D. Graham, and V. Kulathumani, "A low-power, programmable analog event detector for resource-constrained sensing systems," in *IEEE International Midwest Symposium on Circuits and Systems*, Aug 2012, pp. 338–341.

[7] D. Kahng and S. M. Sze, "A floating gate and its application to memory devices," *The Bell System Technical Journal*, vol. 46, no. 6, pp. 1288–1295, 1967.

[8] D. Wen and P. Salsbury, "Analysis and design of a single-state floating gate amplifier," in *1973 IEEE International Solid-State Circuits Conference. Digest of Technical Papers*, vol. XVI, 1973, pp. 154–155.

[9] D. D. Wen, "Design and operation of a floating gate amplifier," *IEEE Journal of Solid-State Circuits*, vol. 9, no. 6, pp. 410–414, 1974.

[10] E. Sackinger and W. Guggenbuhl, "An analog trimming circuit based on a floating-gate device," *IEEE Journal of Solid-State Circuits*, vol. 23, no. 6, pp. 1437–1440, 1988.

[11] P. Hasler, B. A. Minch, and C. Diorio, "Adaptive circuits using pfet floating-gate devices," in *Proceedings 20th Anniversary Conference on Advanced Research in VLSI*, 1999, pp. 215–229.

[12] T. S. Hall, P. Hasler, and D. V. Anderson, "Field-programmable analog arrays: a floating—gate approach," in *International Conference on Field Programmable Logic and Applications, 2002. Proceedings.*, 2002, pp. 142–.

[13] E. Mackensen and C. Muller, "Implementation of reconfigurable micro-sensor interfaces utilizing FPAAs," in *IEEE Sensors*, Nov. 2005, pp. 1064–1067.

[14] S. Ravindran, P. Smith, D. Graham, V. Duangudom, D. Anderson, and P. Hasler, "Towards low-power on-chip auditory processing," *EURASIP J. Applied Sig. Process.*, vol. 2005, no. 7, pp. 1082–1092, May 2005.

[15] B. Rumberg, D. Graham, S. Clites, B. Kelly, M. Navidi, A. Dilello, and V. Kulathumani, "RAMP: Accelerating wireless sensor hardware design with a reconfigurable analog/mixed-signal platform," in *Proc. of ACM/IEEE Inform. Process. in Sensor Networks*, 2015, pp. 47–58.

[16] A. Basu, S. Brink, C. Schlottmann, S. Ramakrishnan, C. Petre, S. Koziol, F. Baskaya, C. M. Twigg, and P. Hasler, "A floating-gate-based field-programmable analog array," *IEEE J Solid-State Circuits*, vol. 45, pp. 1781–1794, 2010.

[17] E. Gutierrez, C. Perez, F. Hernandez, and L. Hernandez, "Vco-based feature extraction architecture for low power speech recognition applications," in *2019 IEEE 62nd International Midwest Symposium on Circuits and Systems*, Aug 2019, pp. 1175–1178.

[18] B. Mishra, S. Thakkar, and N. Jain, "Ultra-low power digital front-end for single lead ecg acquisition integrated with a time-to-digital converter," *IET Comput Digit Tech*, vol. 13, no. 6, pp. 453–460, 2019.

[19] T. S. Hall, C. M. Twigg, P. Hasler, and D. V. Anderson, "Application performance of elements in a floating-gate fpaa," in *2004 IEEE International Symposium on Circuits and Systems*, vol. 2, May 2004, pp. II–589.

[20] S. Lee, "Techniques for low-power high-performance adcs," Ph.D. dissertation, Massachusetts Institute of Technology, 50 Vassar St, Cambridge, MA 02142, 2 2014.

[21] J. Frigo, V. Kulathumani, S. Brennan, E. Rosten, and E. Raby, "Sensor network based vehicle classification and license plate identification system," in *Proceedings of the 2009 Sixth International Conference on Networked Sensing Systems*. IEEE, 2009, pp. 1–4.

[22] R. Sarpeshkar, "Analog versus digital: Extrapolating from electronics to neurobiology," *Neural Comput*, vol. 10, no. 7, pp. 1601–1638, 1998.

[23] P. Hasler, P. Smith, D. Graham, R. Ellis, and D. Anderson, "Analog floating-gate, on-chip auditory sensing system interfaces," *IEEE Sens J*, vol. 5, no. 5, pp. 1027–1034, 2005.

[24] J. Park, Y. Hwang, J. Oh, Y. Song, J. Park, and D. Jeong, "A compact self-capacitance sensing analog front-end for a touch detection in low-power mode," in *2019 IEEE/ACM International Symposium on Low Power Electronics and Design*, July 2019, pp. 1–6.

[25] J. Jeong, J. An, S. Jung, S. Hong, and O. Kwon, "A low-power analog delay line using a current-splitting method for 3-d ultrasound imaging systems," *IEEE Trans Circuits Syst II Express Briefs*, vol. 65, no. 7, pp. 829–833, July 2018.

[26] R. Groza and P. Farago, "Low power current-mode analog front-end for biomedical applications," in *2018 IEEE International Conference on Automation, Quality and Testing, Robotics*, May 2018, pp. 1–5.

[27] F. Morshedlou, N. Ravanshad, and H. Rezaee-Dehsorkh, "A low-power current-mode analog qrs-detection circuit for wearable ecg sensors," in *2018 25th National and 3rd International Iranian Conference on Biomedical Engineering*, Nov 2018, pp. 1–6.

[28] J. Kim, C. Kim, K. Kim, and H. Yoo, "An ultra-low-power analog-digital hybrid cnn face recognition processor integrated with a cis for always-on mobile devices," in *2019 IEEE International Symposium on Circuits and Systems*, May 2019, pp. 1–5.

[29] N. Uetake, R. Zhang, T. Nakada, and Y. Nakashima, "A programmable analog calculation unit for vector computations," in *2018 IEEE Symposium in Low-Power and High-Speed Chips*, April 2018, pp. 1–3.

[30] R. Harrison, J. Bragg, P. Hasler, B. Minch, and S. DeWeerth, "A CMOS programmable analog memory-cell array using floating-gate circuits," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 1, pp. 4–11, Jan 2001.

[31] F. M. Bayat, X. Guo, H. A. Om'mani, N. Do, K. K. Likharev, and D. B. Strukov, "Redesigning commercial floating-gate memory for analog computing applications," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2015, pp. 1921–1924.

[32] B. Rumberg and D. Graham, "A floating-gate memory cell for continuous-time programming," in *IEEE International Midwest Symposium on Circuits and Systems*, Aug 2012, pp. 214–217.

[33] B. Rumberg and D. W. Graham, "Efficiency and reliability of fowler-nordheim tunnelling in cmos floating-gate transistors," *Electronics Letters*, vol. 49, no. 23, pp. 1484–1486, Nov 2013.

[34] D. W. Graham, E. Farquhar, B. Degnan, C. Gordon, and P. Hasler, "Indirect programming of floating-gate transistors," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 54, no. 5, pp. 951–963, May 2007.

[35] B. D. Rumberg, "Low-power and programmable analog circuitry for wireless sensors," Ph.D. dissertation, West Virginia University, December 2014.

[36] G. Serrano, P. D. Smith, H. J. Lo, R. Chawla, T. S. Hall, C. M. Twigg, and P. Hasler, "Automatic rapid programming of large arrays of floating-gate elements," in *2004 IEEE International Symposium on Circuits and Systems (IEEE Cat. No.04CH37512)*, vol. 1, 2004, pp. I–I.

[37] E. K. F. Lee and P. G. Gulak, "A CMOS field-programmable analog array," *IEEE J Solid-State Circuits*, vol. 26, pp. 1860–1867, 1991.

[38] J. Becker, F. Henrici, S. Trendelenburg, M. Ortmanns, and Y. Manoli, "A continuous-time hexagonal field-programmable analog array in 0.13 μm CMOS with 186 MHz GBW," in *Digest of Technical Papers of the IEEE International Solid-State Circuits Conference.* IEEE, 2008, pp. 70–596.

[39] B. Rumberg, D. W. Graham, S. Clites, B. Kelly, M. Navidi, A. Dilello, and V. Kulathumani, "RAMP: Accelerating wireless sensor hardware design with a reconfigurable analog/mixed-signal platform," in *Proc. ACM/IEEE Conf. IPSN*, Seattle, WA, USA, April 2015, pp. 47–58.

[40] "panstamp nrg 2. technical details," https://github.com/panStamp/panstamp/wiki/panStamp-NRG-2.-Technical-details, cited December 2019.

[41] A. Dilello, S. Andryzcik, B. M. Kelly, B. Rumberg, and D. W. Graham, "Temperature compensation of floating-gate transistors in field-programmable analog arrays," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2017, pp. 1–4.

[42] V. Srinivasan, G. Serrano, C. M. Twigg, and P. Hasler, "A floating-gate-based programmable CMOS reference," *IEEE Trans. Circuits Syst. I*, vol. 55, no. 11, pp. 3448–3456, Dec. 2008.

[43] M. Gu and S. Chakrabartty, "Subthreshold, varactor-driven CMOS floating-gate current memory array with less than 150-ppm/° k temperature sensitivity," *IEEE J. Solid-State Circuits*, vol. 47, no. 11, pp. 2846–2856, Nov. 2012.

[44] L. Zhou and S. Chakrabartty, "A continuous-time varactor-based temperature compensation circuit for floating-gate multipliers and inner-product circuits," in *IEEE Int. Symp. on Circuits and Syst.*, May 2015.

[45] S. Lee and E. Sánchez-Sinencio, "Current reference circuits: A tutorial," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 3, pp. 830–836, 2021.

[46] B. Rumberg and D. Graham, "Sub-microwatt analog VLSI trainable pattern classifier," *Proc. IEEE MWSCAS*, pp. 214–217, Aug 2012.

[47] Yanjun Ma, T. Gilliland, Bin Wang, R. Paulsen, A. Pesavento, C. . Wang, Hoc Nguyen, T. Humes, and C. Diorio, "Reliability of pfet eeprom with 70-a tunnel oxide manufactured in generic logic cmos processes," *IEEE Transactions on Device and Materials Reliability*, vol. 4, no. 3, pp. 353–358, 2004.

[48] V. Srinivasan, G. J. Serrano, J. Gray, and P. Hasler, "A precision cmos amplifier using floating-gate transistors for offset cancellation," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 2, pp. 280–291, 2007.

[49] R. L. Hance, K. Erington, and M. A. Chonko, "Mobile ion contamination in cmos circuits: a clear and present danger," in *Proceedings of 1994 IEEE International Integrated Reliability Workshop (IRWS)*, 1994, pp. 3–.

[50] T. Prodromakis, P. Georgiou, K. Michelakis, and C. Toumazou, "Effect of mobile ionic-charge on cmos based ion-sensitive field-effect transistors (isfets)," in *2009 IEEE International Symposium on Circuits and Systems*, 2009, pp. 2165–2168.

[51] E. Aarts, J. Korst, and W. Michiels, *Simulated Annealing.* Boston, MA: Springer US, 2014, pp. 265–285.

[52] R. Chawla, D. W. Graham, P. D. Smith, and P. Hasler, "A low-power, programmable bandpass filter section for higher-order filter-bank applications," in *2005 IEEE International Symposium on Circuits and Systems*, May 2005, pp. 1980–1983 Vol. 3.

[53] T. J. Hamilton and A. van Schaik, "Silicon implementation of the generalized integrate-and-fire neuron model," in *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2011 Seventh International Conference on*, Dec 2011, pp. 108–112.

[54] G. Indiveri, "A low-power adaptive integrate-and-fire neuron circuit," in *Circuits and Systems, 2003. ISCAS '03. Proceedings of the 2003 International Symposium on*, vol. 4, May 2003, pp. IV–820–IV–823 vol.4.

[55] "Particle xenon. technical details," https://docs.particle.io/datasheets/discontinued/xenon-datasheet, cited July 2020.

[56] "Arduino nano. technical details," https://content.arduino.cc/assets/NanoV3.3sch.pdf, cited July 2020.

[57] "Arduino nano 33 iot. technical details," https://content.arduino.cc/assets/NANO33IoTV2.0sch.pdf, cited Feb 2021.

[58] "Digi xbee3. technical details," https://www.digi.com/resources/documentation/ digidocs/pdfs/90000982.pdf, cited July 2020.

[59] S. Bhattacharyya, S. Andryzcik, and D. Graham, "An acoustic vehicle detector and classifier using a reconfigurable analog/mixed signal platform," *Journal of Low Power Electronics and Applications*, vol. 10, no. 1, Mar. 2020.

[60] B. Rumberg, D. W. Graham, and V. Kulathumani, "A low-power, programmable analog event detector for resource-constrained sensing systems," in *Proceedings of the 55th IEEE International Midwest Symposium on Circuits and Systems.* IEEE, 2012, pp. 338–341.

[61] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita, "A line in the sand: A wireless sensor network for target detection, classification, and tracking," *Computer Networks*, vol. 46, pp. 605–634, 2004.

[62] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy-aware wireless microsensor networks," *IEEE Signal Process Mag*, vol. 19, pp. 40–50, 2002.

[63] C. Gu, J. A. Rice, and C. Li, "A wireless smart sensor network based on multi-function interferometric radar sensors for structural health monitoring," in *Proceedings of the 2012 IEEE Topical Conference on Wireless Sensors and Sensor Networks*. IEEE, 2012, pp. 33–36.

[64] A. Arora, R. Ramnath, E. Ertin, P. Sinha, S. Bapat, V. Naik, V. Kulathumani, H. Zhang, H. Cao, M. Sridharan, S. Kumar, N. Seddon, C. Anderson, T. Herman, N. Trivedi, M. Nesterenko, R. Shah, S. Kulkami, M. Aramugam, L. Wang, M. Gouda, Y. Choi, D. Culler, P. Dutta, C. Sharp, G. Tolle, M. Grimmer, B. Ferriera, and K. Parker, "Exscal: elements of an extreme scale wireless sensor network," in *Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*. IEEE, 2005, pp. 102–108.

[65] B. Malhotra, I. Nikolaidis, and J. Harms, "Distributed classification of acoustic targets in wireless audio-sensor networks," *Computer Networks*, vol. 52, pp. 2582–2593, 2008.

[66] M. Uttarakumari, A. S. Koushik, A. S. Raghavendra, A. R. Adiga, and P. Harshita, "Vehicle detection using acoustic signatures," in *Proceedings of the International Conference on Computing, Communication and Automation*. IEEE, 2017, pp. 1173–1177.

[67] M. Malinowski, M. Moskwa, M. Feldmeier, M. Laibowitz, and J. Paradiso, "Cargonet: a low-cost micropower sensor node exploiting quasi-passive wakeup for adaptive asychronous monitoring of exceptional events," in *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*. ACM, 2007, pp. 145–159.

[68] S. Jevtic, M. Kotowsky, R. Dick, P. Dinda, and C. Dowding, "Lucid dreaming: reliable analog event detection for energy-constrained applications," in *Proceedings of the 6th International Conference on Information Processing in Sensor Networks*. ACM, 2007, pp. 350–359.

[69] J. Ding, S.-Y. Cheung, C.-W. Tan, and P. Varaiya, "Signal processing of sensor node data for vehicle detection," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems*. IEEE, 2004, pp. 70–75.

[70] B. Rumberg, D. W. Graham, and V. Kulathumani, "Hibernets: energy-efficient sensor networks using analog signal processing," in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*. ACM, 2010, pp. 129–139.

[71] B. Rumberg and D. W. Graham, "A low-power and high-precision programmable analog filter bank," *IEEE Trans Circuits Syst II Express Briefs*, vol. 59, pp. 234–238, 2012.

[72] B. Rumberg and D. Graham, "A low-power magnitude detector for analysis of transient-rich signals," *IEEE J Solid-State Circuits*, vol. 47, pp. 676–685, 2012.

[73] B. Rumberg, D. W. Graham, and M. M. Navidi, "A regulated charge pump for tunneling floating-gate transistors," *IEEE Trans Circuits Syst I Regul Pap*, vol. 64, pp. 516–527, 2017.

[74] A. V. Zykina, "A lexicographic optimization algorithm," *Automation and Remote Control*, vol. 65, pp. 363–368, 2004.

[75] K. Miettinen, *Nonlinear Multiobjective Optimization*, ser. International Series in Operations Research and Management Science. Boston: Springer, 1999.

[76] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: A unifying approach for margin classifiers," *J Mach Learn Res*, vol. 1, pp. 113—141, 2001.

[77] K. M. H. Badami, S. Lauwereins, W. Meert, and M. Verhelst, "A 90nm CMOS, 6 µw power-proportional acoustic sensing frontend for voice activity detection," *IEEE J Solid-State Circuits*, vol. 51, pp. 291–302, 2016.

[78] M. Yang, C. Yeh, Y. Zhou, J. P. Cerqueira, A. A. Lazar, and M. Seok, "Design of an always-on deep neural network-based 1- μ W voice activity detector aided with a customized software model for analog feature extraction," *IEEE J Solid-State Circuits*, vol. 54, pp. 1764–1777, 2019.

[79] M. Price, J. Glass, and A. P. Chandrakasan, "A low-power speech recognizer and voice activity detector using deep neural networks," *IEEE J Solid-State Circuits*, vol. 53, pp. 66–75, 2018.

[80] S. Jeong, Y. Chen, T. Jang, J. M. Tsai, D. Blaauw, H. Kim, and D. Sylvester, "Always-on 12 nw acoustic sensing and object recognition microsystem for unattended ground sensor nodes," *IEEE J Solid-State Circuits*, vol. 53, pp. 261–274, 2018.

[81] M. Yang, C. Chien, T. Delbruck, and S. Liu, "A 0.5 v 55 μW 64 $\times$ 2 channel binaural silicon cochlea for event-driven stereo-audio sensing," *IEEE J Solid-State Circuits*, vol. 51, pp. 2554–2569, 2016.

[82] P. Mayer, M. Magno, and L. Benini, "Self-sustaining acoustic sensor with programmable pattern recognition for underwater monitoring," *IEEE Trans Instrum Meas*, vol. 68, pp. 2346–2355, 2019.

[83] S. Andryzcik and D. Graham, "Linearization of voltage-controlled oscillators using floating-gate transistors," *IEEE Transactions on Circuits and Systems II: Express Briefs*, Feb. 2021, Early Access Article.

[84] H. Bazzi, M. A. Chanine, A. Mohsen, and A. Harb, "A low-noise voltage-controlled ring oscillator in 28-nm FDSOI technology," in *Proc. ICM*, Beirut, Dec. 2017, pp. 1–4.

[85] M. Voelker, S. Pashmineh, J. Hauer, and M. Ortmanns, "Current feedback linearization applied to oscillator based ADCs," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 11, pp. 3066–3074, July 2014.

[86] A. Babaie-Fishani and P. Rombouts, "Highly linear VCO for use in VCO-ADCs," *Elec. Lett.*, vol. 52, no. 4, pp. 268–270, Feb. 2016.

[87] W. El-Halwagy, M. Dessouky, and H. El-Ghitani, "A programmable 8-bit, 10MHz BW, 6.8mW, 200MSample/sec, 70dB SNDR VCO-based ADC using SC feedback for VCO linearization," in *Proc. ICECS*, Abu Dhabi, Dec. 2013, pp. 157–160.

[88] S. Kabirpour and M. Jalali, "A highly linear current-starved VCO based on a linearized current control mechanism," *Integration, the VLSI Journal*, vol. 69, pp. 1–9, June 2019.

[89] K. M. AL-Tamimi and K. El-Sankary, "Preweighted linearized VCO analog-to-digital converter," *IEEE Trans. Very Large Scale Integr. (VLSI) Systems*, vol. 25, no. 6, pp. 1983–1987, Feb. 2017.

[90] J. Kim, J. Park, and J. G. Kim, "A low phase noise 0.9 / 1.8 ghz dual-band lc vco in 0.18 um cmos technology," in *2018 International Conference on Electronics, Information, and Communication (ICEIC)*, Jan 2018, pp. 1–2.

[91] Q. Gu, T. Huang, L. Pu, and S. Xie, "Design of electrical parameter measurement system of quartz crystal oscillator," in *2017 10th International Symposium on Computational Intelligence and Design (ISCID)*, vol. 1, Dec 2017, pp. 507–510.

[92] J. Lee, A. K. George, and M. Je, "An ultra-low-noise differential relaxation oscillator based on a swing-boosting scheme," in *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan 2018, pp. 277–278.

[93] H. Asano, T. Hirose, T. Ozaki, N. Kuroki, and M. Numa, "An area-efficient, 0.022-mm2, fully integrated resistor-less relaxation oscillator for ultra-low power real-time clock applications," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2017, pp. 1–4.

[94] G. Grau, U. Langmann, W. Winkler, D. Knoll, J. Osten, and K. Pressel, "A current-folded up-conversion mixer and a vco with center-tapped inductor in a sige-hbt technology for 5 ghz wireless lan applications," in *Proceedings of the 1999 Bipolar/BiCMOS Circuits and Technology Meeting (Cat. No.99CH37024)*, 1999, pp. 161–164.

[95] H. Bazzi, M. A. Chanine, A. Mohsen, and A. Harb, "A low-noise voltage-controlled ring oscillator in 28-nm fdsoi technology," in *2017 29th International Conference on Microelectronics (ICM)*, Dec 2017, pp. 1–4.

[96] Y. Takezawa, K. Kiyoyama, K. Shimokawa, Z. Qian, H. Kino, T. Fukushima, and T. Tanaka, "Ultrawide range square wave impedance analysis circuit with ultra-slow ring-oscillator using gate-induced drain-leakage current," in *2017 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Oct 2017, pp. 1–4.

[97] C. Huo, T. Xia, and H. Li, "A 400mhz current starved ring oscillator with temperature and supply voltage insensitivity," in *Solid-State and Integrated Circuit Technology (ICSICT), 2014 12th IEEE International Conference on*, Oct 2014, pp. 1–3.

[98] V. Michal, "On the low-power design, stability improvement and frequency estimation of the cmos ring oscillator," in *Radioelektronika (RADIOELEKTRONIKA), 2012 22nd International Conference*, April 2012, pp. 1–4.

[99] B. Yang, Z. Lu, and J. Zhou, "A 13 ghz wide-tuning-range low-phase-noise ring oscillator utilizing frequency multiplication technique," in *2015 IEEE 11th International Conference on ASIC (ASICON)*, Nov 2015, pp. 1–4.

[100] R. Chawla, F. Adil, G. Serrano, and P. E. Hasler, "Programmable g_m–c filters using floating-gate operational transconductance amplifiers," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 3, pp. 481–491, March 2007.

[101] B. Rumberg, S. Clites, H. Abulaiha, A. DiLello, and D. Graham, "Continuous-time programming of floating-gate transistors for nonvolatile analog memory arrays," *J. Low Power Electron. Appl.*, vol. 11, no. 1, 2021.

[102] R. Harrison, J. Bragg, P. Hasler, B. Minch, and S. DeWeerth, "A CMOS programmable analog memory-cell array using floating-gate circuits," *IEEE Trans. Circuits and Syst. II. Analog Digit. Signal Process.*, vol. 48, no. 1, pp. 4–11, Jan. 2001.

[103] A. Basu and P. E. Hasler, "A fully integrated architecture for fast and accurate programming of floating gates over six decades of current," *IEEE Trans. Very Large Scale Integr. (VLSI) Systems*, vol. 19, no. 6, pp. 953–962, June 2011.

[104] B. Rumberg and D. W. Graham, "Efficiency and reliability of Fowler-Nordheim tunnelling in CMOS floating-gate transistors," *Elec. Lett.*, vol. 49, no. 23, pp. 1484–1486, Nov. 2013.

[105] T. Lee and P. A. Abshire, "An ultra-low frequency ring oscillator with programmable tracking using a phase-locked loop," in *Proc. IEEE MWSCAS*, Boise, ID, USA, Aug. 2012, pp. 17–20.

[106] S. M. Sharroush, "A novel variable-gain amplifier based on an FGMOS transistor," in *Proc. ICEDSA*, Ras Al Khaimah, UAE, Dec. 2016, pp. 1–4.

[107] G. Serrano, P. D. Smith, H. J. Lo, R. Chawla, T. S. Hall, C. M. Twigg, and P. Hasler, "Automatic rapid programming of large arrays of floating-gate elements," in *Proc. IEEE ISCAS*, Vancouver, BC, Canada, May 2004, pp. I–I.

[108] S. Song, K. C. Chun, and C. H. Kim, "A logic-compatible embedded flash memory for zero-standby power system-on-chips featuring a multi-story high voltage switch and a selective refresh scheme," *IEEE J. Solid-State Circuits*, vol. 48, no. 5, pp. 1302–1314, 2013.

[109] E. Rodriguez-Villegas, *Low Power and Low Voltage Circuit Design with the FGMOS Transistor*. Herts, UK: The Institution of Engineering and Technology, 2006.

[110] F. Pepe and P. Andreani, "An accurate analysis of phase noise in CMOS ring oscillators," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, vol. 66, no. 8, pp. 1292–1296, 2019.

[111] V. Srinivasan, G. J. Serrano, J. Gray, and P. Hasler, "A precision CMOS amplifier using floating-gate transistors for offset cancellation," *IEEE J. Solid-State Circuits*, vol. 42, no. 2, pp. 280–291, Jan. 2007.

[112] A. Dilello, S. Andryzcik, B. M. Kelly, B. Rumberg, and D. W. Graham, "Temperature compensation of floating-gate transistors in field-programmable analog arrays," in *Proc. IEEE ISCAS*, Baltimore, MD, USA, May 2017, pp. 1–4.

[113] B. Rumberg, D. W. Graham, and M. M. Navidi, "A regulated charge pump for tunneling floating-gate transistors," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 64, no. 3, pp. 516–527, May 2017.

[114] Y. Ma *et al.*, "Reliability of pFET EEPROM with 70-å tunnel oxide manufactured in generic logic CMOS processes," *IEEE Trans. Device Mater. Rel.*, vol. 4, no. 3, pp. 353–358, Dec. 2004.

[115] G. Taylor and I. Galton, "A mostly-digital variable-rate continuous-time delta-sigma modulator adc," *IEEE Journal of Solid-State Circuits*, vol. 45, no. 12, pp. 2634–2646, Dec 2010.

[116] W. Yu, J. Kim, K. Kim, and S. Cho, "A time-domain high-order mash *deltasigma* adc using voltage-controlled gated-ring oscillator," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 4, pp. 856–866, April 2013.

[117] W. Jiang, V. Hokhikyan, H. Chandrakumar, V. Karkare, and D. Markovic, "A 50mv linear-input-range vco-based neural-recording front-end with digital nonlinearity correction," in *2016 IEEE International Solid-State Circuits Conference (ISSCC)*, Jan 2016, pp. 484–485.

[118] S. Huq, O. Baroi, S. Shihab, and S. Biswas, "Comparative study and design of current starved ring oscillators in 16 nm technology," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, Early Access Article.

[119] M. Trakimas and S. R. Sonkusale, "An adaptive resolution asynchronous adc architecture for data compression in energy constrained sensing applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 58, no. 5, pp. 921–934, 2011.

[120] B. Kelly, A. Dilello, and D. Graham, "Reconfigurable analog preprocessing for efficient asynchronous analog-to-digital conversion," *Journal of Low Power Electronics and Applications*, vol. 25, no. 9, Aug. 2019.

Appendix A

An Asynchronous Peak-Sampling ADC with Floating-Gate Offset Correction

A.1 An Overview of Asynchronous Peak Sampling ADCs

Fixed-rate sampling ADCs such as the VCO-ADC have the benefit of years of characterization, as well as easily understood and predictable performance. However, these ADCs have no method for adaptive energy consumption when sampling signals with sparse information — signals that have long periods of no useful information, followed by short periods of relevant data. The magnitude of samples converted directly corresponds to the power consumption of the ADC: the more samples converted, the more energy is used. The ability to change a sampling rate based on the signal itself allows an ADC to remain dormant during periods of inactivity, and then begin sampling when information rich data is present.

With that in mind, the ADC that will be discussed is an asynchronous peak-sampling ADC (APS-ADC). Asynchronous ADCs only operate when relevant data is present, determined via different metrics such as frequency or amplitude. By way of non-uniform sampling, keeping the ADC in a low power state allows for large improvements to data converter power consumption, especially in resource-constrained environments [119].

The issue with when to wake-up the ADC still remains. This is typically done with an ASIC, designed to detect a specific type of event, and trigger an ADC to begin converting data. Previously in this work, it has been discussed that a reconfigurable ASP front end

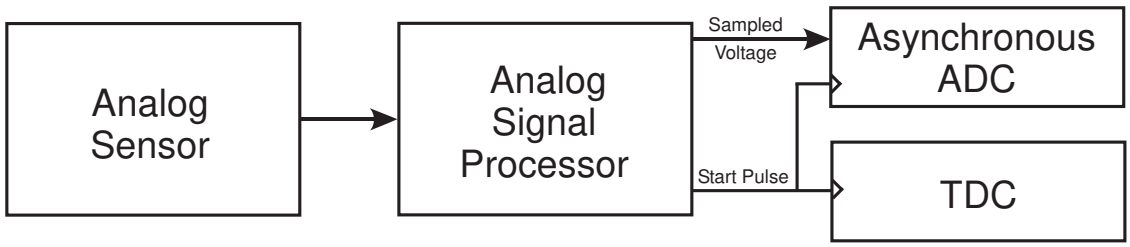


Figure A.1: Block diagram of asynchronous ADC with analog sensor input.

could be used as a catch-all to deal with a large group of event detectors. Chapter 6 covered the use of the RAMP in this exact type of solution. The system presented in [120] utilizes the RAMP as the event detector and asynchronous ADC as the data converter. This work aims to highlight the benefits of such systems, utilizing reconfigurable front-end ASP as the event detection. However, the reconfigurable platforms themselves can benefit from addition low-power detection methods and ADC operation.

The APS-ADC that will be discussed implements its own event detection while also solving nonlinearities found in the design from [120]. This event detection is specifically to pick up only local maxima and local minima. When a local extrema occurs, the ADC triggers a conversion. This method is implemented through analog circuits without the need for precise biasing, as in [120]. The event detection in the APS-ADC could be implemented by the RAMP, or the RAMP could detect that relevant events were occuring (such as speech, but not other transient signals) and cause the APS-ADC to begin sampling only local extrema.

Figure A.1 shows a general block diagram for this type of sampling and asynchronous data conversion. The analog sensor is read by an ASP front end. The ASP front end triggers a time-to-digital conversion (TDC) while also triggering the ADC. The TDC result is a required piece of information so that the signal can be reconstructed; The time between each sample is used to regenerate the original signal. The TDC is constructed the same way as the counter described in section 7.3, with the main difference relating to when it is reset. For the VCO-ADC, the counter is reset at a specific frequency (namely, the sampling frequency of the ADC). An asynchronous ADC does not have a set sampling frequency. Instead, the counter for an asynchronous ADC is only read (and reset) when a new sample occurs. This is what changes the counter into a TDC. The TDC is keeping track of the time since the

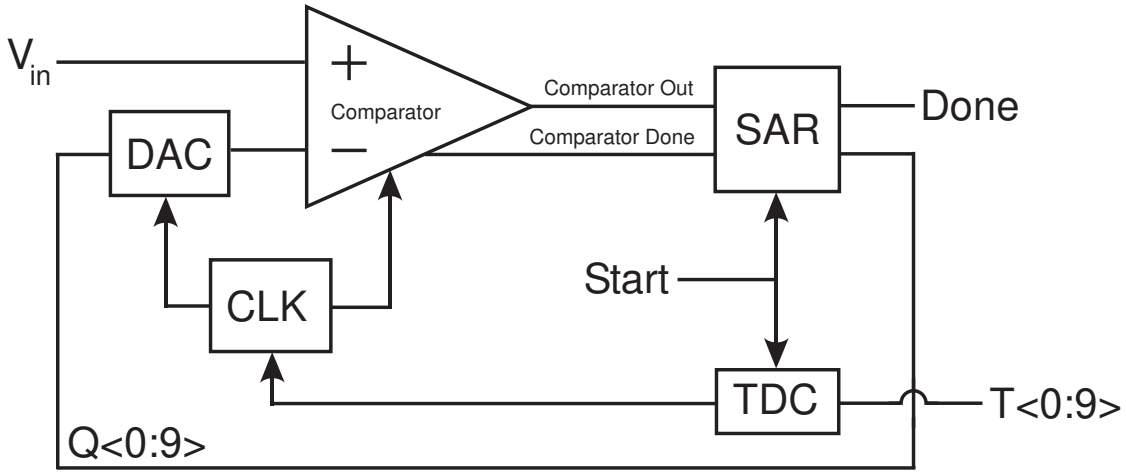


Figure A.2: Block diagram of the SA-ADC.

last data point, allowing for the signal to be reconstructed from data values and time.

A.2 System Architecture

The APS-ADC uses at its core a successive approximation ADC (SA-ADC). The SA-ADC performs the conversion of sampled extrema values. An SA-ADC implements a pseudo binary-search and compares its results (via comparator) to determine the digital output. This pseudo binary-search is accomplished with a successive approximation register (SAR). The SAR output is sent through a digital-to-analog converter (DAC) to be compared back to the original input. Figure A.2 shows the SA-ADC system. The SA-ADC system is utilized within the APS-ADC. Working with the ASP front end, this system is able to detect and convert local extrema values while also keeping track of the time between each sample.

An APS-ADC has been fabricated in a 350nm design process. This Appendix will expand on the operation of the event detection stage and how FG programming will be leveraged to linearize the ADC's performance. The event detection in the APS-ADC from [120] is constructed using the RAMP 1.0 system.

The APS-ADC that has been fabricated implements its own event detection. This new event detection is meant to serve two purposes. The first is for a scenario where this APS-ADC is used as a standalone device. In this case, the RAMP's event detection capabilities,

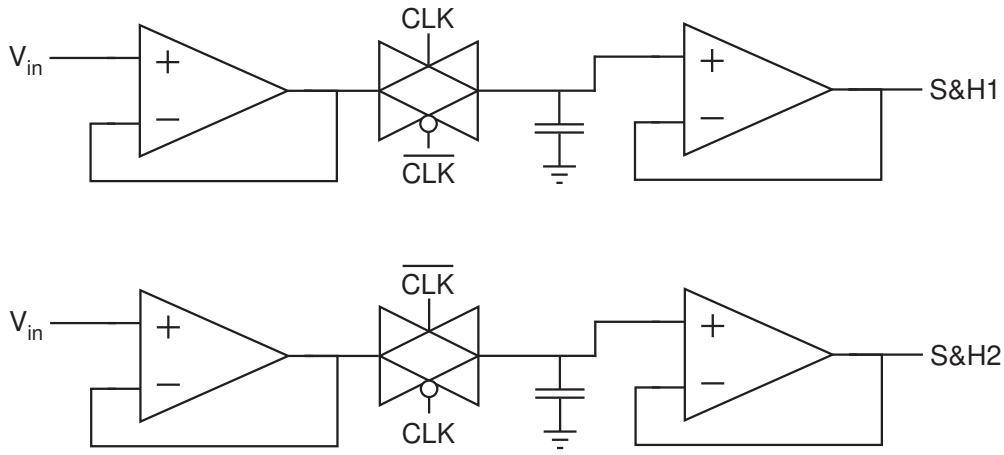


Figure A.3: Schematic of APS-ADC Sample and Holds used as event detectors.

while adequate, are overkill. The APS-ADC overhead is greatly increased when the RAMP is used as the event detector. While the RAMP is an improvement on a digital solution, a custom analog ASP has the benefits of low overhead while still leveraging the type of detection found on the RAMP. The other purpose for this new event detection is to prototype the system for future implementation on the ramp. The RAMP 1.1 does not have the components necessary to synthesize this circuit at the present, but a future, RAMP 1.2 could and should include these devices to enhance its ASP capabilities.

The event detection system is simple and requires the use of an alternating sample and hold (SaH) circuit. Figure A.3 shows the two SaHs required for the event detector. The event detector aims to only trigger a sample from the ADC when a local maximum or local minimum occurs. The SaHs are triggered by their clock signals. The clock signals are connected to the T-gates inside each SaH. With the opposite logic of each T-gate, the SaHs sample every other clock signal. When SaH number 1 is sampling the input, SaH number 2 is holding the sample from the previous clock cycle.

The alternating samples are sent to a comparator for analysis. When considering rising (or falling) transient signals, the comparator will receive signals of alternating magnitude each clock cycle. For a rising transient, SaH number 1 will output a value that is higher than SaH number 2. At the next clock cycle, SaH number 2 will output a value that is higher than SaH number 1. This oscillation will continue as long as the signal is rising. Since the

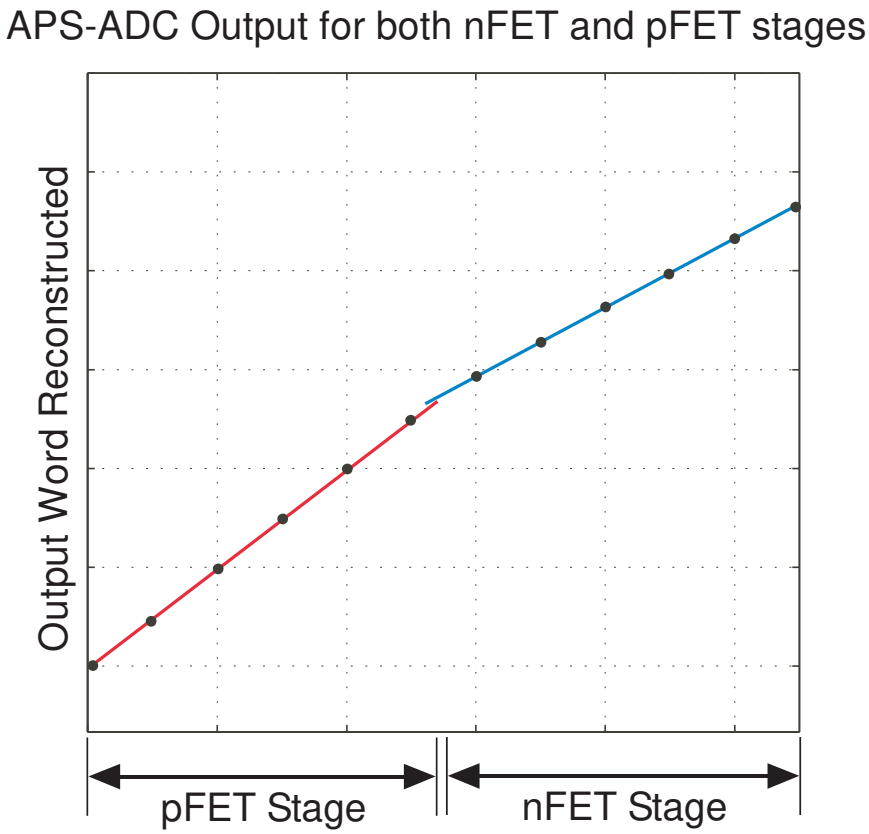


Figure A.4: Output of APS-ADC showing output word reconstructed to its input analog value. The difference in linearity between nFET and pFET stage leads to a large output error in ADC performance.

outputs of the SaH are tied to the two inputs of the comparator, the comparator output will oscillate back and forth between a high and low output. When the rising transient reaches a peak, the comparator will no longer be able to tell the difference between the two sampled values, causing it to output zeros for the next series of clock cycles until the transient begins falling (this explanation is flipped when starting with a falling transient signal).

The output of the comparator is read into a shift register. When the shift register reads a pattern that shows the comparator has stopped oscillating, the current sample is sent to the asynchronous ADC for conversion. The APS-ADC system utilizes two comparators. The first comparator, as discussed is used to check the difference between the two SaH outputs. The second comparator is utilized within the asynchronous ADC itself. The comparator is composed of an n-stage and p-stage for rail-to-rail operation. When the input signal is above mid-rail, the n-stage is utilized for conversion. When the input signal is below mid-rail, the

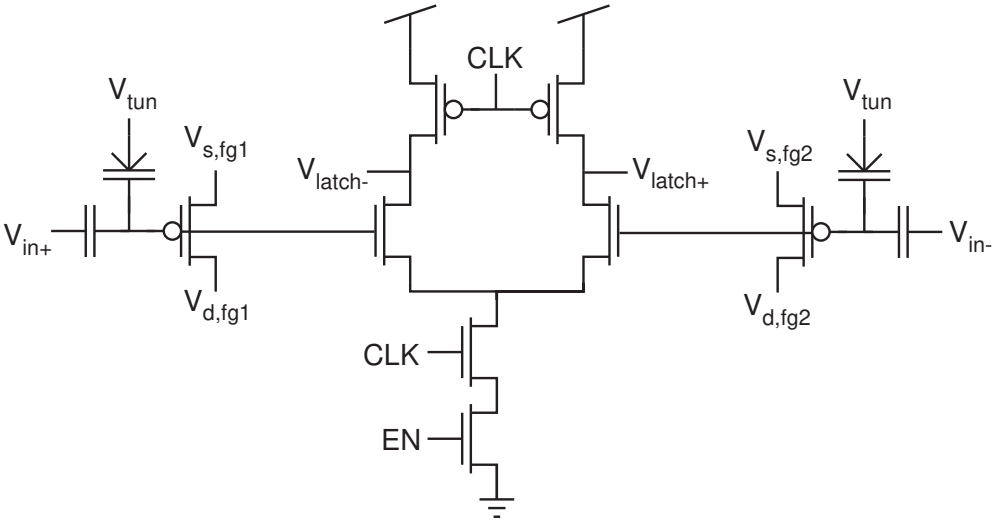


Figure A.5: Schematic of APS-ADC Comparator nFET input stage with FG transistors for programming out the offset.

p-stage is utilized for conversion.

Figure A.4 illustrates the issue that arose for the comparator used in [120]. The n-stages and p-stages had a different offset, causing a large nonlinearity to be present within the system. The n-stage performs differently than the p-stage, causing integral and differential nonlinearities to be present when testing the functionality of the system. In the case of an asynchronous ADC, this leads to a reconstructed signal with a large distortion from the input.

To combat this nonlinearity caused by offset error, the comparator input stages (both n-stage and p-stage) were outfitted with indirect FG transistors at each input. Figure A.5 shows the n-stage of the comparator. The input transitors now include indirect FG transistors for programming out the offset. Both sides of the comparator can be programmed separately to ensure linear operation. The effects of these FGs as well as the ADC as a whole need tested and verified.

Appendix B

Smart-Anchor Monitoring

This Appendix is part of an ongoing research project supported by the Department of Energy's National Energy Technology Laboratory under Federal Grant DE-FE0031825. The research project is aimed at monitoring the health and processing conditions within pulverized coal and fluidized-bed combustion boiler systems.

A Smart-Anchor was developed at WVU with the intent of measuring the internal temperature of the boiler system. The Smart-Anchor outputs a temperature dependent resistance that is used to report on the internal characteristics of the boiler system. This Appendix will detail the work-in-progress done in WVU's CES Lab to interface with the Smart-Anchor and report its electrical characteristics wirelessly to a base-station.

B.1 Sensor Interfacing and Wireless Communication

The objective of this work is to interface the electrical sensing outputs from the ceramic Smart-Anchor with an embedded processor that can wirelessly transmit electrical measurements to a central processing unit. The sensing circuitry can handle a variety of electrical sensors including resistance, voltage, and capacitance. Once data is aggregated at the central unit, it can be used for analysis and presented in real-time.

For the current, resistance-based version of the Smart-Anchor, a Wheatstone bridge has been designed and tested to act as an interface between the Smart-Anchor and the wireless embedded processor. The Wheatstone bridge uses the resistance of the Smart-Anchor,

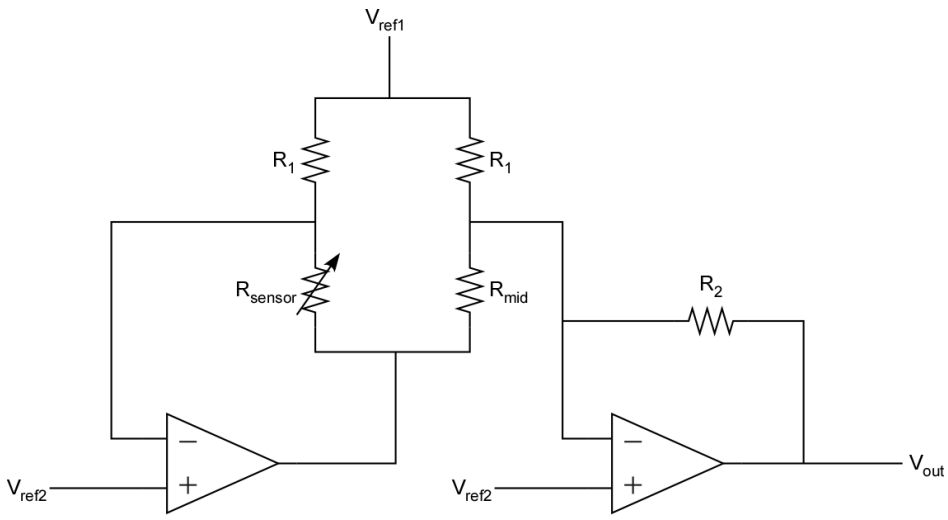


Figure B.1: Schematic of Wheatstone bridge.

labeled R_{sensor} in figure B.1, converting to a voltage that can be read by the embedded processor.

The MCU that was chosen for this project is the Arduino Nano IoT 33 (NIoT33) for its low-enegery Bluetooth (BLE) capabilities. One NIoT33 is connected to the output of the Wheatstone bridge for each Smart-Anchor (called the wireless-node). An additional NIoT33 acts as the base station, connected to a dedicated PC (called the base-station). The wireless-node reads the voltage from the Wheatstone bridge and sends the data over BLE to the base-station. The base-station is capable of data-logging multiple BLE characteristics with data and time-stamps. The PC is then used to monitor the signals being collected by the base-station for any anomaly or unexpected results from the Smart-Anchor.

The NIoT33 was chosen for its plethora of functions supported by the Arduino family of MCUs. The system has been tested directly with the Smart-Anchor to prove the wireless capabilites of the data-logging system over a 35 hour test. The Wheatstone bridge was connected directly to the Smart-Anchor to convert its resistance to a voltage. The voltage was read by the NIoT33 and communicated wirelessly over BLE to the base-station. The base-station then converted the voltage values to resistance of the Smart-Anchor and timestamped each sample. The results of the test can be seen in figure B.2.

To progress the system towards a complete package, a printed circuit board (PCB) was

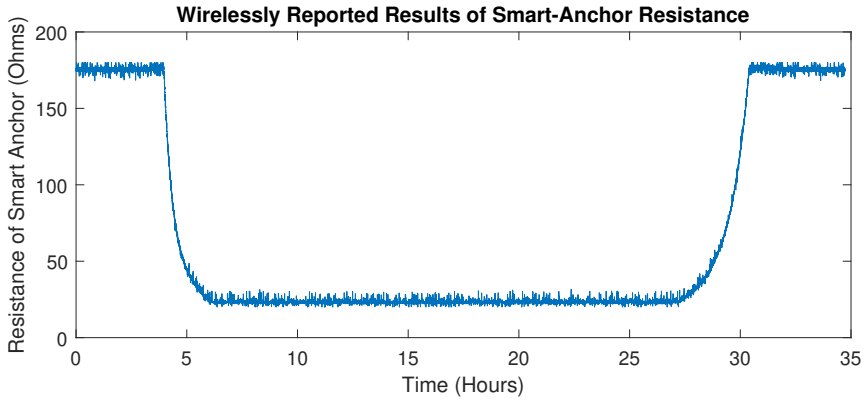


Figure B.2: Results of the wireless test, implementing the Wheatstone bridge interface and BLE communication. The voltage data has been converted at the base-station to the resistance of the Smart-Anchor.

designed to house the wireless-node. The PCB contains the NIoT33, Wheatstone bridge, headers for connecting to the Smart-Anchor, and a battery pack for wireless operation. The 2 layer PCB design can be seen in figure B.3. The design supports future changes to the Smart-Anchor resistive composition. Future work on this project will include additional Wheatstone bridges for selective measuring for higher resolution.

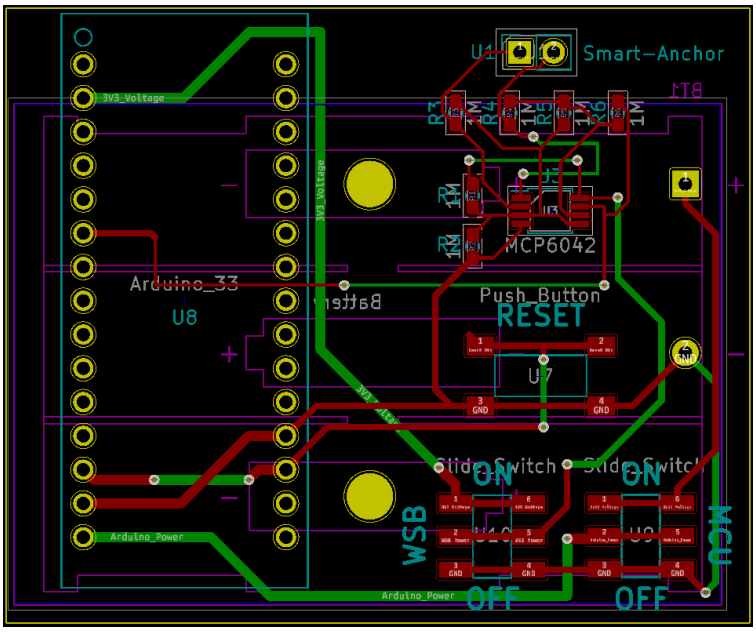


Figure B.3: Printed circuit board for the Arduino Nano IoT 33, SMD components to construct the Wheatstone Bridge, battery pack, and headers for connecting to the Smart-Anchor.

Appendix C

Wireless Programming with Low-Energy Bluetooth

The following two sets of code were implemented on the Arduino Nano IoT 33 for programming the RAMP 1.1 over low-energy Bluetooth (BLE). The first program was installed on the "base-staion." This code is used to broadcast the desired programming setup to the RAMP 1.1. The code uses the BLE service with "char" characteristics to pass the required connection lists.

```
/* Filename: BLE_baseStation.ino
 * Code for BLE wireless programming of the RAMP 1.1 using the
 * Arduino Nano IoT 33
 * Written By: Steven Andryzcik and Swagat Bhattacharyya
 */

// Include Header files
#include "String.h"
#include "BoardLib.h"
#include <ArduinoBLE.h>

// Define variables and constants
#define MaxMode 3                // The max number of operational
                                 // modes for the RAMP
```

```
byte Length;                    // This is the length in character
                                // of the converted input byte
                                // before padding
byte NLoops;                    // This is the number of loops
                                // needed to get all info
char CharBuf[29];               // Stores the character buffer
char CharBuffInit[29] = "";     // Setup temp variable for
                                // BLE service
char CommandInit[4] = "";       // Setup temp variable for
                                // BLE service
int Mode = 1;                   // RAMP Operational mode
BoardLib boardlib;              // Create a new instance of the
                                // BoardLib class

// Set BLE service
BLEService programmingService ...
("01010111-0101-0110-0101-0101XXRAMP11");

// BLE Char Characteristic - custom 128-bit UUID
// readable by wirelss node
BLECharCharacteristic charCharacteristic ...
("01010111-0101-0110-0101-0101XXRAMP11", BLERead | BLEWrite);
BLECharCharacteristic commandCharacteristic ...
("01010111-0101-0110-0101-0101XXRAMP11", BLERead | BLEWrite);

void setup() {
  // Run custom setup commands
  boardlib.begin();
  boardlib.Flash(1);

  // Initialize Bluetooth
  BLE.begin();
```

```
  BLE.setLocalName("baseStation");
  BLE.setAdvertisedService(programmingService);

  // add the characteristics to the service
  programmingService.addCharacteristic(charCharacteristic);
  programmingService.addCharacteristic(commandCharacteristic);

  // add service
  BLE.addService(programmingService);

  // set the initial value for the characeristic:
  charCharacteristic.writeValue(CharBuffInit);
  commandCharacteristic.writeValue(voltageBuf);

  // Begin advertising on BLE
  BLE.advertise();
}

void loop() {
  // Read data from serial port if available
  String Command = "";
  String Numerical = "";
  if(Serial.available() > 0){
    // Read the incoming byte
    String InputByte = String(Serial.read(),BIN);

    // Pad the incoming byte if needed
    Length = InputByte.length();
    for (byte i = 0; i < 8-Length; i++) {
      InputByte = "0" + InputByte;
    }
```

```
// Read the command
for (byte i = 0; i < 4; i++) {
  Command += String(InputByte[i]);
}

// Read the numerical data
for (byte i = 4; i < 8; i++) {
  Numerical += String(InputByte[i]);
}

// Find number of additional loops needed to read
// all data for current command
if (Command == "0010"){            // CAB block SPI
  NLoops = 3;
} else if (Command == "0011"){    // DAC SPI
  NLoops = 2;
} else{
  NLoops = 0;
}

// Keep reading data if needed
for (byte i = 0; i < NLoops; i++) {
  // Check if we do have the relevant information
  if(Serial.available() > 0){
    // Read the incoming byte
    String InputByte = String(Serial.read(),BIN);

    // Pad the incoming byte if needed
    Length = InputByte.length();
    for (byte i = 0; i < 8-Length; i++) {
      InputByte = "0" + InputByte;
    }
```

```
        // Read the numerical data
        for (byte i = 0; i < 8; i++) {
          Numerical += String(InputByte[i]);
        }

      } else{
        Command = "1110";
        break;
      }
    }

  }
  else{       // Put in the skip command
    Command = "1111";
    Numerical = "1111";
  }

  // Check if a program command is
  // being given (will start with 'C' or 'F')
  if (Command == "0010" || Command == "0011"){
    // Read string to buffer
    Numerical.toCharArray(CharBuf, 29);

    // Wait for a BLE connection to wireless node
    BLEDevice central = BLE.central();

        // If the connected to the wireless node
    if (central) {
      // Turn on the LED to indicate the connection:
      digitalWrite(LED_BUILTIN, HIGH);
      while (central.connected()) {
```

```
        // Set BLE characteristics to broadcast
        // the command and the connection list
        charCharacteristic.writeValue(CharBuf);
        commandCharacteristic.writeValue(Command);
      }
    digitalWrite(LED_BUILTIN, LOW);
    }
  }
}
```

The second program was installed on the "wireless-node." This code is used to scan for the BLE service being broadcast by the base-station. When a base-station with the correct UUID and BLE service is detected, the wireless-node connects and reads the connections list being broadcast. The connections list is then passed into the RAMP 1.1 via serial peripheral interface.

```
/* Filename: BLE_baseStation.ino
 * Code for BLE wireless programming of the RAMP 1.1 using the
 * Arduino Nano IoT 33
 * Written By: Steven Andryzcik and Swagat Bhattacharyya
 */

// Include some header files
#include "String.h"
#include "BoardLib.h"
#include <ArduinoBLE.h>

// Define some other variables and constants
#define MaxMode 3              // The max number of operational
                               // modes for the RAMP
char CharBuf[29];              // Stores the character buffer
char Command[4];               // Stores the command
int Mode = 1;                  // RAMP Operational mode
unsigned short ResState = 1;   // State of power measurement
```

```
                                  // short resistor
unsigned short InjState = 0;      // State of the injection prep
BoardLib boardlib;                // Create a new instance of
                                  // the BoardLib class

void setup() {
  // Run custom setup commands
  boardlib.begin();
  boardlib.Flash(1);

  // Initialize Bluetooth
  BLE.begin();
  BLE.setLocalName("wirelessNode");

  // start scanning for base station
  BLE.scanForUuid("01010111-0101-0110-0101-0101XXRAMP11");
}

void loop() {
  // check if wireless node has been discovered
  BLEDevice peripheral = BLE.available();

  // ensure connected to the correct BLE device
  if (peripheral.localName() != "baseStation") {
      return;
    }

  // stop scanning
  BLE.stopScan();

  if (peripheral.connect()) {
```

```
// Turn on the LED to indicate the connection
digitalWrite(LED_BUILTIN, HIGH);

// Retrieve the service characteristics from the base station
BLECharCharacteristic charCharacteristic = ...
peripheral.charCharacteristic ...
("01010111-0101-0110-0101-0101XXRAMP11");
BLECharCharacteristic commandCharacteristic = ...
peripheral.commandCharacteristic ...
("01010111-0101-0110-0101-0101XXRAMP11");

// Read the characteristics into memory
charCharacteristic.readValue(CharBuf);
commandCharacteristic.readValue(Command);

// Adjust the mode based on serial inputs and ping back
if (Command == "0000"){ // Mode change command
  if (Numerical == "0001"){
    Mode = 1;
  } else if (Numerical == "0010"){
    Mode = 2;
  } else if (Numerical == "0011"){
    Mode = 3;
  }
  // Use skip code
  Command = "1111";
}else if(Command == "1001"){
  Command = "1111";
}

switch(Mode){
  case 1:  // Programming mode
```

```
// Parse the command
if (Command == "0010" || Command == "0011"){
  // Read string to buffer
  Numerical.toCharArray(CharBuf, 29);

  // Output commands to SPI ports
  if (Command == "0010"){
     // CAB block SPI
    boardlib.SPIOut(BitsCAB, SCLK_CAB, ...
    CS_CAB, DIN_CAB, CharBuf);
  } else if (Command == "0011"){
    // DAC SPI
    boardlib.SPIOut(BitsDAC, SCLK_DAC, ...
    CS_DAC, DIN_DAC, CharBuf);
  }
  // Print success code
  Serial.write(1);
} else if (Command == "1000"){
  // Reset switches
  boardlib.Reset();
} else if (Command == "0001"){
  // Measurement resistor toggle
  ResState++;
  boardlib.shortRes(ResState);
} else if (Command == "0101"){
  // Injection mode toggle
  InjState++;
  boardlib.injCtrl(InjState);
} else if (Command == "0110"){
  // Read chip temperature
  boardlib.dispTemp();
} else if (Command == "0111"){
```

```
            // Read FPAA output pins
            boardlib.Measure();
          } else if (Command == "0100"){
            // Tunnel the FPAA
            boardlib.Tunnel();
          } else{
            Serial.write(0);
            boardlib.Flash(1);
          }
          break;
        case 2: // Measurement mode (Read analog pins)
          boardlib.Measure();
          break;
        case 3: // Sleep mode
          boardlib.Sleep();
          Mode++;
          break;
      }
      digitalWrite(LED_BUILTIN, LOW);
    }
}
```