2020

Programming of Floating-Gate Transistors for Nonvolatile Analog Memory Array

Haifa Abulaiha
hmabulaiha@mix.wvu.edu

Programming of Floating-Gate Transistors for Nonvolatile Analog Memory Array

Haifa Abulaiha

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
Computer Engineering

David W. Graham, Ph.D., Chair
Roy S. Nutter, Jr., Ph.D.
Hany H. Ammar, Ph.D.
Brian D. Woerner, Ph.D.
Debangsu Bhattacharyya, Ph.D.

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2020

Keywords: Floating-gate transistor; nonvolatile memory; continuous-time programming; floating-gate memory array; FPAA; reconfigurable

Abstract

Programming of Floating-Gate Transistors for Nonvolatile Analog Memory Array

Haifa Abulaiha

Since they were introduced, floating-gate (FG) transistors have been used as non-volatile digital memory. Recent research has shown that floating-gate transistors can be successfully used as analog memory, specifically as programmable voltage and current sources. However, their proliferation has been limited due to the complex programming procedure and the complex testing equipment. Analog applications such as field-programmable analog arrays (FPAAs) require hundreds to thousands of floating-gate transistors on a single chip which makes the programming process even more complicated and very challenging. Therefore, a simplified, compact, and low-power scheme to program FGs are necessary. This work presents an improved version of the typical methodology for FG programming. Additionally, a novel programming methodology that utilizes negative voltages is presented here. This method simplifies the programming process by eliminating the use of supplementary and complicated infrastructure circuits, which makes the FG transistor a good candidate for low-power wireless sensor nodes and portable systems.

Dedication

I dedicate this work to Sarah and Summer.

Acknowledgments

First, I would like to sincerely thank my committee chair and advisor, Dr. David W Graham, for giving me this opportunity to work with him. This dissertation would not be possible without his guidance and support. I would also like to thank all my lab mates for their help and continuous support.

I also want to thank Dr. Brian Woerner, Dr. Roy Nutter, Dr. Hany Ammar and Dr. Debangsu Bhattacharyya for serving on my committee, and for their valuable time and advice.

Thank you to my family for all of the love and support over the years. Special thanks to my parents, Mohamed and Khiria, for their unconditional support and for always believing in me.

And finally, to my husband, Akram. He has been there form the beginning encouraging me and pulling me up whenever I felt like falling down. He never doubted me and has been always there for me. We finally did it.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The real world is analog. Sensors and transducers are analog; they convert real-world parameters such as temperature into electrical analog signals. Typically, those analog signals are converted to the digital domain to be processed and for decisions to be made. Then, the digital signals are converted back to analog domain to either send them back to the outside world or to perform application-specific actions. A general block diagram of a sensor/actuator system example is shown in Fig. 1.1. However, digital processing systems consume high power and requires longer computational delay due to the computational complexity. Additionally, the process of analog-to-digital and digital-to-analog conversion is very power consuming as well. In [6], using a cooperative analog-digital signal processing shows a gain in power efficiency approximately equals 20-years. The trend in digital signal processing power consumption presented in [6] follows the trend given by Gene's Law [7]. Therefore, it is optimal to keep everything in the analog domain - even the processing part - or minimize the use of digital microprocessors by performing some or all of the computation in the analog domain.

One other way of comparing analog to digital systems is the cost of system resolution. In [8], it is shown that the cost of digital systems linearly varies with the digital system resolution, while the cost of analog systems exponentially varies with the system resolution. The cost here refers to a range of metrics including power dissipation, delay, area, design and manufacturing expenses. [8] shows the difference between the two systems in term of cost. For a resolution of less than a specific threshold, between 8-14 bits [9], analog systems is less expensive.

Performing computation in the analog domain requires customized integrated circuits

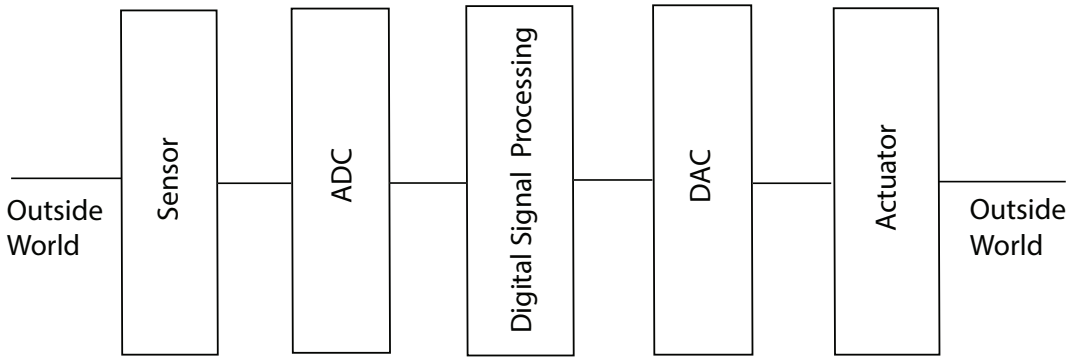

Figure 1.1: A block diagram of a standard sensing system. An analog-to-digital converter is used to convert analog signals to digital domain which is processed using a power-consuming digital signal processor.

specifically designed for that application. However, any change in the sensing specifications of that application, after the circuit has being fabricated, would require redesign of the whole integrated circuit. Therefore, a low-power reprogrammable integrated circuit in the analog domain, analogous to FPGA in the digital domain, is needed. Field-programmable analog arrays (FPAAs) are low-power analog systems designed for energy-constrained applications such as remote sensing applications, Fig. 1.2. FPAA consists not only of analog elements that can be connected in different configurations, but also analog components that can be programmed themselves to different parameters such as filters with programmable corner frequencies.

Programmable parameters of analog circuits are adjusted using biasing current/voltage. Floating-gate transistors are compact analog components that could be used as programmable current/voltage sources to set the programmable parameters for analog circuits. A floating-gate transistor is an electrical element that can be used in both digital and analog applications. Initially, FG transistors were mostly used in digital applications like flash memory; however, analog applications are becoming more popular, and more research is focusing on utilizing FG transistors in analog circuits. FG transistors are programmable analog memory elements known as analog non-volatile memory (NVM), which are suitable for large-scale programmable analog systems. A specific amount of charge can be stored-on/deleted-from the floating-gate using different programming methods. However, programming FG transistors in battery-powered applications requires low-power programmer circuits that utilize high DC voltage. This also requires a lot of infrastructure to switch between different operation

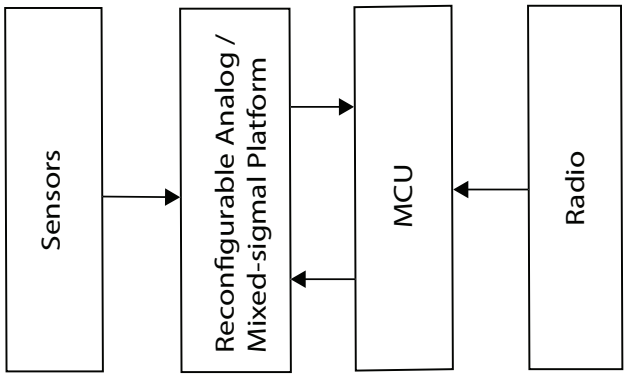

Figure 1.2: A block diagram of a real-world sensing system.

modes and to select/unselect a single FG transistor in an array configuration. Additionally, achieving accurate and repeatable programmed current/voltage is a challenge, especially in FG-dense applications.

The objective of this research is to develop programming technique for FG transistors that enables the use of FG analog application to fulfill the requirements of quick, low-power, repeatable and low-infrastructure overhead for field-programmable analog arrays, specifically for the RAMP platform [2]. In this document, I present two different FG programming methodologies—positive and negative voltages— that facilitate the use of FG transistors as analog memory in an array configuration. For each method, distinct FG memory cell and programmer circuit are introduced. These components are then used to build FG memory array. Two different programmer typologies for negative voltages injection are discussed. I tested all of the circuits for accuracy and repeatability. I also tested their feasibility with an application circuit that utilizes FG transistor array..

In chapter 2, I present floating-gate transistors, what are they, and how they have been used; additionally, I will explain their structure, how floating-gate transistors operate, and their applications in both digital and analog domains. In chapter 3, I give a more in-depth discussion of different programming techniques and present the first programming methodology to program a floating-gate transistor in a memory array. In chapter 4, I present two novel programmer circuits that utilize below-ground voltages to inject floating-gate transistors and how they are used with the FG array. In chapter 5, I present a newer prototype programmer circuit that is based on the circuits presented in chapters 3 and 4. In chapter 6, I will demonstrate how floating-gate transistors are integrated into a completely reconfigurable analog system to program programmable analog components. Finally, in chapter

7, I will summarize all the research work I have completed, and I will conclude with my recommendations for future work based on my research.

Chapter 2

Floating-Gate Transistor Overview

Since they were introduced, floating-gate transistors have had a wide range of applications in both digital and analog domains. In this chapter, I will present the different applications that utilizes FG transistor as a digital and analog memory element. I will also present the structure of the floating-gate transistor, operation modes, and different techniques to inject/ erase the floating-gate.

2.1 Flash Memory

Flash memory provides nonvolatile storage with a faster access time than hard disks. Additionally, they are small devices with very low power consumption which make them very adequate for low-power portable applications. The two major applications of digital flash memory are to be integrated into the logic system or to be used as a storage devices [10] . In the first application, the flash memory is used to store the system related software such as updates, while in the second application, flash memory is used as the main memory such as solid-state hard drives.

Since FG transistors are programmable devices and can store a charge for a long time even when the power supply is turned off, they have been used for nonvolatile digital (NVM) such as EPROM, EEPROM, and flash memory. The charge stored on the floating-gate node determines the state of the cell, due to the change in the threshold voltage of the transistor with the stored charge. For example, if the FG transistor is fully injected with electrons, and an input voltage is applied, the output voltage is going to be different than if the FG transistor was erased (completely tunneled). FG transistors are used as flash memory in two

main configurations: NAND and NOR [11, 12], based on how FG cells are organized. NOR flash memory was presented by Intel in 1978, while NAND flash memory was presented in 1984 by Toshiba. Most of the newer designs are using NAND over NOR because of its lower price and more dense configuration that is suitable for high-performance applications.

2.1.1 NAND Flash Memories

In NAND flash memory, FG transistors are arranged in groups of 16 or 32 that are connected in series. A basic structure of the NAND memory cell is shown in Fig. 2.1. For each group, two selection transistors are used, one transistor M_{SSL} to connect the series group to ground and the second transistor M_{DSL} to connect the series to bitline. When a cell is being read, a 0V is applied at its gate while a high voltage $\sim 5V$ is applied at the other gates of the series groups. This will allow transistors to work as pass-transistor regardless of their threshold voltage. When a NAND cell is erased, the threshold voltage for that cell is shifted to be negative. On the other hand, a programmed NAND cell has a positive threshold voltage. Therefore, when an erased NAND cell is selected, it sinks current, while if a programmed NAND cell is selected no current is sank. NAND flash cells could be grouped into blocks, where a block can be globally erased. Erasing a block placing 1 on all of the cells in the block. In general a NAND block can last upto 100,000 programming/ erasing cycles.

2.1.2 NOR Flash Memories

In NOR flash memory, FG transistors are arranged in an array configuration. In NOR memory, FG cells are programmed using Hot-electron injection and erased using Fowler-Nordheim tunneling. FG cells are arranged in an array in a NOR-like structure. A NOR memory array is shown in Fig. 2.2. The three terminal of FG cells are shared for the whole memory array. The shared gate terminal is called wordline (WL) and the shared drain terminal is called bitline (BL). Hot-electron injection in NOR flash memory is done at the drain terminal. In NOR memory, logical state "1" is represented by a positive/ neutral (tunneled) FG charge, while logical "0" is represented by a negative charge (fully injected) on the FG

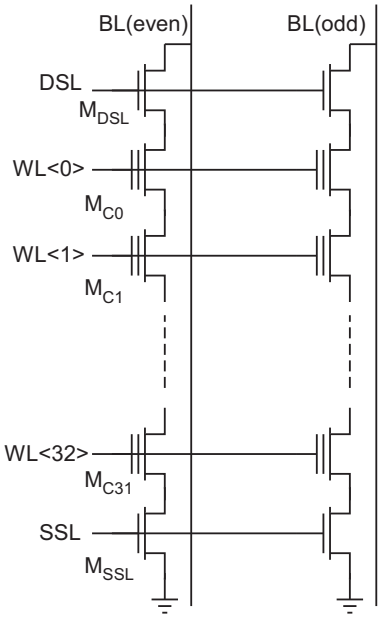

Figure 2.1: Matrix structure in NAND architecture

2.1.3 NAND Flash vs. NOR Flash

NAND and NOR flash memory are a digital application of FG transistor where a charge is stored on th floating-gate is read to be either 0 or 1. A comparison between the two flash memories is shown in Fig. 2.3. Overall, NAND flash cell is smaller than NOR flash cell; however, NOR flash memory allows for easy and faster access to every memory cell. This makes NAND flash memory suitable for higher densities and smaller die area applications such as USB drives and multimedia cards (MMCs). On the other hand, NOR flash memory are used for code storage and execution components such as low-end cell phones.

In term of memory access, NOR flash memories has enough address pins to map the entire array which allows easy access and makes NOR memory a random access memory. Alternatively, NAND flash memories are interfaced serially through a complicated I/O interface which makes NAND memories suitable for a file or a sequential data applications. Since NAND memory transistors are connected in series, read process is much slower than NOR memory. This is because the signal gets weaker through series transistors. NOR random access time is around $0.075\mu s$ while NAND serial access time is around $25\mu s$. However, NAND memory has a faster programming and erasing time [12]. A summary comparison between the two flash memories is shown in Table. 2.1.

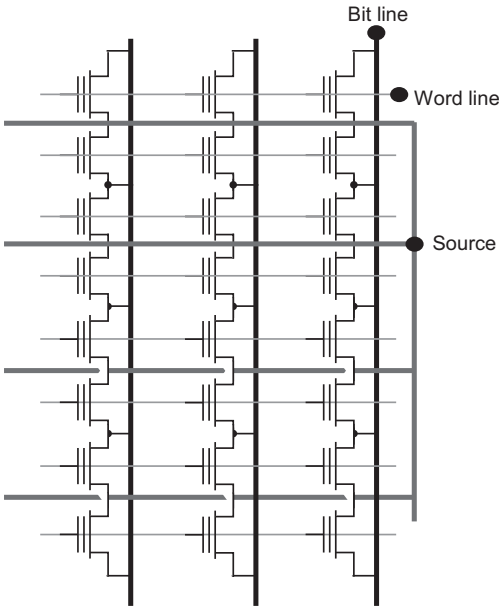

Figure 2.2: NOR Flash array equivalent circuit.

2.2 Floating-Gate Device Overview

A typical CMOS transistor consists of three basic terminals - gate, drain, and source, Fig.2.4 (a) . The voltage applied at the gate V_g controls the transistor operation, allowing the transistor to work as an on/off switch. Similarly, a floating-gate transistor has three basic terminals - control-gate, drain, and source. The gate terminal in FG transistors are referred to as control gate to distinguish them from the transistor's floating-gate.

PMOS-based FG transistors are the commonly used FG transistors. NMOS FG transistors are not preferred because of their low endurance, the capability of maintaining stored

Table 2.1: NAND Flash vs. NOR Flash

	NAND	NOR
Advantages	Fast Program	Random Access
	Fast Erase	Byte Programs possible
Disadvantages	Slow random access	Slow Programs
	Byte Programs Difficult	Slow Erase
Applications	File (disk) applications	Replacement of EPROM
	Voice, data, video recorder	Execute directly from Non-Volatile Memory
	Any large sequential data	

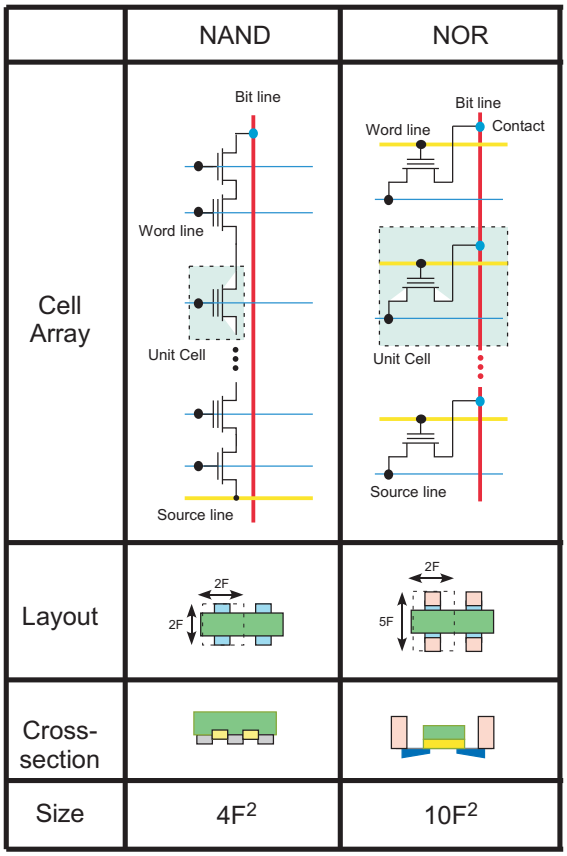

Figure 2.3: Comparison of NAND and NOR Flash Cells

information after erase/ program/read cycle, and retention, which is the capability of keeping the stored information in time.

A typical p-channel MOSFET floating-gate transistor consists of a pFET transistor where the gate is isolated by silicone-dioxide and does not have a direct connection to any voltage. A schematic and cross-sectional representation of both conventional pFET and FG-pFET is shown in Fig. 2.4. In a typical (MOSFET) transistor, the gate has a metal connection to apply a voltage at the gate and bias the transistor. As the figure Fig. 2.4 (b) shows, a floating-gate transistor has a second layer of gate oxide and polysilicon on top of the conventional pFET, which creates a second isolated and concealed gate, hence a floating-gate. The accessible gate of the floating-gate transistor is referred to as control gate V_{cg}, which is used similarly to V_g of the conventional pFET to operate FG transistor. The two gates are made of polysilicon, which is better for long-term charge retention than a metal material. As the two gate-plates form a parallel plate structure inside the FG transistor, it is schematically represented as a capacitor C_g. Lastly, V_{fg} is the effective voltage on the

floating-gate where the charge is stored.

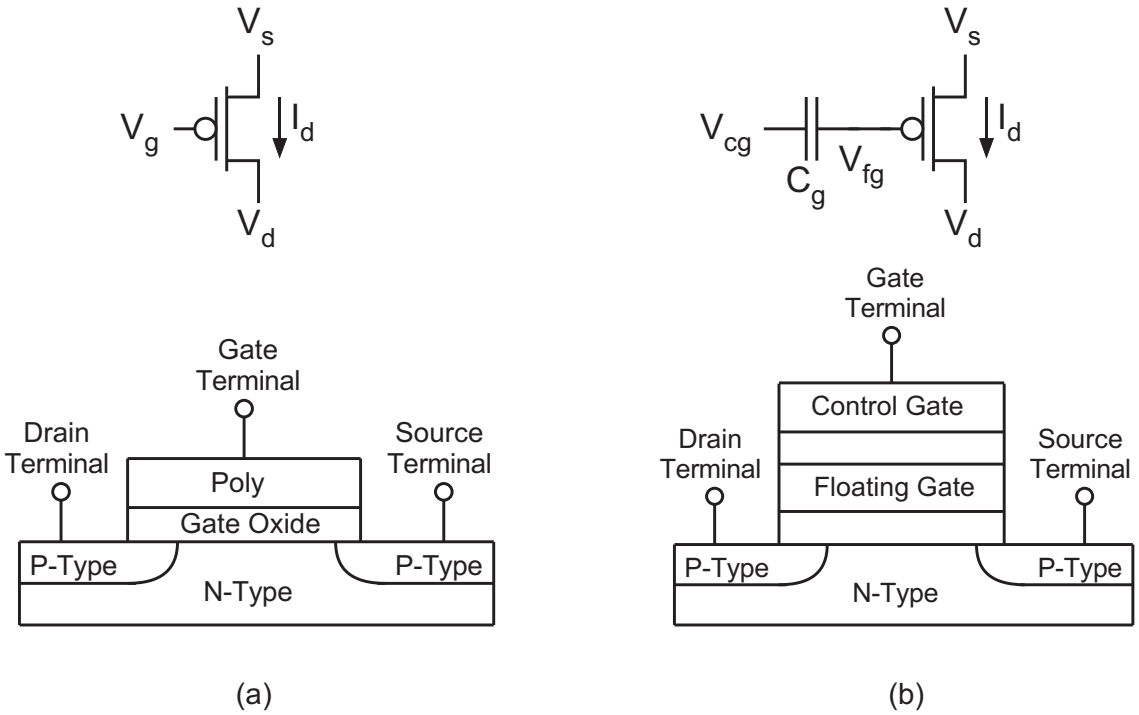

Figure 2.4: Schematic representation and physical cross-section of (a) a pFET transistor and (b) a FG transistor, respectively.

The charge on the floating-gate is permanently stored, as it is completely isolated with a high-quality insulator. The floating-gate transistor is capable of storing the charge for as long as tens of years with a minimal and insignificant charge loss [13, 14]. In [15], a floating-gate cell shows an accumulative long-term drift of 10 ppm/5th root of 1000 h.

The charge on the floating gate can be modified with the programming process by creating a potential condition to distort the band diagram so the charge can transfer to/from the floating gate. Two quantum mechanical processes are widely used: Fowler-Nordheim tunneling and hot-electron injection. Once the transistor is injected, the channel current i_d of FG transistor is a function of the amount of charge stored on the floating-gate node V_{fg} and the voltage applied at the control gate V_{cg}.

The effect of injection/ erasure processes on the functionality of the FG transistor is shown in Fig. 2.5. The figure shows the I-V curve of the FG transistor after injection and after tunneling. From the figure, after a transistor is injected (red curve), the IV-curve is shifted to the right; i.e. a smaller threshold voltage of the FG from the perspective of the control gate (which means the transistor could be easily turned on). As more negative

charge (electrons) are injected into the floating-gate, V_{fg} is pulled lower, which means lower V_{cg} voltage is needed to achieve the same pre-injection current.

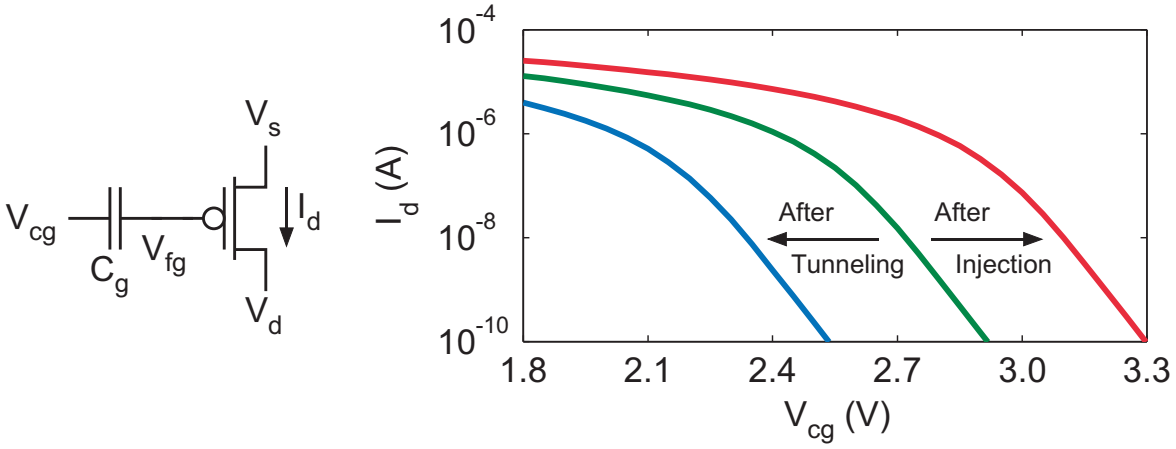

Figure 2.5: Floating-gate transistors allow for programmable threshold to shift. Injection programming adds electrons to the floating-gate, while tunneling removes electrons from the floating-gate.

2.3 Analog Application of FG Transistors

Floating-gate transistors (FG) were first introduced in 1967 as a non-volatile memory (NVM) device by Kahng and Sze [16]. FG transistors were initially used as non-volatile digital storage in erasable programmable read only memory (EPROM), electrically erasable programmable read only memory (EEPROM), and flash memories [17, 18, 19]. As the name suggests, non-volatile memory is a memory that holds its state after it loses power.

Since the amount of charge placed on the floating-gate can be controlled and adjusted to many different charge values, floating-gate transistors could be used as excellent analog memory devices. Initially, analog floating-gate applications used ultra-violet (UV) light to modify the charge on the floating-gate, which proved impractical for real-time adaption. Some of these early applications, summarized in [20], are Mead's adaptive retina circuit presented in [21], Shibata and Ohmi's neuron metal-oxide semiconductor field effect transistor (MOSFET) presented in [22], and Intel's electrically trainable artificial neural network (ETANN) presented in [23] that employs an array of FGs to create learning synapses . In 1991, Thomsen and Brooke presented a tunneling technique using a standard complementary metal-oxide semiconductor (CMOS) [24] that does not need a specialized flash fabrication

technique. Even though many of the first FG applications were biologically-inspired, more traditional analog application that utilized FG came after. For example, FGs were used to solve the problem of post-fabrication mismatch through trimmable current sources [25, 26] . Some other early analog applications are adaptive voltage tap for analog-to-digital converter [27] and input-offset compensation for amplifiers [28].

Programmable analog filter banks are another analog application of FG transistors. FG transistors are used as programmable current sources to bias and control corner frequencies of the programmable analog filter bank as shown in Fig. 2.6 [3, 29]. This filter bank utilizes FG transistors as current sources to change the transconductance of the two OTAs that are responsible to tune the two corner frequencies of the bandpass filter.

Analog-to-Digital Converters (ADCs) are circuits used to convert analog signals to digital words. Resistor mismatch in the resistive divider results in incorrect and non-linear voltage division. In [30], an array of adaptive FG comparators is presented. The array is used to separately store each reference voltage. The adaptive FG comparator presented in [31] used non-volatile memory for automatic offset cancellation. The system achieved an accuracy of 13-bit resolution. In [32, 33, 34], floating-gate transistors were used similarly as programmable voltage and current references.

The FG has been typically used for more traditional analog circuits problems such as offset correction [35], current trimming [25, 26, 36] and auto-zeroing [37, 28]. However, the need for the floating-gate transistors in analog applications started to move to more general reconfigurable systems by the 2000s. One type of reconfigurable analog electronics is field-programmable analog arrays ($FPAAs$). FPAAs contain programmable analog blocks that

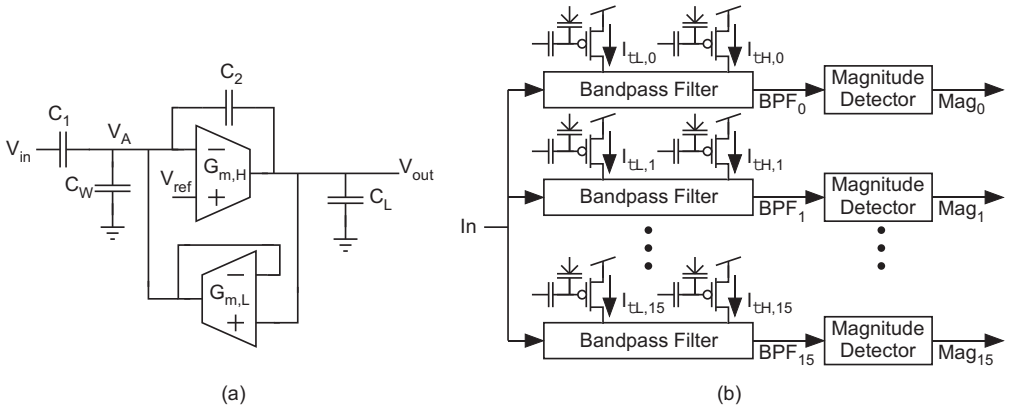

Figure 2.6: Block diagram of the BPF and the and the OTA based C^4 filter.

can be set up in any configuration to build analog systems. The FG transistors are used in FPAAs to provide biasing values, as non-volatile switches, and as programmable references. One example is the reconfigurable analog/mixed-signal platform (RAMP) presented in [2]. The floating-gate transistors are used in the RAMP as tunable current references to bias or modify circuit parameters like transconductance.

Generally, for analog applications, the FG memory cells are integrated on-chip with the analog application. However, to calculate the cost for a single FG memory cell, for 200mm wafer, we would be able to get more than 6 million FG memory cells. So the cost for individual FG memory cell would be $1/6 \times 10^6$ of the total cost of the wafer. However, the number of FGs per wafer and the wafer cost is process dependent, the smaller the technology the higher the cost.

In this work, I present different methodologies to inject the FG transistor arrays. The injection technique with better accuracy results and the least infrastructure overhead is to be used in the next iteration of the RAMP chip. Additionally, as a proof-of-concept, the FG arrays presented in this work are tested with a programmable bandpass filter bank.

2.4 Floating-Gate Transistor Structure and Operation

2.4.1 Floating-Gate Transistor Structure

The FG transistor used in this work is similar to the one shown in Fig. 2.4b. It consists of three components: a pFET transistors, a floating-gate, and a supplementary node called tunneling node. A schematic and a cross sectional model of the floating-gate transistor used in this work are shown in Fig. 2.7. From the schematic and cross sectional view, the first component of the FG transistor is the pFET transistor M_{fg} which is the left device in the cross sectional representation with its own N-well. The control gate of the transistor V_{cg} is implemented with a polysilicon-2 layer. The second component of the FG transistor is the floating-gate node V_{fg} which is realized with a polysilicon-1 layer. The final component is the second interface V_{tun} to the floating-gate node on the right side of the device. This node is used through the tunneling process (to erase the charge from the floating-gate node).

The structure of the tunneling node has a significant effect on the speed, power consumption, and long-term reliability of the FG transistor; therefore, a pFET MOSCAP is used to implement the tunneling junction for its efficiency and reliability features. In addition, a p^+

MOS capacitor in an N-well in 0.35μm process is more consistent and has a higher tunneling current than an n^+ MOS capacitor [38]. A pFET MOSCAP is a pFET transistor, M_{tun}, in which the source, drain and well connection are connected to each other. The cross section figure shows that the pFET transistor (left side) and tunneling junction M_{tun} (right side) share the floating-gate polysilicon-1 connection.

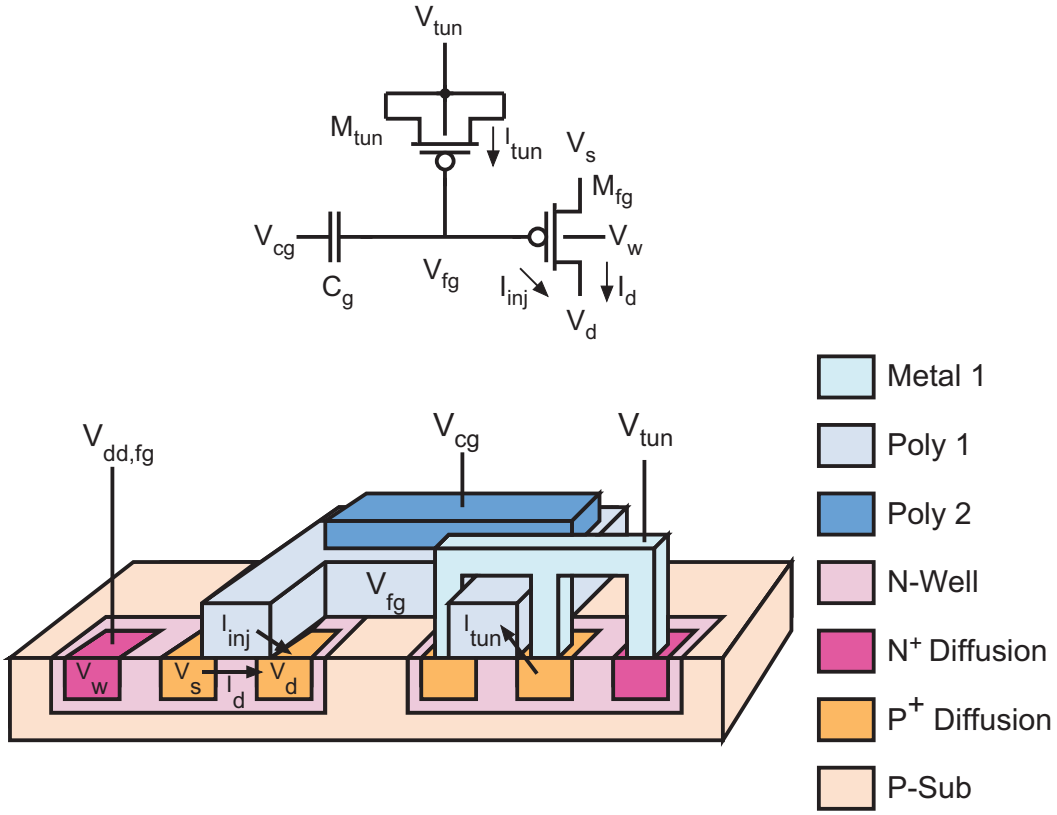

Figure 2.7: Physical cross-sectional representation of a FG transistor.

Fig.2.8 shows a physical cross-sectional representation of the FG transistor with all the parasitic capacitors.

2.4.2 Above Threshold Operation

Generally, above threshold operation provides high speed with the cost of power consumption. The FG transistor acts differently than the conventional MOSFET in the above threshold region. One different characteristic is that the FG drain current I_{DS} does not saturate and is dependent on the drain voltage. Also, the transconductance of the FG transistor

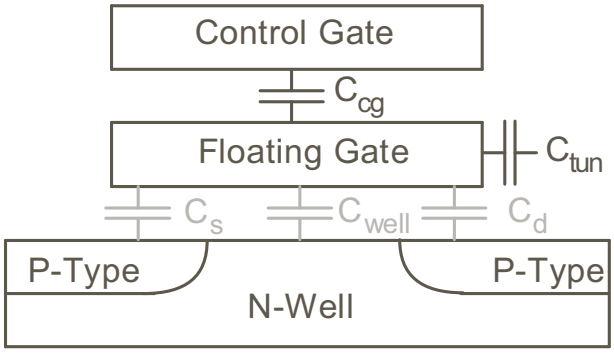

Figure 2.8: Physical cross-sectional layout of a FG transistor showing all capacitors coupled to the FG including parasitic capacitors.

g_m depends on the voltage V_{DS} and can be calculated by the following equation:

$$g_m = \alpha_G \beta (V_{GS} + fV_{DS} - V_T) \tag{2.1}$$

Some other distinguishing IV characteristics of FG transistor are: the capacitive coupling factor f depends only on C_D and C_{FC} and can be established as follows:

$$f = -\frac{\partial V_{GS}}{\partial V_{DS}} \tag{2.2}$$

Finally, because of the effect of fV_{DS}, the FG transistor can go to the depletion mode even when $|V_{GS}| < |V_T|$.

For the FG transistor to operate in the above threshold triode region $|V_{DS}| < \alpha_G |V_{GS} + fV_{DS} - V_T|$ the drain current is:

$$I_{DS} = \beta[(V_{GS} - V_T)V_{DS} - (f - \frac{1}{2\alpha_G})V_{DS}{}^2] \tag{2.3}$$

On the other hand, the condition for above threshold saturation region is $|V_{DS}| \geq \alpha_G |V_{GS} + fV_{DS} - V_T|$ and the equation for the drain current is:

$$I_{DS} = \frac{\beta}{2}\alpha_G (V_{GS} + fV_{DS} - V_T)^2 \tag{2.4}$$

Where the threshold voltage V_T and the conductivity factor β are measured from the prospective of the control gate.

2.4.3 Subthreshold Operation

When voltage (V_{cg}) is applied to the control gate, it couples through capacitor (C_g) to the floating-gate and causes a current I_d to flow through the transistor channel. As low-power application is the interest of this report, sub-threshold regime ($V_{sg} < V_{TH}$) is the best

area of operation to save the most power. Therefore, the current through typical MOSFET transistors in sub-threshold regime can be represented by the following equation:

$$I_d = I_0 \frac{W}{L} \exp\left(\frac{\kappa V_g}{U_T}\right)\left[exp\left(-\frac{V_s}{U_T}\right) - exp\left(-\frac{V_d}{U_T}\right)\right] \tag{2.5}$$

Where:

I_0	pre-exponential current scaler;
$\frac{W}{L}$	FET dimensions;
κ	subthreshold slope;
V_g	gate voltage;
U_T	thermal voltage $\approx 26mV$ at room temperature;
V_s	source voltage;
V_d	drain voltage;
V_A	Early voltage.

For an FG transistor, V_g in Equation 2.5 is replaced by the effective floating-gate voltage V_{fg} which equals the charge on the FG, Q_{fg}, divided by the total effective capacitance plus the voltage coupled from node V_{cg}:

$$V_{fg} = \frac{Q_{fg} + C_{cg}V_{cg} + C_dV_d + C_sV_s + C_{tun}V_{tun} + C_{well}V_{well}}{C_{total}} \tag{2.6}$$

Where:

Q_{fg}	total charge present on the FG;
C_{cg},C_{tun}	node capacitances (see Fig. 2.8);
C_s,C_d,C_{well}	parasitic capacitances (see Fig. 2.8);
C_{total}	sum total of all capacitances coupled to the FG.

As C_g dominates C_{total}, V_{fg} can be simplified as

$$V_{fg} = \frac{Q_{fg} + C_gV_{cg}}{C_{total}} \tag{2.7}$$

Therefore, substituting 2.7 into 2.5 yields

$$I_d = I_0 \frac{W}{L} \exp\left(\frac{\kappa(Q_{fg} + C_gV_{cg})}{C_{total}U_T}\right)\left[exp\left(-\frac{V_s}{U_T}\right) - exp\left(-\frac{V_d}{U_T}\right)\right] \tag{2.8}$$

As V_d is typically connected to a low voltage, it could be neglected in 2.8 and therefore drain current can be simplified to

$$I_d = I_0 \frac{W}{L} \exp\left(\frac{\kappa(Q_{fg} + C_g V_{cg})}{C_{total} U_T}\right)\left[exp\Big(-\frac{V_s}{U_T}\Big)\right] \tag{2.9}$$

As can be seen from Equation 2.9, any change in Q_{fg} would result in change in the drain current and the threshold voltage of the floating-gate transistor. This affect is illustrated in Fig. 2.5c, in which the threshold voltage decreases after programming and increases after tunnelling.

2.5 Floating-Gate Charge Modification

To place charge on the floating-gate, different techniques were used. In this work, we are referring to the process of adding electrons to the floating-gate as "FG programming," while the reverse process of erasing the charge from the floating-gate is referred to as "FG tunneling." Originally, Fowler-Nordheim tunneling was used for FG programming and UV light was used to tunnel FG devices. Depending on the type of the FG device, some devices could be programmed, tunneled, and reprogrammed many times, known as (EPROM), erasable programmable read-only memory. At a later time, hot electron injection started to become more popular to re-program RPROM devices. With the use of hot electron injection for programming came the process of electrically programmable and erasable devices (EEPROM). EEPROM used an array of addressable FGs in a NOR flash configuration, utilizing hot electron injection for the selective programming of a specific FG.

2.5.1 Tunneling and Electron Removal

In tunneling process, a high voltage is applied at the tunneling connection, M_{tun}, in Fig. 2.7 [39] to distort the band diagram and reduce the effectiveness of the oxide barrier, so the trapped electrons on the floating-gate can tunnel through the oxide layer. This tunneling process is called Fowler-Nordheim tunneling, in which electrons tunnel through the oxide with the effect of a large electric field. The oxide layer thickness determines how high the tunneling voltage, V_{tun}, should be to successfully perform tunneling. For M_{tun} with relatively thick oxide, a voltage of as high as 30V is required to tunnel the floating-gate. The thinner the oxide barrier, the lower the required V_{tun} [10]. To lower the voltage required to perform the

tunneling process, a simple pFET MOSCAP M_{tun} is used due to its thinner oxide compared to a typical poly-insulated-poly capacitor.

The flow of electrons across the oxide barrier (tunneling current) can be expresses by the following equation [38]:

$$I_{tun} = \alpha exp\frac{-\beta t_{ox}}{V_{ox}} \tag{2.10}$$

Where:

t_{ox}	oxide thickness;
V_{ox}	voltage across the oxide (voltage difference between V_{tun} and V_{fg};
α and β	device fits that depend on the device type and CMOS process.

As equation 2.10 shows, the thinner the oxide layer the higher the tunneling current. For small CMOS processes with a thin oxide layer, high voltage I/O devices are used to prevent undesired tunneling current. Fig. 2.9 shows the varying V_{tun} voltages needed for tunneling over different CMOS technology. The top pane shows that the voltage required for V_{tun} is way higher than the rated supply voltage V_{dd} for each process. The bottom pane of Fig. 2.9 shows the ratio (V_{tun}/V_{dd}); this ratio is an important factor for designers to design a voltage step-up circuit that pumps up the supply voltage, V_{dd}, to generate the required high tunneling voltages. This ratio is smaller for larger devices and higher for smaller devices due to the use of higher voltage I/O devices in smaller process.

Since the voltage required for tunneling is much higher than the rated supply voltage for the CMOS process, it is difficult to isolate a single FG cell in an FG array; therefore, FN tunneling is used to globally erase the charge from all FG transistors on the same chip at the same time.

2.5.2 Hot-Electron Injection and Electron Addition

Injection is the process of adding electrons on the floating-gate node of the floating-gate transistor. Hot-electron injection refers to the technique of electrons being injected into the floating-gate node by gaining enough energy to overcome the oxide barrier from the channel interface. Two conditions are essential to successfully performing hot-electron injection: 1) high channel current through the transistor channel, and 2) large potential differences between the source and the drain, V_{ds}.

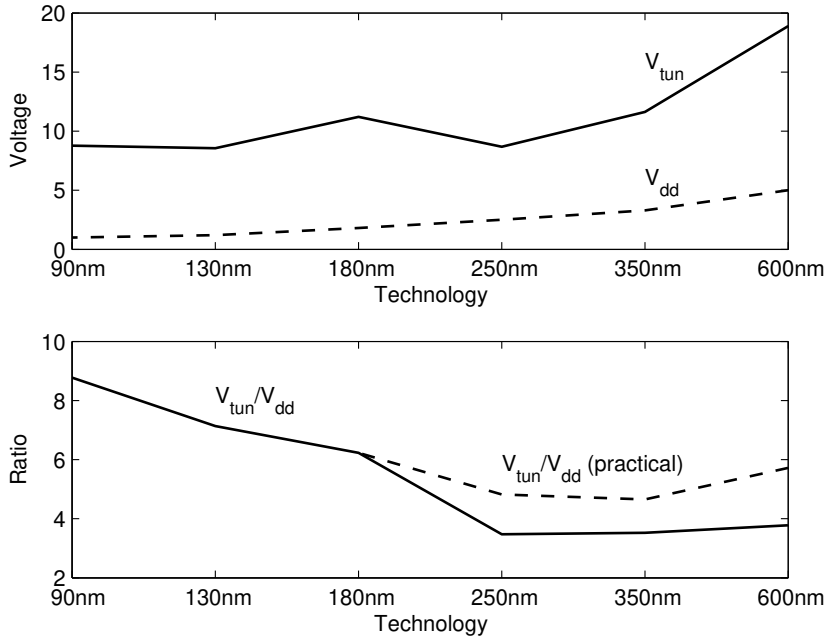

Figure 2.9: The required V_{tun} and V_{dd} for scaling CMOS technologies. (Top) shows scaling of V_{tun} and V_{dd}. (Bottom) the ratio of V_{tun} to V_{dd}.

When applying high (V_{ds}), a strong electric field is created at the area between of the drain depletion region and the channel. This strong electric field causes the majority carrier at the drain (conducting holes in pFET) to become highly energized and collide with the lattice. These collisions generate an impact-ionized hot electron-hole pair. Part of those ionized electrons have high energy levels enough to escape the channel and travel through the oxide layer to the gate electrode. Hot-electron injection current in pFET floating-gate transistors can be calculated as follows

$$I_{inj} \approx \beta I_d^{\alpha}(e^{V_{sd}/V_{inj}}) \tag{2.11}$$

Where:

I_{inj}	the injection current;
I_d	the drain current;
α, β and V_{inj}	device-dependent fits [40].

As equation 2.11 shows, injection current depends on the two aforementioned factors I_d and V_{ds}. This injection equation is for the FG transistor in the subthreshold region, as

injection efficiency is better for subthreshold current [41]. The voltage V_{sd} is to be raised to a value higher than the rated supply voltage depending on the device process. The bigger the process, the higher the V_{sd} required for injection. Fig. 2.10 shows how standard CMOS process and oxide thickness affects $V_{sd,inj}$ needed for injection. The top plot shows that injection voltage V_{sd} scales with technology process. $V_{sd,inj}$ continues to increase for larger process; this is due to the increase in the source and drain nodes depth as the process becomes bigger [42]. The process of rising V_{dd} to $V_{sd,inj}$ during the injection process is called ramping up. A charge pump circuit is used to "ramp up" the supply voltage to the injection voltage.

In Fig. 2.10, the bottom plot shows the ratio ($V_{sd,inj}/V_{dd}$). For large technologies, the ratio is smaller; this is due to a larger supply voltage V_{dd}. This step-up ratio is used to design the charge pump circuit necessary for injection. For smaller processes, the step-up is lowest and relatively constant; this is because for processes smaller than $250nm$, higher-voltage I/O devices are used to minimize floating gate leakage.

Both charge modification techniques, hot-electron injection and Fowler-Nordheim tunneling, require a voltage higher than the rated supply voltage. However, as the voltage required to perform hot-electron injection is significantly lower than the voltage required to perform FN tunneling, hot-electron injection is the preferred method to inject a single FG transistor in an FG array. On the other hand, Fowler-Nordheim tunneling is the preferred method to globally erase all FG transistors on one chip at the same time. Once injected, a FG transistor can hold the charge for a very long period of time. Charge loss is recorded to be as low as 1% over a period of 10 years [43, 35]. For non-volatile analog memory applications, pFET based floating-gate transistors are preferred. This is because of the direction of the field lines in nFETs that makes the injection process more difficult to control.

2.6 Programming Methodologies

Different programming techniques can be used to program floating-gate transistors depending on the application and the designer. The floating-gate transistors are used as infrastructure components for different analog applications, therefore an additional circuit is needed to perform injection. In this work, we refer to this circuit as programmer circuit. A programmer circuit is a supplementary circuit used to perform injection on a FG on a chip. Programmer circuits are designed to be accurate, fast, linear, and have low-power consumption and low infrastructure overhead. In most cases, a programmer circuit is integrated

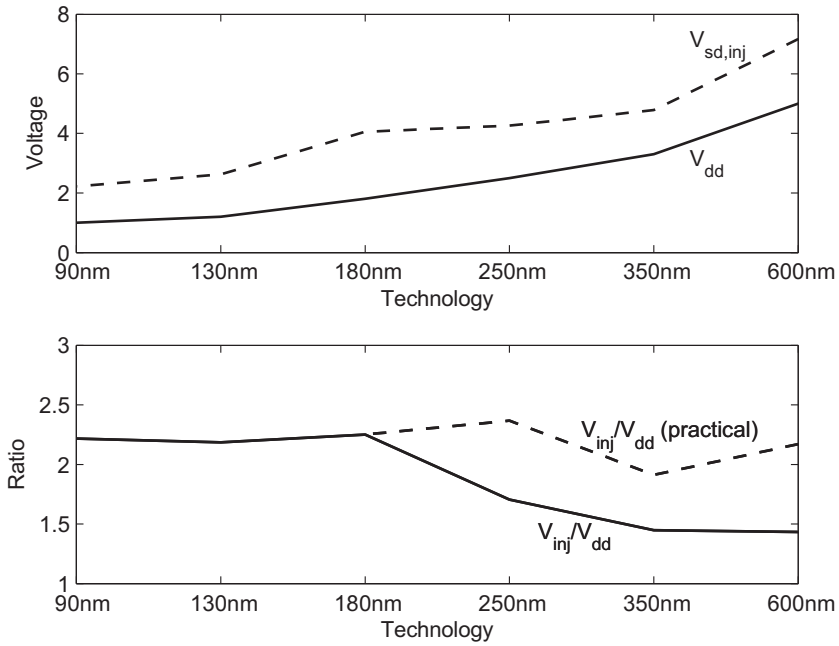

Figure 2.10: Comparison of the required V_{inj} voltage for scaling CMOS technologies. Top plot shows that V_{dd} and V_{inj} scale similarly over the downsizing of technologies. Bottom plot shows the ratio of V_{inj} to V_{dd}. In practice, the supply V_{dd} should be kept lower than standard V_{dd} to prevent unwanted HEI. Taking this into consideration, the ratio stays around two over all technologies [1]

.

on-chip to program an array of floating-gate transistors. The main purpose of programmer circuits is to control the flow of injection electrons to the floating gate node and to stop the programming process once the target voltage/ current is achieved.

There are two main programmer schemes generally used to program FG transistors: pulsed programming and continuous-time programming.

2.6.1 Pulsed Programming

Pulsed programming is a simple method to program a floating-gate to a certain voltage. In pulsed programming, short programming and reading intervals are applied alternatively until a target current is read through read interval. The programming process is ended when the target current or voltage is observed. In the programming interval, the FG transistor is being programmed for a short time while in read interval and the output current or voltage is measured to evaluate the effect of the preceding programming interval. Pulsed programming is shown in Fig. 2.11a. Programming pulse duration and programming voltage V_{sd} control

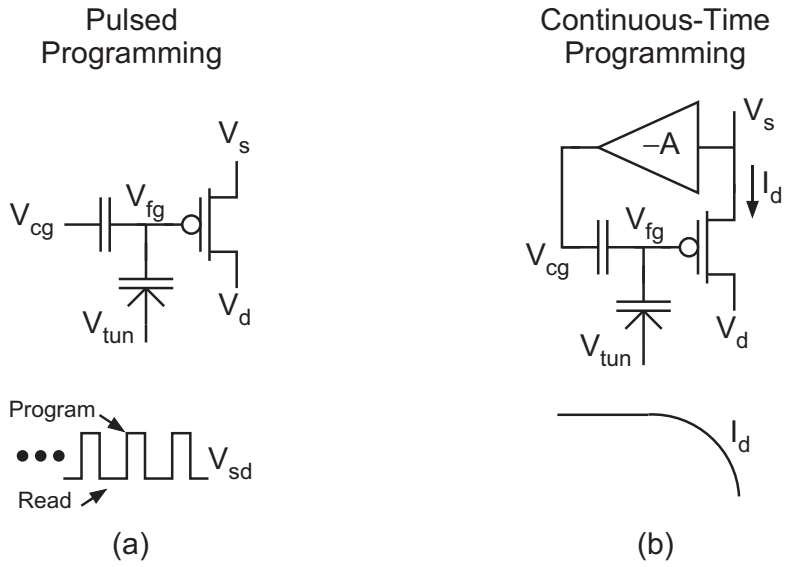

Figure 2.11: (a) Pulsed programming (b) Continuous programming.

the injection rate during programming intervals.

Pulsed-based programming has high accuracy programming as the length of the programming interval can be changed, and thus the amount of charge (injection rate) in the programming interval can be adjusted. Additionally, current measurement in read mode is done in conditions similar to run-mode (i.e. nominal V_{sd} voltage); this will further improve programming accuracy. The pulsed programming approach presented in [32] provides an accuracy of $>$ 13 bits. However, pulse programming has few issues. One big limitation of this method is that it requires a large circuitry to switch between reading and programming mode which is not suitable for dense analog memory arrays. In addition, programming speed is limited by the length of program/ read intervals and by how big the target current is (the higher the target current, the longer it takes to program one FG cell, especially when higher accuracy is required) . With this higher programming accuracy requirement, a smaller programming interval is used to inject a smaller charge amount into the floating-gate node. This will increase the number of programming/ reading pulses which consequently increases the overall injection process.

To solve this issue, an optimal injection rate is calculated to minimize programming time [41]. In this approach, calibration mode is used first to characterize the injection rate, and then a mathematical model is used to summarize the characterization data into few parameters required to choose the optimal V_{sd} for programming. The calibration mode

makes this approach complicated and not suitable for large FG arrays. Furthermore, this approach is not suitable for portable applications as high-precision converters and wide-range current measurements are required.

2.6.2 Continuous Programming

Continuous programming is performed in one single programming period as shown in Fig. 2.11(b); after which the FG cell is placed in run mode to read the programmed current/voltage. Programming process length is controlled using a negative feedback path and is stopped when the floating-gate transistor reaches its target. Using negative feedback is also beneficial to ensure a constant V_{sd} that linearizes the injection process which provides the condition for predictable injection rate. With the use of the single programming period and linear injection process, continuous programming is faster than pulsed programming and provides predictable injection results, which makes it the more favorable injection method.

Pulsed programming as well utilizes negative feedback to linearize the injection process. [32, 13] used a picoammeter, on-chip transimpedence amplifier, and off-chip analog-to-digital converter to achieve linear injection. However, the circuit presented in [13] needs supplementary circuits such as a field-programmable gate array (FPGA) or microprocessor (μP) to index measured values from preceding periods and calculated values for the next periods.

Employing negative feedback in continuous programming is less complicated and more efficient. For example, a self-converging single transistor circuit is presented in [40] and shown in Fig.2.12(a). In this configuration, inherent feedback stops injection once the programming target is reached. However, the continuous change in V_{fg} during programming is causing I_{inj} to change, thus there is a nonlinear injection rate.

To solve the nonlinearity problem in continuous programming, one approach used a closed-loop amplifier [32]. An inverting amplifier is used as the feedback bath between the source and the control-gate of the transistor Fig.2.12(c) and (d) . In this scheme, the feedback inverting amplifier holds V_s constant through compensating the charge added on the floating-gate by increasing V_{cg} which allows for a constant injection rate (i.e. linearity). A similar but compact technique is presented in [44], where the feedback opamp is replaced with a single pFET common-source transistor between V_s and the control-gate V_{cg}. This configuration is further discussed and is utilized to build a FG memory array in chapter 3. This configuration keeps V_{fg} constant by adjusting V_{cg} during programming, resulting in

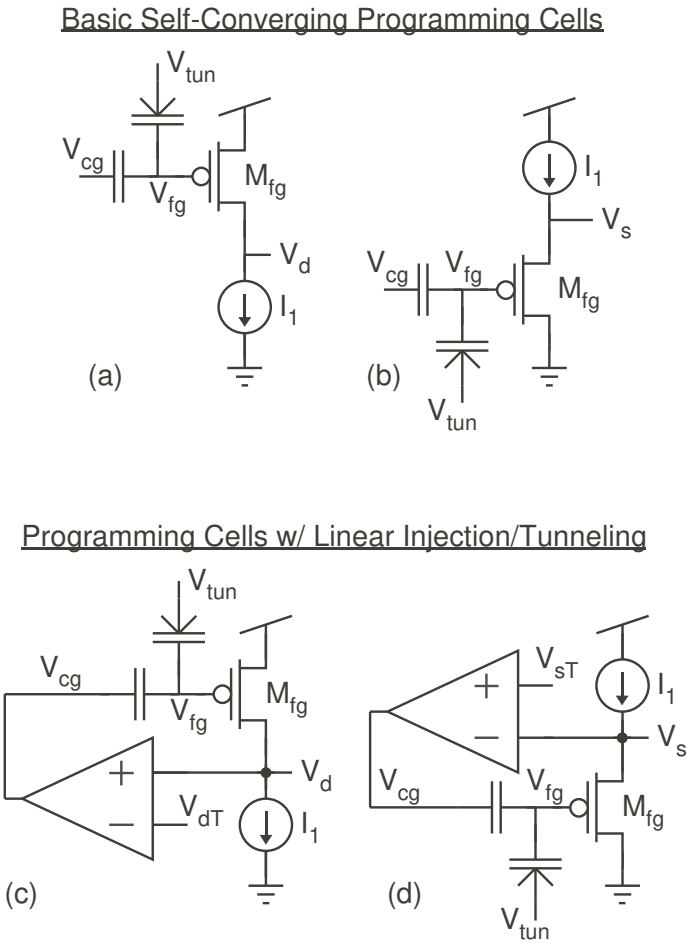

Figure 2.12: Continuous programmers. (a) and (b) Self-converging single transistor injection configuration. (c) and (d) Operational amplifier feedback to allow linear injection.

linear and predictable injection.

Some other circuits used feedback to achieve linear injection are: [45] in which a comparator is used with a three-transistor memory to stop injection once the target is reached; [46] which presents a programmer circuit that uses a differential FG amplifier to linearize tunneling and cancel out the tunneling junction's capacitive coupling; [47] presents a memory cell that uses both hot-electron and hot-hole injection to converge bidirectionally toward the target.

2.6.3 Serial vs Parallel

As discussed in sections 2.6.1 and 2.6.2, a programming process could be either pulsed or continuous. However, there is another programming characteristic that differentiates

programming methodology: serial or parallel programming. FG transistors are used in many applications in an array configuration that consists of many FG memory cells. Therefore, a technique to control the process of programming those many FGs is needed. As their names imply, serial and parallel programming are methods to program a number of floating-gate transistors in series or parallel manners.

In serial programming, FG transistors are programmed one transistor at a time. This would require only one programmer circuit per chip as only one FG is being programmed at any time. However, to program a large array of N floating-gate transistors, N programming cycles are required, which makes this method slower than parallel programming.

On the contrary, parallel programming employs programming a number of floating-gate transistors simultaneously, which makes this method much faster than serial programming. One programming cycle is needed to program N floating-gate transistor. However, to program N FGs at the same time, N programmer circuits are need on-chip. This would make this method not practical when the area is a restriction, especially for dense systems with considerably large numbers of FGs.

2.7 Reading Operation

After the FG transistor is injected, the floating-gate transistor is placed under read mode. This is done by lowering the supply voltage V_{dd} to the process rated supply voltage, applying a voltage to the control gate V_{cg} of the FG transistor and measuring the current flowing through its channel. As mentioned before, the current in read mode depends on the charge stored on the floating-gate node Q_{FG} and the voltage applied at the control gate V_{cg}. Assuming the FG transistor is already injected and a charge is stored on the floating-gate node, the read-mode current can be expressed as:

$$V_{FG} = \alpha_G V_{GS} + \alpha_D V_{DS} + \frac{Q}{C_T} \tag{2.12}$$

$$V_T{}^{\mathrm{CG}} = \frac{1}{\alpha_G} V_T{}^{\mathrm{FG}} - \frac{Q}{C_T \alpha_G} \tag{2.13}$$

$$I_{DS} = \beta[(V_{GS} - V_T - (1 - \frac{1}{\alpha_G})\frac{Q}{C_T})V_{DS} + (f - \frac{1}{2\alpha_G})V_{DS}{}^2] \tag{2.14}$$

To achieve a certain read mode current, for example 1μA, a lot of different Q_{fg} and V_{cg} combination can be used. For example, after injection and placing a specific charge Q_{fg1} a control gate voltage of V_{cg1} is required to achieve a read mode current of $100nA$, while we can achieve the same current by injecting the FG transistor with more charge Q_{fg2} and lowering the control gate to V_{cg2}.

2.8 Summary

The FG transistors are widely used for both digital and analog applications. Programmable current sources is one important analog application of the FG transistor. An FG transistor is a pFET transistor which gate does not have a direct connection to any potential. The gate is isolated with a capacitor to keep the charge stored on the floating-gate for a long time. Another connection to the floating-gate node is the tunneling MOSCAP. The charge on the floating-gate is modified using two main processes: programming and tunneling. Hot electron injection is used to program individual FG transistor on the chip, while Fowler-Nordheim tunneling is used to perform global tunneling for all the FG transistors on the chip. Higher voltages are required to perform injection/ tunneling, therefore a step-up charge pump circuit is used to multiply the supply voltage by a constant and generate the required voltages.

For analog applications, floating-gate transistors mostly used in array architecture and a programmer circuit is needed on-chip to perform injection. In the following chapters, I present different floating-gate memory cells and how each cell can be utilized to build a FG memory array to be used in different analog applications. Furthermore, different programmer circuits and techniques are designed, fabricated, and tested. Finally, those designs are tested for accuracy and practicality with application circuits.

2.8.1 Objective of this research

The main objective of this work is to facilitate the implementation of FG transistors in reconfigurable low-power analog applications. Field-programmable analog arrays (FPAAs) need current sources to bias programmable analog circuits. Instead of using on-chip current sources to set programmable parameter, FG transistors are used. FG transistors can be used as programmable current sources to provide accurate biasing current for different FPAA

components. Injecting FG transistor with a precise charge is the biggest challenge of using FG in FPAA. This is due to the low-power requirements of FPAA. Also, obtaining repeatable results with high accuracy in large scale systems is challenging. Additionally, a compact and power-efficient programmer circuit is needed to linearize the injection process.

In the next chapters, my work addresses those challenges and I present FG programming methods to achieve repeatable and accurate programming results with minimal power consumption and minimal infrastructure overhead. In chapter 3, I present an FG cell with linear-injection characteristic that is used on the RAMP chip [2] along with programmer circuit that achieved very accurate, repeatable results. The circuits I present in chapter 4 employ different injection methods that utilize negative-voltages to perform injection. I exploited the indirect FG memory cell configuration to make below-ground injection possible in FG arrays. The circuits presented in 3 and 4 address the challenge of linearizing the programming current and the challenge of injecting a specific FG in FG-dense application. Additionally, those circuits provide accurate programming results as high as 13 bits. In chapter 5, I present a prototype programmer circuit that is based on the circuits presented in chapters 3 and 4.

Chapter 3

Floating-Gate Above Ground Injection

For analog applications, floating-gate transistors are mostly used in a dense configuration, for example, to set the corner frequency for a bandpass filter bank. In this chapter, I present a compact floating-gate memory cell that can be used to construct a large FG array. I also present a programmer circuit that is used to inject the FG array in a continuous parallel injection methodology.

3.1 Floating-Gate Injection

Two distinct injection techniques were discussed in 2.6.1 and 2.6.2, pulsed programming and continuous-time programming. Pulsed programming was shown to be accurate but it requires large circuitry to switch between program/ read mode and it takes longer time to converge especially when programming for higher targets in FG-dense applications. Alternatively, continuous-time programming is faster and provides linear injection with predictable results. Two different continuous-time programming FG cells are shown in Fig. 3.1 and Fig. 3.2. Fig. 3.1 shows a basic self-converging cell that stops programming once the target is reached. When injection starts, the floating-gate node voltage V_{fg} will continue to decrease, causing V_{sd} to decrease and therefore decreasing I_{inj} [40]. To set the injection target, different I_1 values are used with constant V_{cg} or a constant I_1 is used with different V_{cg} values. The problem with this self-conversing cell is that convergence time depends on the initial condition on the floating gate node, thus if V_{fg} was initially high yielding to a small initial

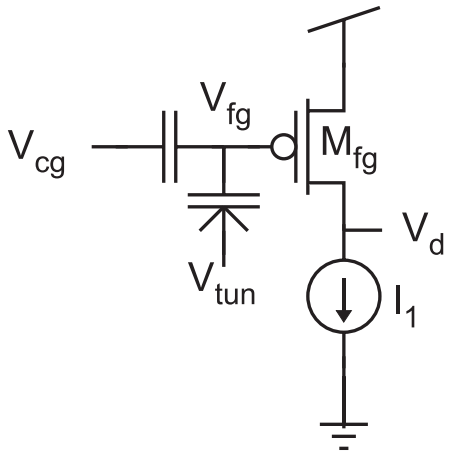

Figure 3.1: Continuous programmer self-converging single transistor injection configuration.

drain current, therefore convergence can take up to several seconds.

A negative feedback can be used to solve the slow convergence of a self-converging FG cell. The circuit shown in Fig. 3.2 (a) uses a negative feedback amplifier to linearize injection process by holding all M_{fg} terminals constant throughout injection. During injection process, V_{fg} is held constant and V_{cg} ramps up to compensate the charge change on the floating-gate node Fig. 3.3. The source follower configuration shown in 3.2 (b) is the same as that used in pulsed programming and achieved 13-bit precision and programming times on the order of 50sec/200mV [32]. While this configuration provides linear injection and gives accurate results, additional programming circuitry is needed to stop injection once the target is reached.

As discussed in section 2.5.2, in order to start hot-electron injection, two conditions are required: high channel current I_d and high source-to-drain voltage V_{sd}. The source-to-drain voltage V_{sd} is to be higher than the rated process supply voltage but less than the junction breakdown voltage. To achieve high V_{sd}, the drain is kept at ground and the source voltage is pulled above the supply V_{dd}. This high voltage is applied as a pulse for a duration long enough to complete the injection process. The injection pulse FG_{dd} is generated by multiplying V_{dd} with a factor using charge-pump circuit.

3.2 FG Memory Cell

The FG memory cell we present here has the same good characteristics as 3.2 (a) but with compact configuration that is suitable for FG-dense applications. In this memory cell,

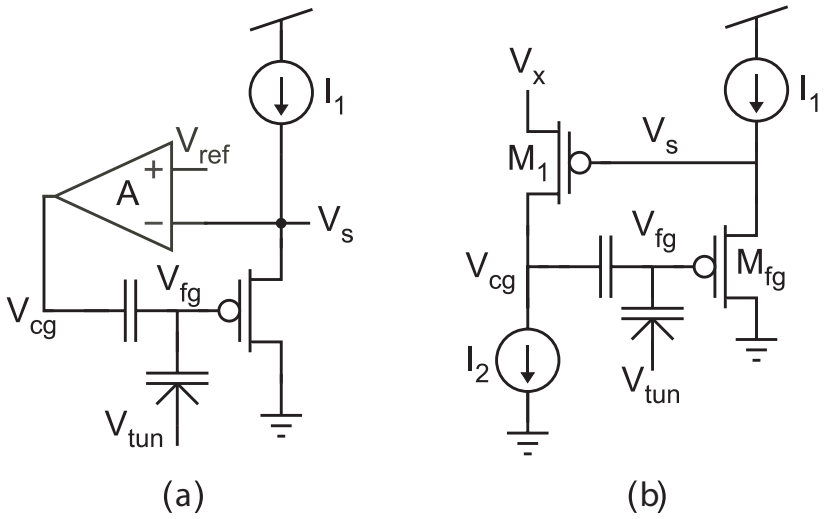

Figure 3.2: Continuous programmers. (a) Operational amplifier feedback to allow linear injection(b) Current-controlled current conveyor structure that allows for independent control of V_s and consequently V_{sd} via negative feedback through V_x and I_2.

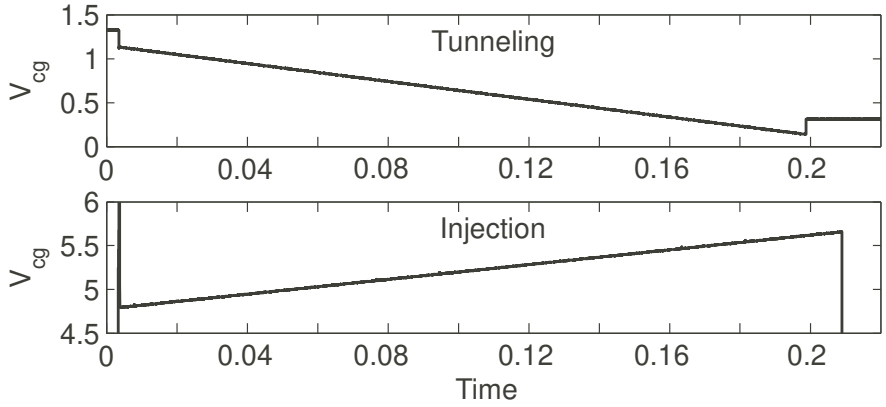

Figure 3.3: Demonstrate the linear programming characteristics of the circuit in Fig 3.2 (a).

a single pFET common source transistor is used as the feedback path between the source V_s and the control-gate V_{cg} 3.2 (b). This configuration is called current-controlled current conveyor which was originally described in [48]. However, this configuration was first applied in an FG programmer circuit presented in [44]. Similar to 3.2 (a), the negative feedback is used to keep V_s and V_{fg} constant throughout injection by adjusting V_{cg}. While V_s is constant, it still can be controlled by both V_X and I_2. On the other hand, V_{fg} can be controlled by the voltage V_s and the current I_1. Therefore, the injection current I_{inj}, in subthreshold can

be written as a function of the control terminals:

$$I_{inj} \approx \beta {I_1}^{\alpha} \left(\frac{I_2}{I_0}\right)^{-\frac{U_T}{\kappa V_{inj}}} e^{\frac{V_x-(1-\kappa)V_{dd}}{\kappa V_{inj}}} \tag{3.1}$$

Where:

I_0	the pre-exponential current scaler for M_1;
κ	the subthreshold slope for M_1.

The structure is referred to as "current- controlled current conveyor" because the current I_1 sets V_{cg} through transistor M_1 (i.e. current controlled) and the current I_2 is conveyed to the voltage V_X. This configuration provides a more suitable and compact design to utilize FG transistors in FG-dense applications that contain hundreds of FGs such as FPAAs [2, 49].

3.3 Programmer Circuit

The FG cell presented in the previous section 3.2 provides linear injection of FG. However, additional programmer circuit is needed to end the injection process once the target voltage is reached. The programmer circuit presented here monitors the change in V_{cg} as injection proceeds and compares the control gate voltage V_{cg} with the target voltage and consequently stops the injection process when V_{cg} reaches V_{targ}. Fig. 3.5 shows the programmer circuit presented in this work. The circuit in the figure includes the FG memory cell presented in 3.2, the programmer comparator feedback OTA (operational transconductance amplifier), and the current mirror M_2 - M_3. This configuration controls the injection process by adjusting the current I_1 flowing through FG transistor during programming. The programmer OTA works as a voltage comparator that converts the difference between the V_{cg} and V_{targ} into current I_1.

The programmer OTA is used here as a comparator to compare the difference between the FG cell V_{cg} and the target voltage V_{targ} and converts the difference to current that controls the FG injection rate. This OTA is 5-transistors OTA, and a schematic of the comparator OTA is shown in Fig. 3.4. When injection starts, V_{targ} is way higher than V_{cg}; this causes the OTA output current to saturate. The current mirror M_2-M_3 is used here to rectify the OTA output current into the FG transistor. The saturated output current is forced into the FG transistor through the current mirror resulting in a constant injection rate. As injection continues, V_{cg} will starts to increase linearly, which causes the OTA output current to decrease consequently. Lower current I_1 will result in lower injection current I_{inj};

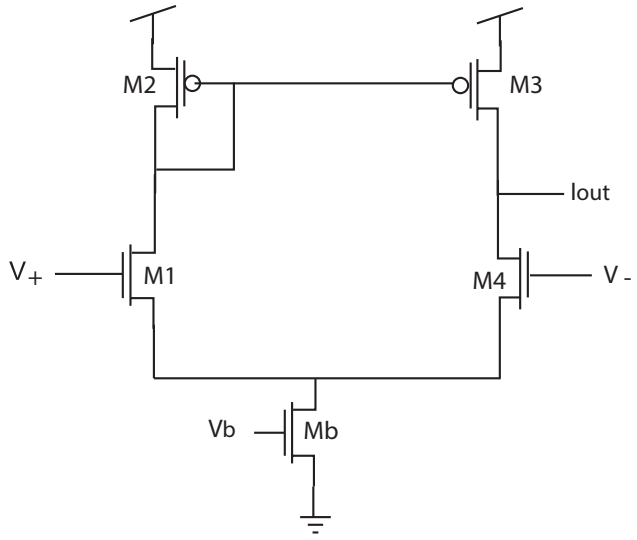

Figure 3.4: Programmer OTA structure.

therefore, the injection rate will decrease. As V_{cg} approaches V_{targ}, and consequently the OTA $V^+ > V^-$, the OTA output is pulled high, which places the pFETs in the current mirror M_2-M_3 in cuttoff. Consequently, V_s is forced to be low, and the negative feedback common-source amplifier M_1 no longer will have the proper biasing condition to operate as an amplifier to compensate the negative charge on the FG node. As a result, I_1 is to be set to zero, which eventually will end the injection process.

A transient demonstration of the programmer circuit from [44] is presented in Fig. 3.6. The top plot illustrates V_{targ} and V_{cg} as programming process progresses. The bottom plot displays the current I_1 through the FG transistor. Before injection starts, assuming that initially FG is tunneled with no charge on the floating gate node, the supply voltage is set to be at run-mode level (3V), so V_s is going to be around 2.6V. And as the FG is not connected to the programmer yet, V_{cg} is at ground and current I_1 is zero. This is displayed in interval (i). When the injection process starts in interval (ii), the supply voltage is ramped up (5.4V) to provide high V_{sd} necessary for injection. V_{cg} pulled up and experienced a small discharge through I_1 for a short time. Once discharged, V_{cg} starts to linearly rise and I_1 is saturated to the OTA output current. Once V_{cg} reaches V_{targ} interval (iii), the OTA current is shut off, causing injection to stop and pulling V_{cg} up to the supply voltage. This completes the injection process.

Once the injection pulse is concluded, the floating-gate transistor is placed in read-mode

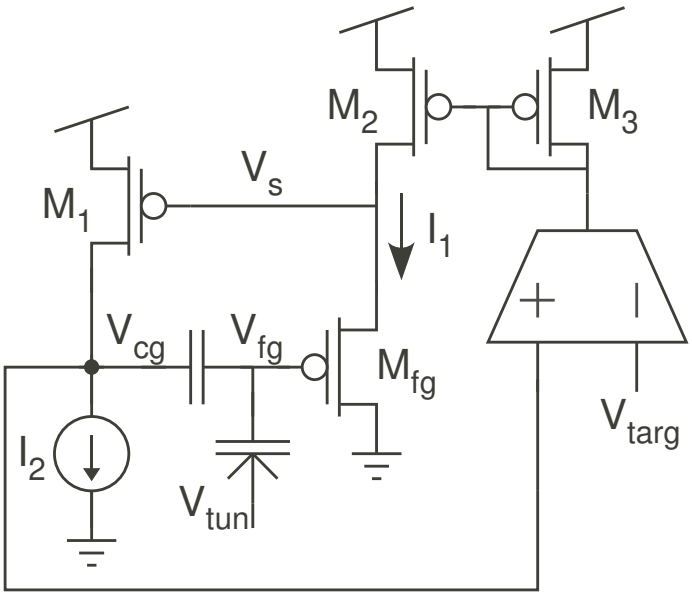

Figure 3.5: Continuous-time programmer with comparator OTA and current mirror M_2-M_3.

interval (iv). The supply voltage is ramped down to the process supply. The comparator OTA is disconnected and a constant voltage is applied at the gate of transistor M_2 to configure the FG transistor as voltage reference cell. The output voltage is read at node V_{cg}. However, to configure the cell to operate as a programmable current source, a constant voltage is applied at V_{cg}, and the drain terminal is connected to either a picoammeter to read the current or the application circuit to be biased. The output current depends on both the charge stored on the floating-gate node (i.e. V_{targ} in programming mode) and on the voltage V_{cg} applied in read-mode.

The length of the injection pulse is proportional to the injection voltage FGV_{dd} and the biasing current of the comparator OTA that is mirrored into the floating-gate during injection. Obviously, higher injection voltage and biasing current result in higher injection rate, faster injection and shorter injection pulses. However, since the injection voltage FGV_{dd} is supplied on-chip by multiplying the supply voltage V_{dd} by a factor through a charge-pump circuit, it is limited as to how high the injection voltage can be. Additionally, the injection voltage FGV_{dd} is shown to affect programming repeatability accuracy as discussed in the next section.

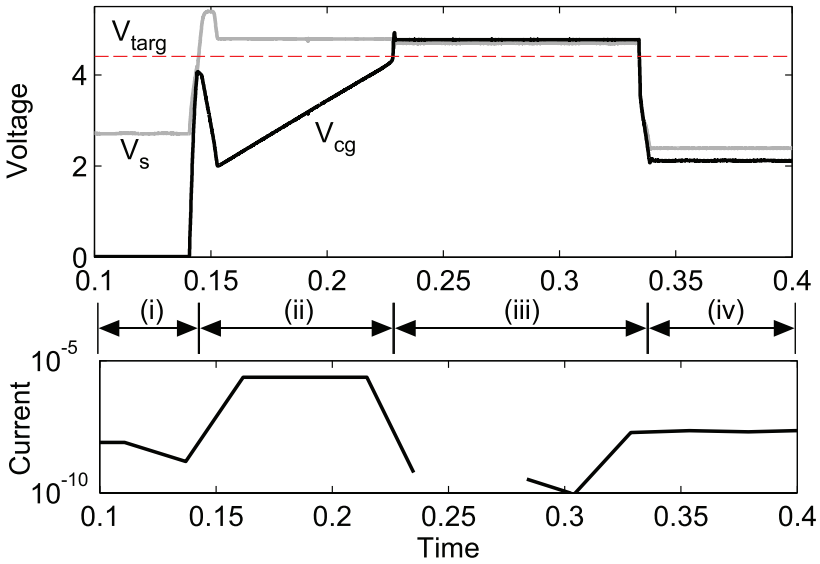

Figure 3.6: Timing diagram of the FG cell and programmer circuit. While $V_{targ} > V_{cg}$, injection takes place and V_{cg} rises linearly.

3.4 Programming Accuracy Measurements

The presented circuit was examined to measure injection linearity and repeatability. I administered those tests on an FG cell on the second version of the RAMP (Reconfigurable Mixed Signal Platform) chip. The RAMP chip was fabricated on 0.35μm CMOS process. Measurements were taken to characterize the programming accuracy. Fig. 3.7 shows the linear results of the programming process. The figure is for a target voltage V_{targ} range of 1.3V ranging from 0.8V- 2.1V. The correspondent voltage V_{cg} is the voltage applied at the control gate in read-mode to get a fixed read current of 10μA. A line of best-fit was depicted through the V_{targ} vs. V_{cg} dataset and the deviation from this line is calculated to compute the programming error. The top plot shows that the memory cell has a linear relationship between V_{targ} and the ramped down V_{cg} (with a slope of 1.0025 and an offset of 157mV). The deviation from a straight line is shown in the bottom plot with a deviation of 0.5mV at most.

The programmer/memory-cell combination is capable of programming to a larger range of voltages, but the relationship begins to deviate slightly from a straight line with a larger V_{targ} range. Figure 3.8 shows the V_{targ} to V_{cg} relationship for a voltage range of 2.2V. However, the V_{cg} values deviate as much as 7mV from the ideal straight line. However, a simple calibration step can be used to correct for these deviations from the straight line. The

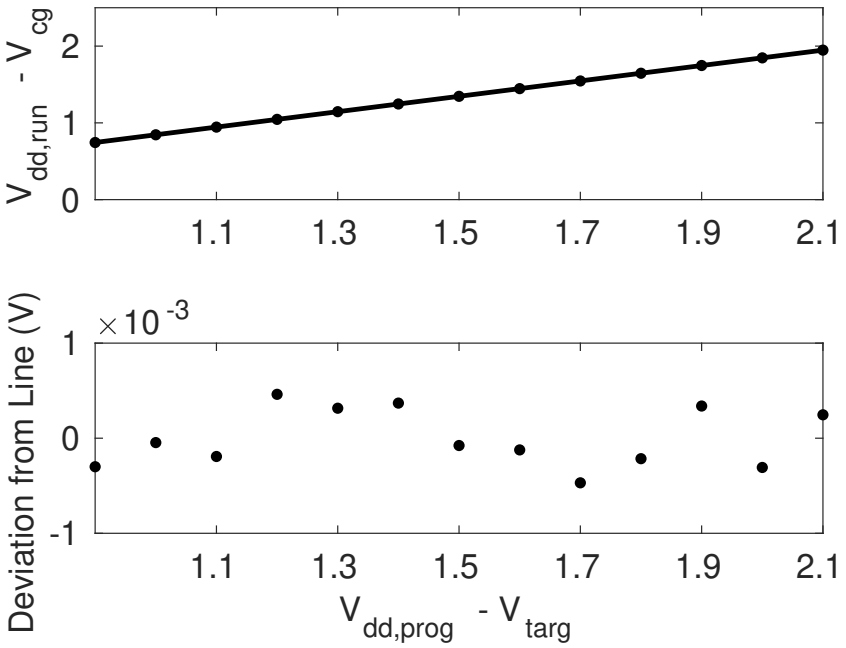

Figure 3.7: Measured Programming Accuracy. Top plot is the linear relation between V_{cg} and V_{targ}. Bottom plot is the deviation of the actual measurements from the projected line.

curvature of the output V_{cg} values has an approximately third-order relationship. Therefore, using third-order polynomial to calibrate the V_{targ} to V_{cg} relationship results in a worst-case deviation of 0.9mV from a straight line, as is shown in Figure 3.8.

In addition, the circuit was characterized in terms of programming repeatbility. Instead of measuring the charge on the floating gate, which is a sensitive node to probe, the measurements have been taken in terms of current and then were converted back to voltage in terms of V_{cg}. This is more relevant considering the application in which the FG cells are used in as a programmable current source. This methodology shows the change in current among n experiments and should show the equivalent change in terms of V_{cg} (i.e. the variance in V_{cg}). It is beneficial especially for current biasing circuits to know how precise V_{cg} should be to achieve an equivalent output current.

The methodology for taking current repeatability measurements and mapping those current measurements to voltage was done as follows:

1. A reference curve is created by injecting the FG cell into an arbitrary target voltage V_{targ} then measure the read-mode current I_d while sweeping the control gate voltage V_{cg}. This results in an arbitrary I_d-V_{cg} curve. The arbitrary V_{targ} should be chosen

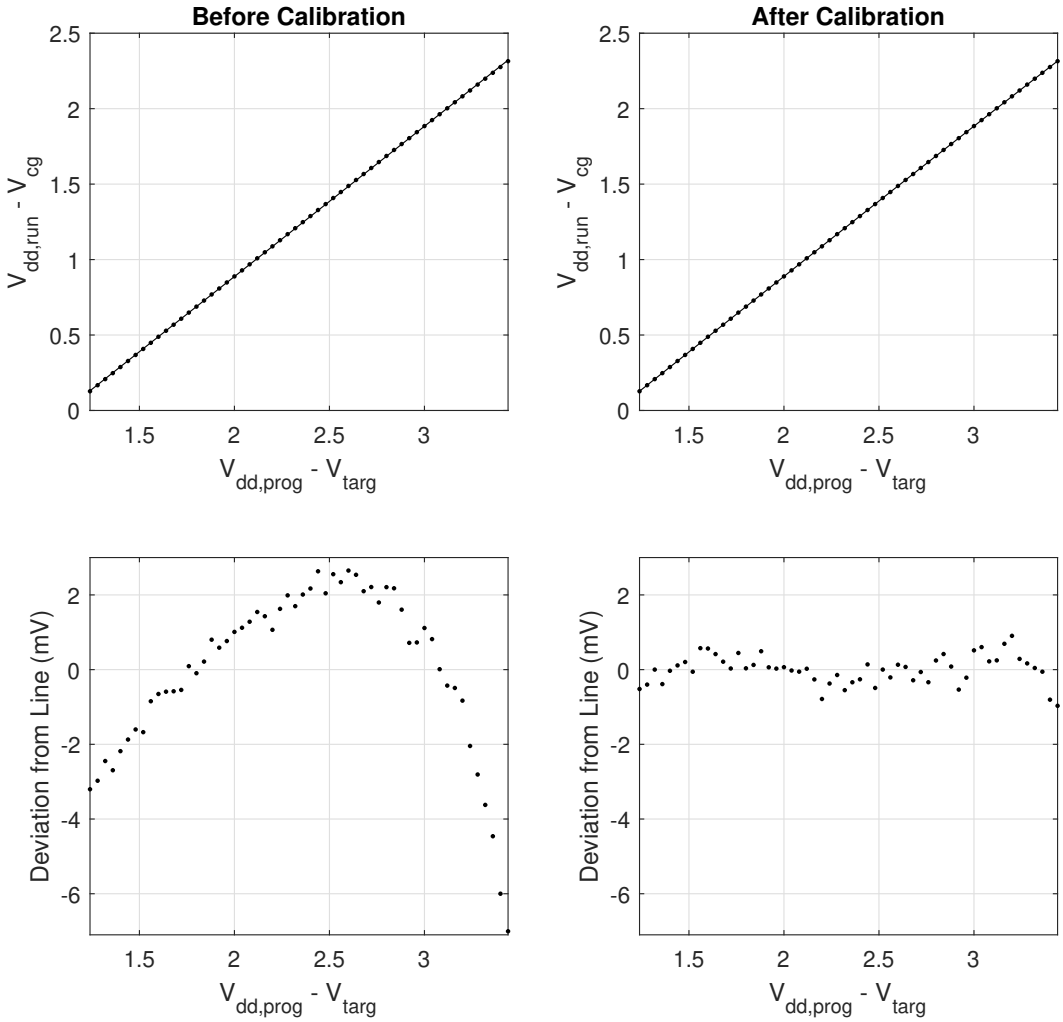

Figure 3.8: Measured programming accuracy before and after calibration.

so that the read-mode current I_d extends to multiple magnitudes of ten to capture all possible current that could be mapped.

2. For a floating-gate transistor in the array, a set of different current ranges are defined such that: $1n$A, $10n$A, $100n$A, 1μA and 10μA. For each of these currents, we found the voltage V_{cg} that gives that corresponding current for a range of different target voltages. For example, if the FG is injected to $V_{targ} = 3.6$V, to get a current of $100n$A a $V_{cg} = 0.9739$V is applied at read-mode.

3. The FG cell is injected to the same range of V_{targ} as step 2. For each V_{targ}, the five

measured values of V_{cg} from step 2 are used to read the current I_d. For instance, for V_{targ}= 3.9V, the following V_{cg}: ([1.054V, 1.5816V, 1.7991V, 1.9286V, 2.0449V) are applied to get a read current of(10μA, 1μA, 100nA, 10nA and 1nA) respectively.

4. Once the five current ranges measurements are taken, the FG cell is tunneled.

5. For the same target voltage, steps 3 and 4 are repeated for n iterations. In the case of this work, the measurements were repeated 50 and 100 iterations.

6. Once measurements are done, all of the n measurements for all of the five current ranges for all of the target voltage range are then mapped to their corresponding voltage V_{cg} using the reference I-V created in step 1.

7. Different statistics for the measured current and the mapped V_{cg} are calculated. This includes average, minimum, maximum, and the standard deviation of V_{cg} from the reference set.

Fig. 3.9 shows one of the data sets measured using the following parameters: biasing current for the programmer comparator OTA = 18nA, injection voltage FGV_{dd}= 5.5V, and the target voltage V_{targ}=2.8V-5V. For every data point, the memory cell was programmed 50 times in order to verify repeatability; from these data are derived V_{cg} standard deviation and the error bars which show maximum and minimum values. The figure shows that the maximum V_{cg} standard deviation is 0.28mv over the range of 2.2V.

For a different programming setting and different target current, a wider range of V_{targ} was injected. The FG memory cell was programmed 100 times to each target value ranging from 1.24V$-$3.56V (2.32V total range), with a full erasure after each write/measurement. Each programming cycle was 100ms in duration and used $V_{dd,fg}$ = 6V. Figure 3.9 shows the standard deviation of the 100 measurements of V_{cg} for each V_{targ}, which had a worst-case value of 280μV. This corresponds to a resolution equal to 13.0 bits for a 2.32V range according to:

$$ENOB = \log_2\left(\frac{FSR}{error}\right) = \log_2\left(\frac{2.32V}{0.28mV}\right) = 13.0 - bits \tag{3.2}$$

Where:

ENOB	the effective number of bits;
FSR	the full scale range.

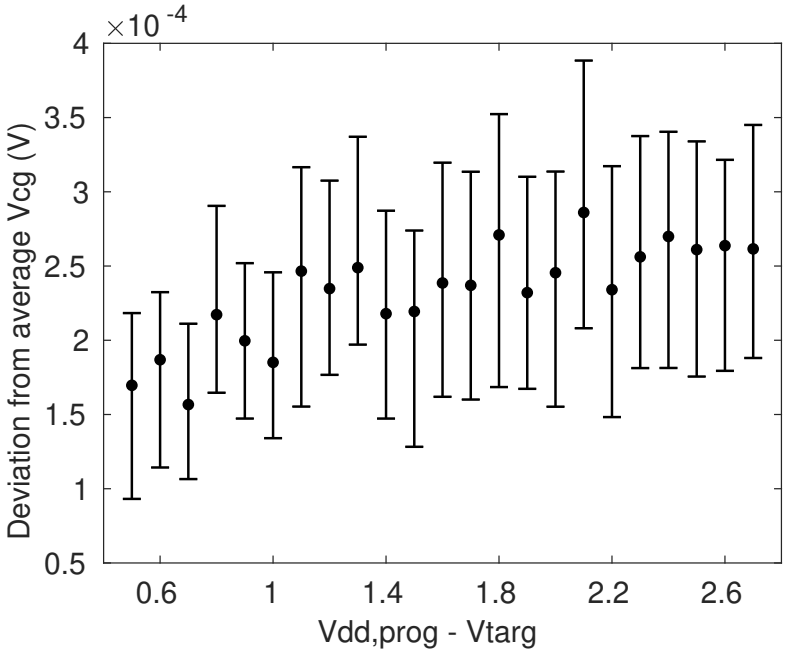

Figure 3.9: Measured V_{cg} standard deviation (100 iteration).

According to equation 2.11, injection current I_{inj} is proportional to the V_{sd} and I_d during injection. Consequently, injection speed can be adjusted by altering V_{sd} or I_d. V_{sd} can be adjusted by using different injection voltages FGV_{dd}, while the current I_d can be adjusted by using different biasing current for the programmer comparator OTA. Depending on the analog application, properly adjusting I_1 and V_{sd} during injection could conclude injection in as fast as few milliseconds.

To characterize the circuit further, different measurements have been taken with various parameter values. Increasing injection speed by increasing the current I_1 does not affect the programming accuracy. Figure 3.11 shows injection time and repeatability for different I_1 values. The top pane shows how using different current during injection has no correlation with programming repeatability. The bottom pane shows the higher the current I_1 the faster the injection.

Fig. 3.12 displays repeatability measurements for similar parameter values as Fig. 3.9. The test parameters are set as follows: programmer OTA biasing current= $18nA$ while FGV_{dd}= 5.5V and 6V.

The results of this test show that injection voltage FGV_{dd} can affect circuit accuracy. The pattern that is observed from this test is that the lower the FGV_{dd} the better the accuracy.

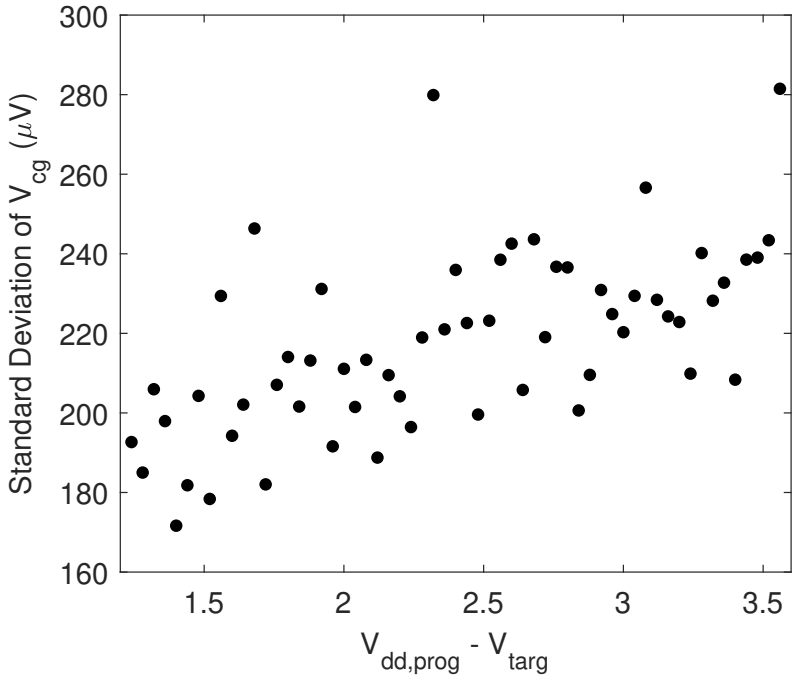

Figure 3.10: Measured programming precision.

This could be explained from equation 3.1, as the injection current depends exponentially on injection voltage, and the higher the injection current the bigger the charge rate on the floating-gate node. Overall, lower FGV_{dd} voltages produce more accurate, repeatable target currents.

Another general perceived pattern is that the higher the target voltage V_{targ}, the better the accuracy. This is due to the programmer OTA being an nFET input pair OTA. Therefore, depending on the OTA biasing condition, smaller target voltages might pull one or more of the input FETs out of the saturation region. Moreover, some lower target voltages did not meet the threshold voltage, causing the floating-gate transistor to not be injected or inaccurately under-injected.

Finally, since the main application of the FG memory cell introduced here is to be used in low-power analog systems, power dissipation of the cell is calculated. In voltage reference mode, the FG cell was biased with $I_1 = 20nA$ and $I_2 = 2nA$, yielding a low power consumption of 66nW/cell.

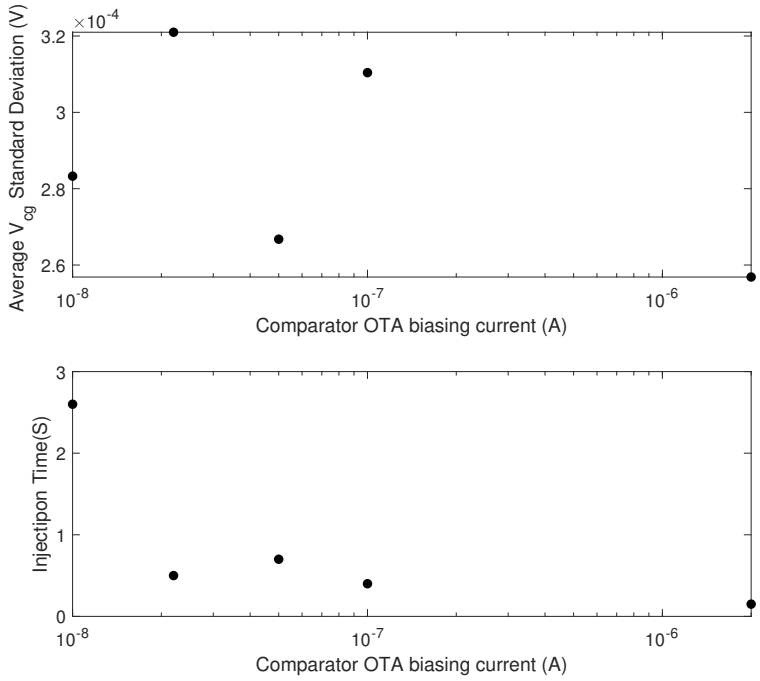

Figure 3.11: Measured Programming Accuracy

3.5 FG Memory Array

In previous context, it was demonstrated that for analog applications, the FG transistor is rarely used as a single element on a chip. Most analog applications, such as FPAA, use a large number (hundreds to thousands) of floating-gate transistors. In this section, I show that the presented configuration, the FG cell and programmer circuit, can be utilized to build a FG memory array where only selected cells are to be programmed. I also will introduce a system that can be used to parallel program more than one FG transistor at any time.

Fig. 3.13 displays two-by-two FG arrays of the memory cell discussed in section 3.2. Bigger arrays could be constructed with the FG cell and the programmer circuit; however, only two-by-two is shown here to easily explain how the array works with the programmer. Using this configuration, only one programmer OTA is needed to inject one FG cell at a time.

In program-mode, to select a FG, switches in that specific cell are configured as follows:

1. S_{w1} is opened while switch S_{w2} is closed to connect V_{cg} to the programmer OTA. S_{w5} is set at position 1 to connect the current source M_{I1} to the circuit.

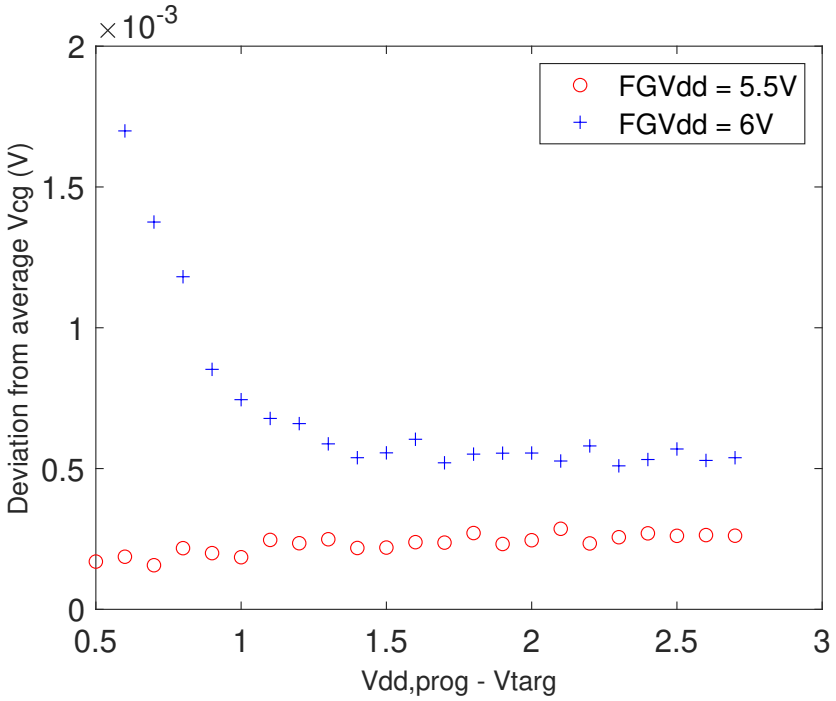

Figure 3.12: Measured V_{cg} standard deviation (100 iterations) with two different FGV_{dd} voltages.

2. S_{w3} is set at position 2 to connect the current mirror M_{I2}-M_{I3} to the source of the FG, while S_{w4} is set at position 4 to connect the FG drain to ground.

However, unselected FG cells in programming-mode are set to place those cells in an idle state and their switches are set as follows:

1. S_{w1} and S_{w2} are opened to disconnect V_{cg} form any voltage, while switch S_{w5} is set at position 2 to turn off the current source M_{I1}.

2. S_{w3} is set at position 1 to force M_2 to cuttoff and the FG source to be low, while S_{w4} is set at position 4 to connect the FG drain to ground. As a result, the unselected FG will not experience high V_{sd} to start injection (i.e. $V_{sd} = 0V$)

In run-mode, the FG is configured to either bias a circuit "on-chip" or supply current to external circuit "off-chip." In this configuration, V_{cg} is pinned out and connected to external voltage to supply the voltage required to bias the FG transistor. FGs switches are set as below:

1. S_{w1} and S_{w2} are closed to connect V_{cg} to external voltage necessary for biasing the FG transistor, while switch S_{w5} is set at position 2 to turn off the current source M_{I1}.

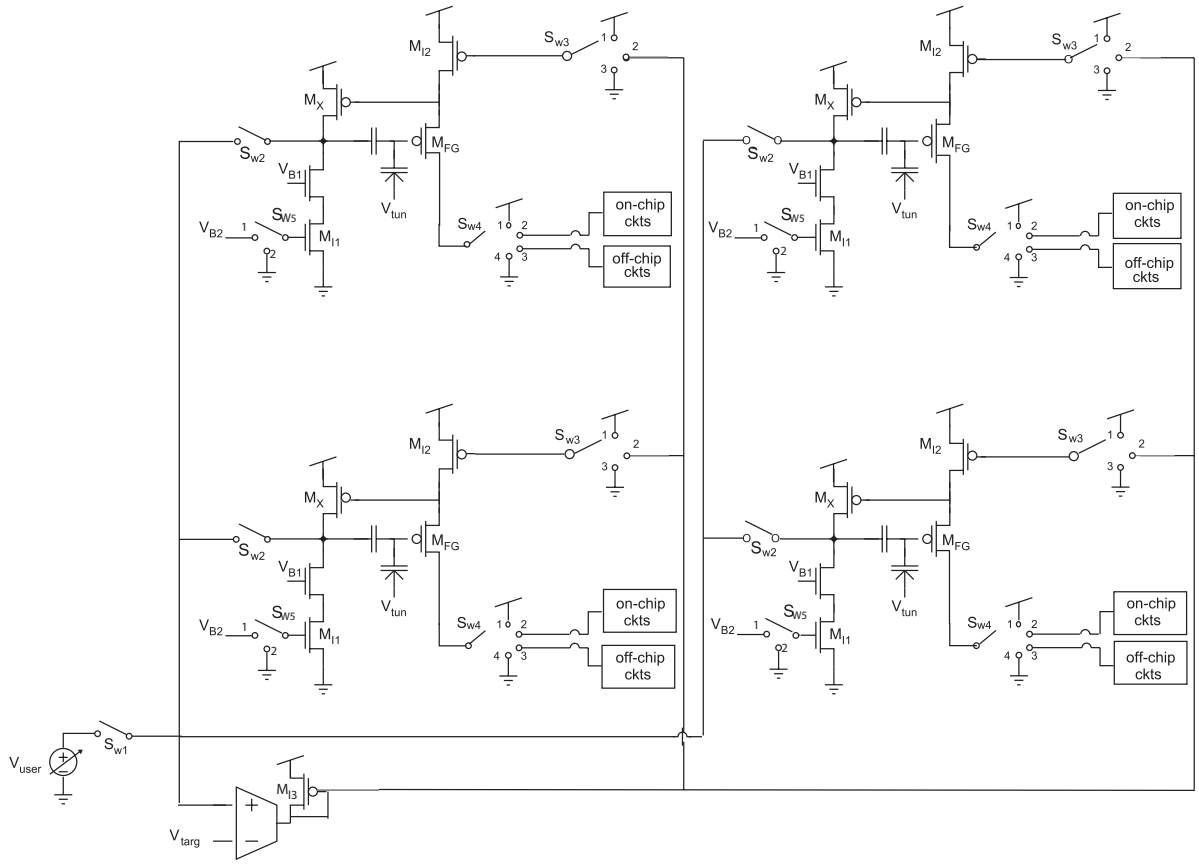

Figure 3.13: Two-by-two array architecture for the memory cell and programmer for above-ground injection.

2. S_{w3} is set at position 3 to turn on M_2 and connect FG source V_s supply voltage, while S_{w4} is set at either position 1 if the FG is not selected, position 2 if the FG is to be used with "on-chip" circuit, position 3 if the FG to be connected to "off-chip" circuit or to measure the read-mode current for testing purposes.

Switches S_{w1}, S_{w2}, S_{w3}, S_{w4} and S_{w5} are controlled by digital signals that controls the operation mode of the circuit and whether the cell is connected to the programmer or not.

3.6 System Application

FG transistors have a wide range of analog applications that require a large number of FGs to be integrated on a single die. These applications range from simple filter banks to more

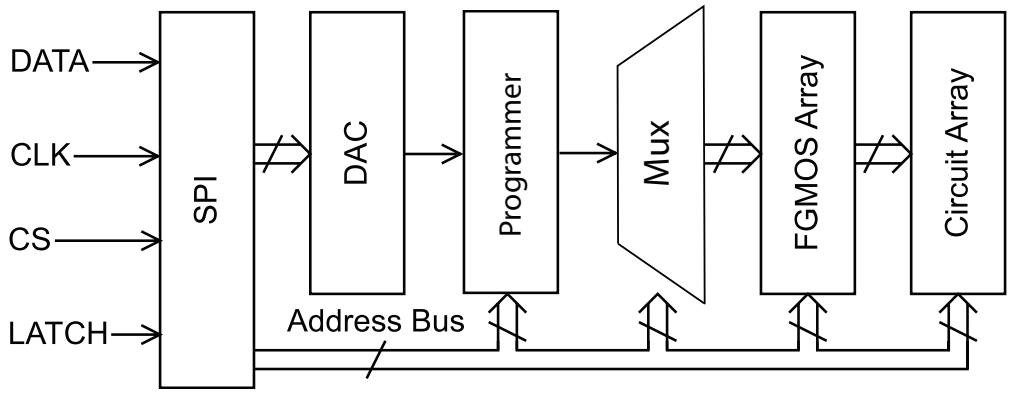

Figure 3.14: Signal flow diagram of serial programming architecture used on the programmable analog system [2]

complicated field-programmable analog arrays (FPAAs). The above-ground programmer circuit was used to inject FGs on the RAMP chip [2]. The programming method used on the RAMP chip is serial programming where one FG is injected at a time. When $V_{dd,fg}$ is elevated, injection starts, and only the selected FG is injected. All unselected FG cells are configured to prevent injection. This is established by pulling the unselected cells' V_{cg} to $V_{dd,fg}$ and by setting the drain-to-source voltage, V_{sd}, to be low ($\sim 0V$) by connecting the source and drain to ground. Once the selected FG cell is injected to the desired target, a new FG cell is selected and connected to the programmer, and the process is repeated for each of the FG cells needing to be injected. Since only one programmer circuit is used in serial programming, $N \times M$ programming cycles are required to program an $N \times M$ array. A signal-flow block diagram of the serial programming method is shown in Figure 3.14.

To speed up the injection process of large FG array, extra circuitry could be added to the FG array and programmer configuration to enable parallel programming where more than one FG cell can be programmed at any time. While one programming cycle is needed, a number of programmer circuits per chip are required to perform parallel programming. In general, to program N FG cells in parallel, N programmer circuits are required. Here, I present a circuit that utilized the FG array-programmer presented earlier to parallel program $N \times M$ array of FGs using only four digital inputs and only M (i.e. number of columns) programmer circuits.

This parallel continuous-time programmer was originally introduced in [50]. A block diagram of the circuit is shown in 3.15. In this methodology, from left to right, digital inputs are loaded into SPI circuit using the four digital pins (Data, CLk, CS, and LATCH). Then the

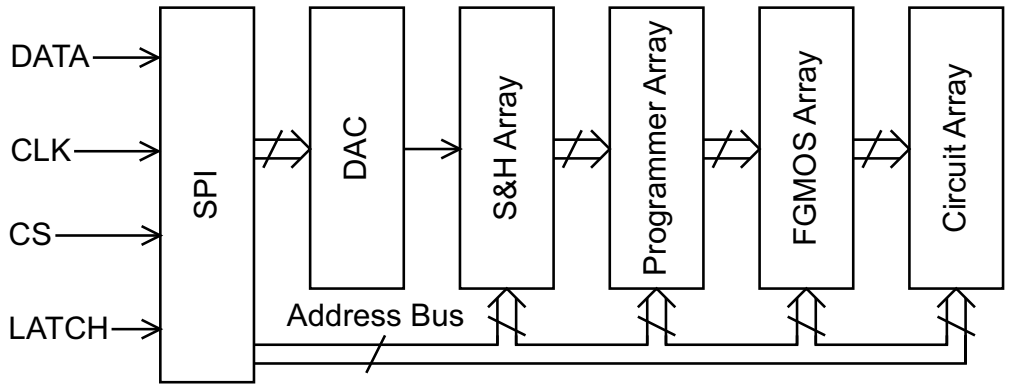

Figure 3.15: Signal flow diagram of the parallel continuous-time programming architecture.

digital-to-analog converter (DAC) is used to generate analog target voltages. A sample-and-hold (S/H) array is used to sample and apply the analog target voltages to the programmer which injects the selected FG and perform programming. Each row in the FG array is programmed in parallel, which means one row needs to be selected for each programming cycle. For each column, a programmer OTA is required, and for each programming cycle this OTA is connected to the cell in the corresponding column in the selected row. A DONE circuit is used to monitor V_{cg} of each floating-gate cell, once the array have finished programming the DONE circuit generate high which lowers the supply voltage, disconnects the FGs from their programmers and connects FG cells to their application circuits. A high START pulse is used to provoke the injection process and start programming.

Figure 3.16 demonstrates the programming time for serial and parallel programming. As the figure illustrates, to program an array of FGs serially, one FG is programmed at a time which makes programming process linearly proportional to the overall size of the array. Using this method for programming results in overall programming time of:

$$TotalTime_{SerialProgramming} = N \times M \times (t_s + t_i) \tag{3.3}$$

where N is the number of rows, M is the number of columns, t_i is the injection time, and t_s is pre-injection time (time to select the FG cell, connect the programmer and apply the start pulse) which is very short compared to t_i. On the other hand, using the parallel programming method presented here dramatically reduces the overall programming time by staggering FG programming through time which, as shown in Fig. 3.16. To program $N \times M$ array using

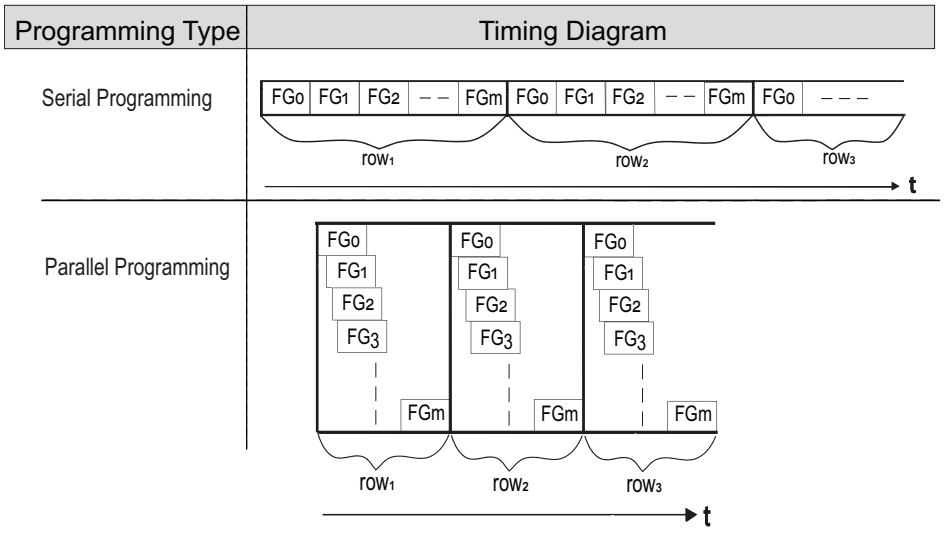

Figure 3.16: Serial vs parallel programming

our parallel programming method, the overall programming time is:

$$TotalTime_{ParallelProgramming} = N \times (M \times t_s + t_i) \tag{3.4}$$

Generally, comparing the two methods, our parallel programming neutralizes between minimizing die area (by using a programmer circuit per each column) and programming time, which make it the most appropriate programming method for FG-dense analog applications.

As a proof-of-concept, the aforementioned FG memory array and the continuous-time parallel programmer circuit are used to tune the corner frequencies of bandpass filter bank. The FG memory array, programmer circuit and filter bank were fabricated in a $0.5\mu m$ standard CMOS process available through MOSIS. The chip contains 8 sample-and-holds, 8 programmer OTAs, 16 floating-gate transistors, and 8 bandpass filters as well as the SPI, DAC, and miscellaneous peripheral circuitry such as level-shifters, decoder, multiplexers and switches. For each bandpass filter two floating-gate transistors are used to set the two corner frequencies of the filter. Therefor 16 floating-gate transistors are needed, two for each of the 8 bandpass filters. Those FGs are arranged in two rows and eight columns. One row (8 FGs) is to be programmed in parallel, resulting in two programming cycles to program the whole FG array. A die photograph of the chip is shown in Fig. 3.17.

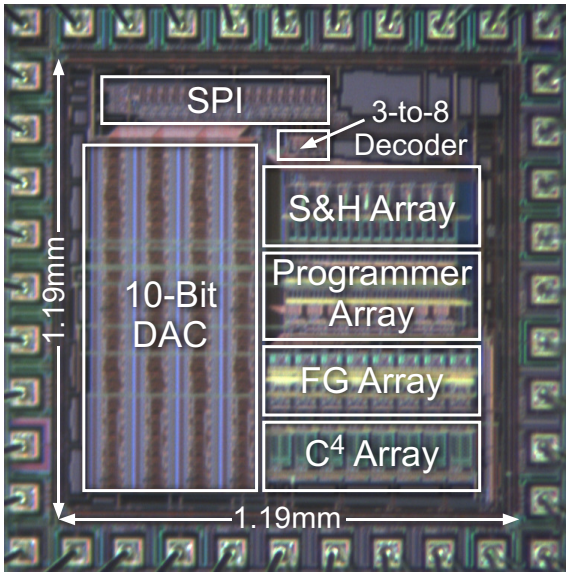

Figure 3.17: Die photograph of the programmable bandpass array chip.

3.6.1 The C^4 Bandpass Filter

n this work, capacitively-coupled current conveyor (C^4) shown in Fig. 3.18 (a) is used to build the filter bank. This bandpass filter was originally presented in [3]. It is an OTA-based capacitively coupled current conveyor (C4) whose corner frequencies, a Gain (A_v) and quality factor (Q) are proportional to the trnasconductance of the OTA. The transfer function of C^4 is given by

$$\frac{V_{out}}{V_{in}} = -\frac{C_1}{C_2}\frac{s\tau_l(1-s\tau_f)}{1+s\left(\tau_l+\tau_f\left(\frac{C_O}{C_2}-1\right)\right)+s^2\tau_h\tau_l} \tag{3.5}$$

where:

$$C_T = C_1 + C_2 + C_W \qquad C_O = C_2 + C_L \tag{3.6}$$

and τ_l and τ_h are the time constants of the low corner and high corner frequencies of the bandpass filter. And the zero τ_f is designed to be at sufficiently high frequency that its effect can be ignored.

$$\tau_l = \frac{C_2}{G_{m,L}} \quad \tau_h = \frac{C_O C_T - C_2^2}{C_2 G_{m,H}} \quad \tau_f = \frac{C_2}{G_{m,H}} \tag{3.7}$$

The two OTA transconductances $C_{m,L}$ and $G_{m,H}$ from 3.18 (a) controls the two corner frequency of the bandpass filter. As transconductance is proportional to biasing current 3.8, floating-gate transistors can be used as current sources to bias those two OTAS, control their

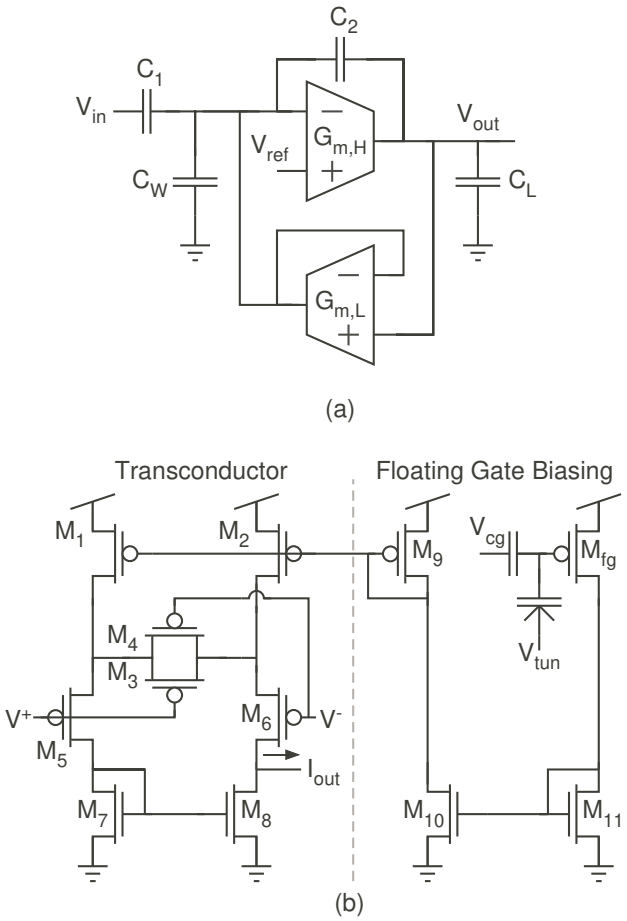

Figure 3.18: Overview of the OTA-based $\mathbf{C^4}$ bandpass filter. (a) Schematic of OTA-based $\mathbf{C^4}$. (b) Schematic of bump-linearized OTA used in $\mathbf{C^4}$.

trnasconductance and tuning the filter's both corner frequencies, Fig. 3.18 (b).

$$G_m = \frac{\kappa I}{V_T} \tag{3.8}$$

The data shown in 3.19 was presented in [3] and was measured on a 0.35μm CMOS process.

3.6.2 C^4 Programming

Each floating-gate transistor is programmed to a specific target to set the high/ low frequency of a bandpass filter in the array. Fig. 3.20 shows the results performing frequency decomposition for various bandwidth and filter spacing by tuning C^4 bandpass filter bank. Quality factor (Q) for each configuration was selected so that the filters cross at their -3dB points. Fig. 3.20 (a) shows the results of octave spacing with Q $\sim$ 1.4 and a center frequency

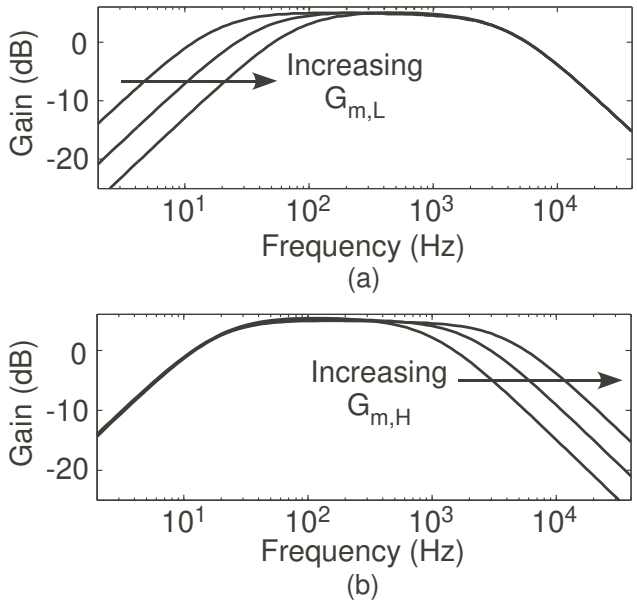

Figure 3.19: Frequency response were measured on 0.35μm CMOS process presented in [3](a) Frequency response when $G_{m,L}$ is increased. (b) Frequency response when $G_{m,H}$ is increased.

starting at $88Hz$, (b) shows the results of half-octave spacing with Q $\sim$ 2.9 and a center frequency starting at $300Hz$, (c) shows the results of third-octave spacing with Q $\sim$ 4.3 and a center frequency starting at $445Hz$.

3.7 Conclusion

Continuous programming is one of the methods used to inject floating-gate transistors. A negative feedback is needed to linearize the injection process. However, supplementary programmer circuit is needed to stop injection process once the target is reached. In this chapter, a floating-gate transistor memory cell that utilizes hot-electron injection and Fowler-Nordheim tunneling is presented. Furthermore, a continuous-time programmer circuit that employs hot-electron injection is introduced. The programmer circuit controls the injection process and stops injection once the target voltage is accomplished. The floating-gate memory cell is used to build a memory array that can be used in analog applications.

The programmer circuit was characterized and tested for repeatable programming. It was shown that the FG cell with the programmer circuit endure linear injection with a slope of 1.0025 and an offset of 157mV. Additionally, it was shown that the programmer circuit could produce a programming accuracy of 11.94 bits. However, few patterns were

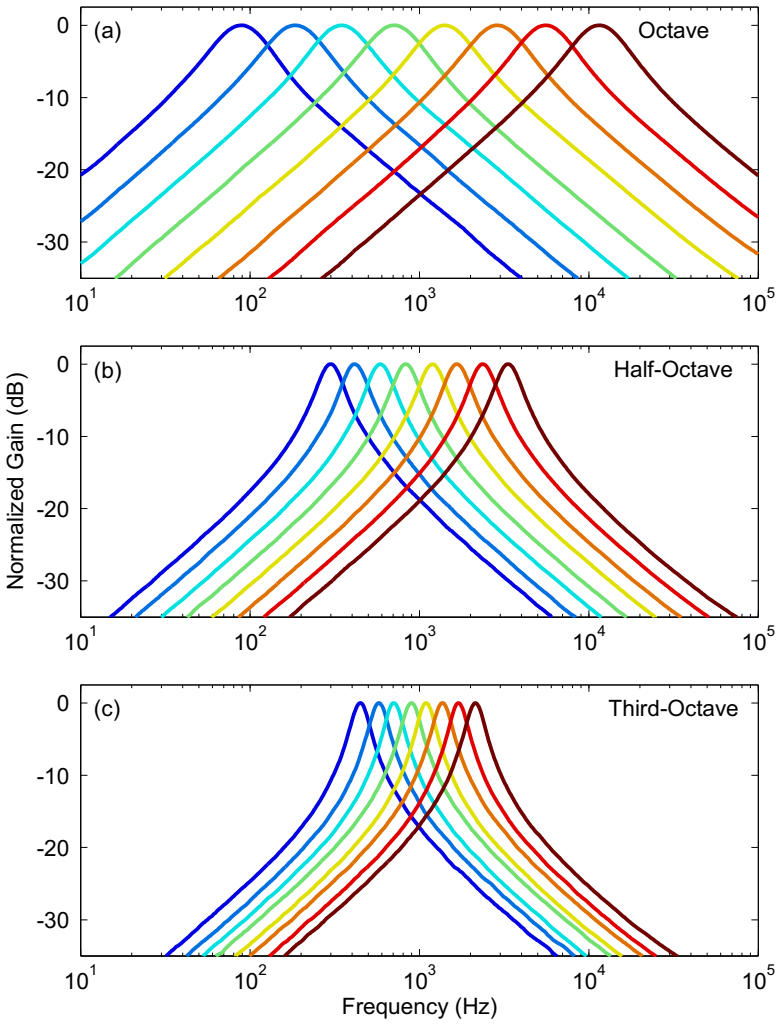

Figure 3.20: Programmed $\mathbf{C^4}$ array frequency responses for above-ground programming. (a) octave spacing starting at $\mathbf{f_c = 88Hz}$, (b) half-octave spacing starting at $\mathbf{f_c = 300Hz}$, and (c) third-octave spacing starting at $\mathbf{f_c = 445Hz}$.

observed when repeatability was tested, one of the major patterns is that injection voltage FGV_{dd} affects injection accuracy, where lowering the injection voltage gives better results was observed. Furthermore, it was shown that the programmer circuit with the FG array could be configured to perform parallel programming. The configuration discussed here achieved parallel programming with a minimal number of programmer circuit and only four pins of the chip. Moreover,the programmer circuit with the memory array were used as programmable current source to tune the corner frequencies of the analog application of C^4 bandpass filter bank. Three different filter spacing were produced to prove the concept of accurately programming FG array in FG-dense applications.

Chapter 4

Floating-Gate Below Ground Injection

Injecting and tunneling floating-gate transistor requires a complex programming procedure and a significant infrastructure overhead to generate a voltage drop greater than V_{dd} that is required for programming and tunnelling. In this chapter, two novel programmer circuits to inject FG array are presented. Instead of holding drain at ground and elevating V_s to a voltage above V_{dd}, the presented circuit creates high V_{sd} by lowering the drain to negative voltages. I will refer to this technique as below-ground injection. The presented programmer circuit used "below-ground voltages" that allows injection of individual floating-gate cell in large arrays without using isolation switches to operate below the substrate voltage. This makes the technique possible in any standard single-well CMOS process.

Using negative voltage as an alternative to create high injection voltage V_{sd} to inject FG array had not been tested before. The issue with using below-ground injection has always been how to implement it in a single-well CMOS process without forward biasing the diffusion area for the switching circuitry. This issue is resolved in this work via using "indirect programming" of floating gate transistor [51]. As injection voltage in below-ground is controlled through V_d, a less sensitive node than V_s, consequently, below-ground injection is anticipated to give more accurate results than above-ground, which employs V_s to control injection process.

4.1 Negative Voltage Injection

Floating-gate hot-electron injection requires a high voltage that exceeds the rated process V_{dd}. Typically this high voltage is accomplished by raising the whole circuit's supply voltage resulting in more complicated configuration. On the other hand, below-ground injection keeps the supply voltage V_{dd} at the process rated voltage while lowering the drain voltage to negative potential to created high V_{sd} enough to proceed with injection process. Additionally, the above-ground programming technique of increasing source voltage V_s is not ideal as it would alter V_{sg} resulting in changing the channel current to unknown value, alternatively, all nodes voltages associated with the FG transistor (including the drain) are ramped up to maintain constant V_{sg} through out injection, which results in significant infrastructure. Furthermore, typically injection V_s is generated by a charge-pump with some ripple on it which affects programming accuracy as well. Finally, above-ground programming technique requires high-side [52] switches to switch between the higher V_{dd} to program and the lower V_{dd} in read mode, which increases the power consumption and die area, therefore, using below-ground injection is advantageous over above-ground injection in that it requires less overhead infrastructure, which yield to more simplified circuit with less power dissipation that is practical for large FG arrays in analog applications.

In below-ground injection, it is the drain voltage V_d that is altered to create the high V_{sd} required to fulfill injection condition, therefore, the channel current is not affected by the change in V_{sd} specially when operate in sub-threshold region. The negative voltage at the drain is generated by a negative charge-pump and any ripple at the output will not be coupled to the channel current due to the larger impedance at the drain node. Additionally, to generate V_{sd} high enough to proceed with injection, a lower voltage magnitude is needed from the negative charge-pump. This is because the supply voltage V_{dd} contributes to the injection V_{sd}as the injection $V_{sd} = V_{dd} + V_{blwgnd}$. For example, instead of applying 6V at the source to achieve V_{sd} = 6V that is enough for injection, below-ground would only require -3.5V at the drain for $0.35\mu m$ process with a supply voltage of 2.5V. Overall, the two main parameters that control the injection process (i.e. V_{sd} and I_d) are precisely controlled using below-ground injection, this is due to the constant V_{sg} through out programming process which results in more predictable performance and programming results, also due to the channel current not being affected by the ripple at the charge-pump output and the change in V_d.

4.2 Negative Charge-Pump

To generate the negative voltage at the drain node, negative charge-pump is used. A block diagram of a negative charge-pump discussed in [4] is shown in Fig. 4.1. This charge pump is based on the work in [1], an improved version with reduced output ripple is presented in [53]. The negative charge-pump employs four Charge Transfer Switch (CTS) stages and a voltage divider to shift the output voltage up so the feedback voltage V_{fb} stays within the operating rage between ground and V_{dd}. This charge-pump utilizes a variable-frequency regulation technique to lower the ripple at the output voltage. The OTA compare the different between the feedback voltage V_{fb} and the target voltage V_{targ} and converts the difference to current that to modulate the frequency of the current-controlled ring oscillator. The edgifier circuit is used here to sharpen the the edges of the oscillation generated by the ring oscillator.

Unlike positive charge-pump, negative charge-pumps use ground as the input signal. The charge-pump output will set at supply voltage V_{dd} when the charge-pump is disabled. The relation between V_{out} and V_{targ} is as following:

$$V_{out} = \left(1 + \frac{R_2}{R_1}\right) V_{targ} - \frac{R_2}{R_1} V_{dd} \tag{4.1}$$

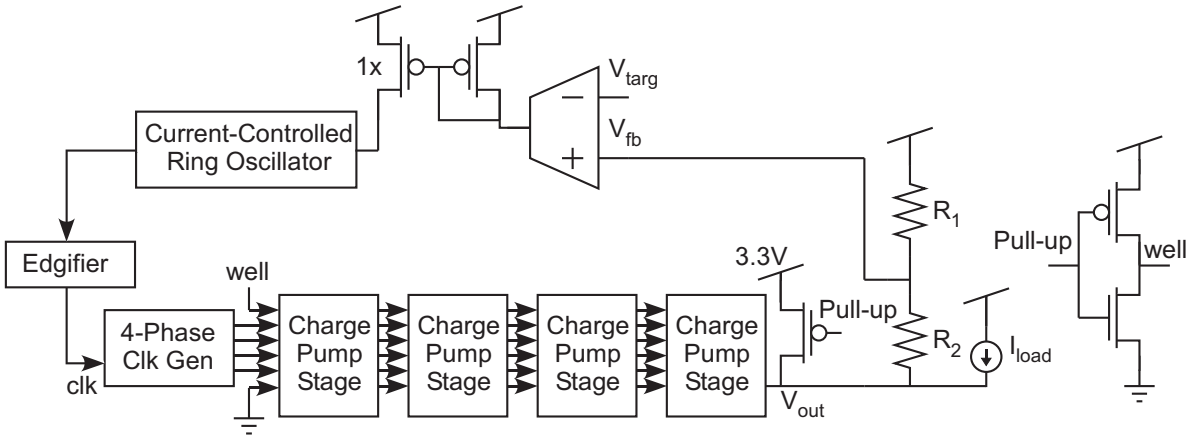

Figure 4.1: Block diagram of negative charge pump from [4]

4.3 Current Conveyor Programmer Circuit for FG Array

As discussed in chapter 3, continuous-time programming utilizes negative feedback to linearize injection process and to stop injection once the target is reached. The circuit I present here, employs similar negative feedback configuration. However, it utilizes different injection technique that adopts the approach of lowering the drain voltage to below-ground (i.e. negative voltages). Consequently, the original FG cell and programmer circuit used in 3.2 and 3.3 were adjusted to utilize below-ground injection. This circuit is developed based on the work presented in [5]

4.3.1 FG Memory Cell

To make below-ground injection possible without the use of switching circuitry at the drain node which is not applicable in single-well CMOS process, indirect programming of FG transistor is used, The cell I used for this circuit was originally presented in [5]. This is based on the work presented in [51]. Indirect programming cell consists of two pFET transistor M_{inj} and $M_{circuit}$ as shown in Fig 4.2. While these two transistors share the same floating-gate, their source and drain terminals are connected to different nodes in the circuit. Transistor M_{inj} is used to inject the floating-gate node, while transistor $M_{circuit}$ is used in read mode to provide the biasing current/ voltage for the application circuit. In program mode Fig. 4.2, transistor M_{inj} is connected to the programmer circuit to perform injection. The source V_s of M_{inj} is connected the negative feedback amplifier M_1 to provide linear injection through programming process, while the drain $V_{negative}$ is connected to the output of the negative charge-pump to provide negative voltage. However, the source and drain of $M_{circuit}$ are connected to ground so this transistor will be shut-off and will not participate in the injection process.

Alternatively, in read mode Fig. 4.3, a read-mode voltage V_{cg} is applied at the control gate to bias $M_{circuit}$ to provide the biasing current required by the analog application. Transistor M_{inj} is shut-off by connecting its source ans drain to supply voltage V_{dd}. On the other hand, transistor $M_{circuit}$ is connected to the application circuit by connecting its source to supply voltage V_{dd} and its drain to the application circuit, the current flowing through $M_{circuit}$ in read-mode is set by the charge injected to the floating-gate node through M_{inj} in program-

mode. Overall, to switch between program and read mode, the corresponding transistor is activated by applying the appropriate V_{sd} voltage depending on the operation mode.

Using indirect programming eliminates the need for selection switch at the drain node of the floating-gate transistor, which is not possible in a single-well process. The traditional method to implement selection circuitry is to use transmission gate, this would cause an issue if used at the drain for below-ground injection. Negative voltage would cause the n-type diffusion regions of the transmission gate to be forward bias with the p-type substrate (i.e. pn-junction). Instead, for below-ground programming, switching is employed at the source node to change the potential at that node depending on the operation mode. This technique facilitates the selection of individual FG cell in FG array with no selection circuitry operating below ground.

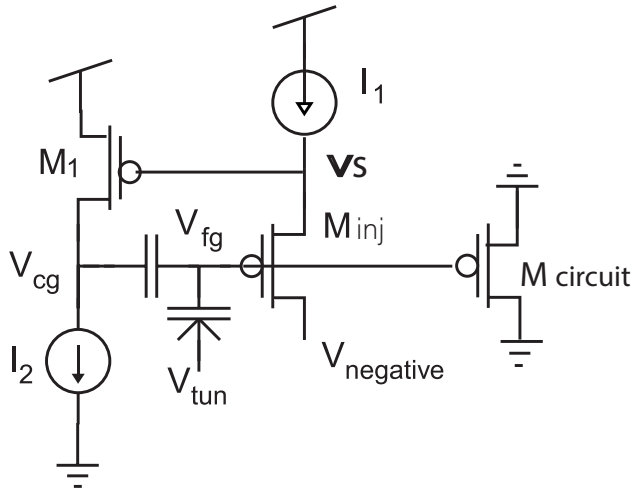

Figure 4.2: FG cell in below-ground programming in program-mode. M_{inj} is placed under high V_{sd} while $M_{circuit}$ is put in dormant state as its source and drain are at ground.

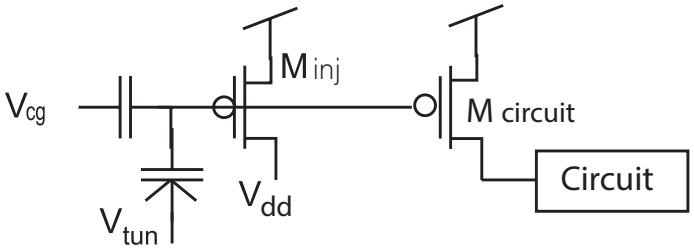

Figure 4.3: FG cell in below-ground programming in read-mode. M_{inj} is in idle state with its source and drain at V_{dd} while $M_{circuit}$ is used as programmable current source and provides biasing current to application circuit through its drain.

4.3.2 FG Array and Programmer Circuit

To stop injection process once target is reached, a comparator OTA is used. Similar to above-ground programmer 3.3, this OTA is used through injection process to compare V_{cg} a target voltage V_{targ}. Fig. 4.4 shows the programmer circuit connected to the FG cell during injection. Through injection V_{cg} keeps increasing as more charge placed on the floating-gate. The programmer OTA converts the difference between V_{cg} and V_{targ} to current I_1 that is mirrored through the current mirror M_3 - M_2 to FG transistor M_{inj}. Once V_{cg} reaches V_{targ} the OTA current is shut off causing injection to stop.

When programming process starts, the multiplexer connected to transistor M_{inj} is set so the source of the transistor is connected to V_s, the feedback path between the source and control gate of M_{inj} transistor. In addition, the multiplexer connected to the source of $M_{circuit}$ transistor is connected to ground. On the other hand, the drain of M_{inj} transistor is connected to the negative voltage $V_{negative}$, while the drain of $M_{circuit}$ is connected to ground. This will place the channel of transistor M_{inj} under a high V_{sd} while the channel of transistor $M_{circuit}$ will have small V_{sd}. This will cause injection current to flow only through M_{inj} and no current through $M_{circuit}$. As electrons are injected to the floating-gate due to injection current, the control-gate voltage V_{cg} will linearly increase. Because V_{cg} is connected to the OTA, the OTA will continue supply a constant current to the current mirror (M_3 and M_2) and the injection process will continue until V_{cg} gets close to V_{targ}, after which the OTA will start to decrease the current mirrored to M_{inj} through M_3 and M_2. This reduction in current would cause smaller injection rate which consequently results in better accuracy. When biasing current I_1 decreases, transistor M_1 goes into deep ohmic region which cause V_{cg} to jump to supply voltage and stops the injection process.

Transistor M_{Reset} is used in the circuit to provide a short reset pulse. This pulse resets the feedback loop and makes V_{cg} to go to a stable voltage lower than V_{targ}. While the use of M_{bias} is optional, it is used to limit the current during the reset pulse.

In read mode, $M_{circuit}$ is used to supply the required current. Using off-chip control signal the cell is disconnected from the programmer circuit, and an external constant voltageV_{cg} is applied to the control gate. In addition, the source of $M_{circuit}$ is connected to the supply voltage V_{dd} using *select* control signal in Fig4.4 and the drain is connected to the application circuit, consequently a current $I_{circuit}$ flows through its channel into the application circuit. On the other hand, the source and the drain of M_{inj} is connected to V_{dd} causing no current

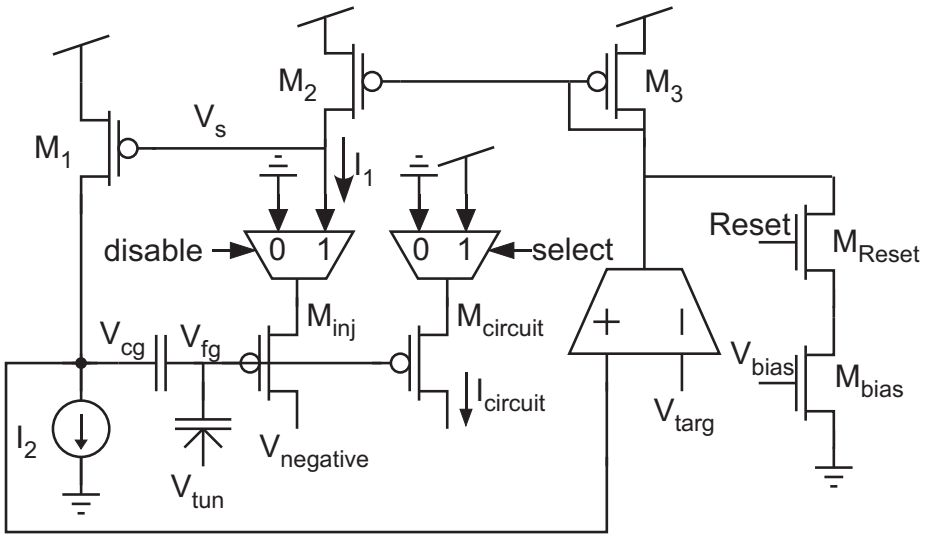

Figure 4.4: Complete below-ground programming structure, it consists of indirect FG cell and the programmer OTA.

through it. Timing Diagram of the programming process from [5] with the reset pulse is shown in Fig 4.5.

The main purpose of developing an effective technique to inject FG transistor is to be able to implement it in array configuration to use it in analog applications. Since using indirect programming makes below-ground injection possible without exposing selection circuitry to negative voltages, it was practical to use the below-ground memory cell from 4.2 to build an FG memory array. A single negative charge-pump is shared among all FGs in the array to supply the negative voltage. Drains of all M_{inj}s in the array are connected to the output of the charge-pump, therefore drains of all M_{inj}s will experience the negative voltage during injection of any cell at any time. However, to prevent unselected cells from being injected and isolate the only selected cell in the array, only the source of the selected M_{inj} is connected to the programmer circuit and the negative feedback amplifier while the sources of the unselected M_{inj} are set at ground, therefore they experience a relatively small V_{sd} that is not enough to excite injection.

Fig. 4.6 shows two-by-two FG array configuration I used in this work. Bigger array of any size could be developed using this configuration. In program-mode, to selected a FG, switches in that selected cell are set as following:

1. S_{w1} is opened while switch S_{w2} is closed to connect V_{cg} to the programmer OTA.

2. S_{w3} is set at position 2 to connect the current mirror M_{I2}-M_{I3} to the source of the

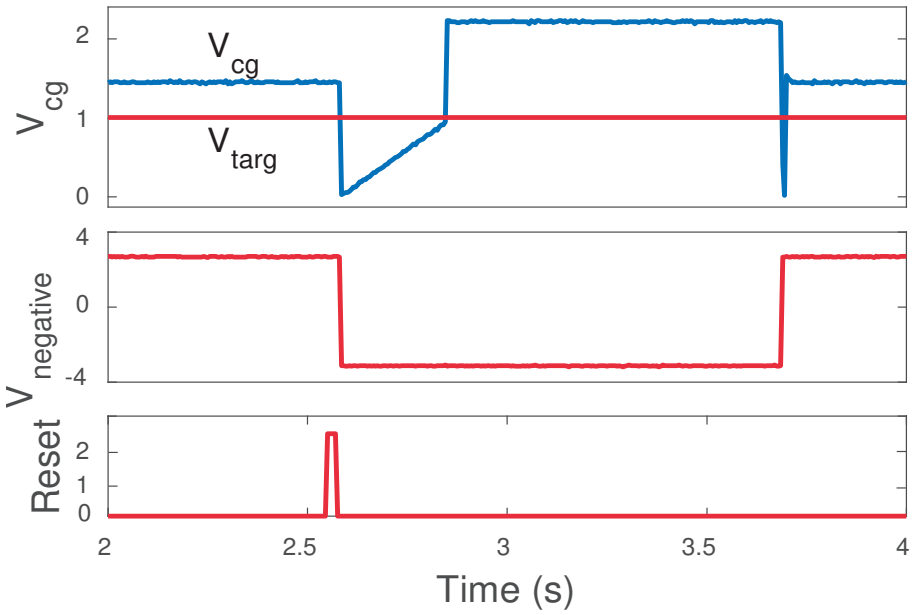

Figure 4.5: Timing diagram of injecting FG transistor [5]

FG, while S_{w4} is set at position 2 to connect the FG source to the negative feedback common-source transistor M_x.

3. Finally, switches S_{w5} and S_{w6} at the source and drain of $M_{circuit}$ of the selected cell are set at positions 2 and 4 respectively, to place $M_{circuit}$ under small $V_{sd} \approx 0$.

However, unselected FGs cell in programming-mode are set to place those cells in idle state and their switches are set following:

1. S_{w1} and S_{w2} are opened to disconnect V_{cg} form any voltage.

2. S_{w3} is set at position 1 to force M_2 to cuttoff and the FG source to be low, while S_{w4} is set at position 3 to connect the FG source to ground, resulting in small V_{sd} not enough for injection.

3. For all $M_{circuit}$ in unselected cells, S_{w5} is at position 2, while S_{w6} is at position 4 to neutralize trnasistor $M_{circuit}$.

Similar to above-ground circuit, in run-mode, the FG is configured to either bias a circuit "on-chip" or supply current to external circuit "off-chip". In this configuration, V_{cg} is connected to external voltage to bias the FG transistor. The selected cell is configured as following:

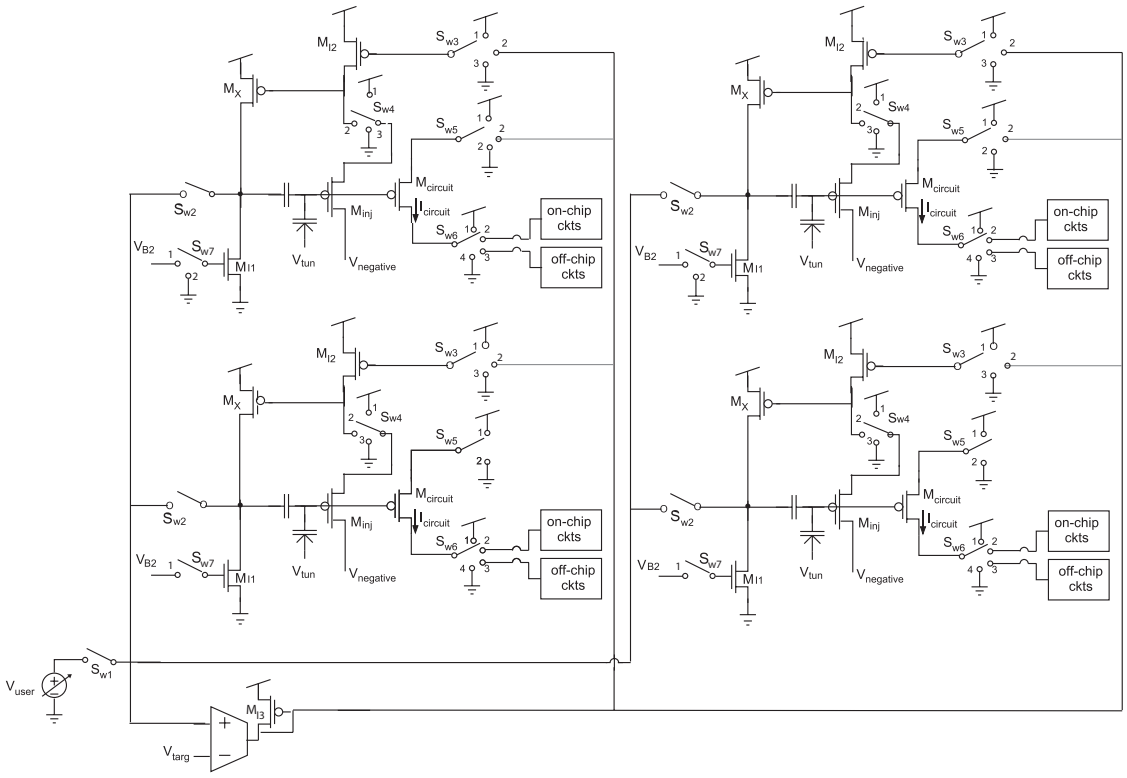

Figure 4.6: Two-by-two array architecture for the memory cell and programmer for below-ground injection.

1. S_{w1} and S_{w2} are closed to connect V_{cg} to the external voltage to bias $M_{circuit}$. In read more V_{cg} is shared among all the FG cells in the array, and V_{sd} of $M_{circuit}$ determines if the transistor connected to the application.

2. S_{w3} and S_w4 are set at position 1 to set the source of M_{inj} at V_{dd}. This is because in read-mode the output of negative charge-pump is at the supplt voltage V_{dd}. Therefore setting the source of M_{inj} at V_{dd} will provide $V_{sd} \approx 0$.

3. S_{w5} is at position 1 to supply V_{dd} at the source of $M_{circuit}$. While S_{w6} is at either position 2 or 3 to connect $M_{circuit}$ transistor to either off-chip or on-chip application circuit.

Overall, the use of indirect injection enables below-ground injection if FG memory array. Still, few switching circuits are needed to select/deselect a specific cell in the array. However,

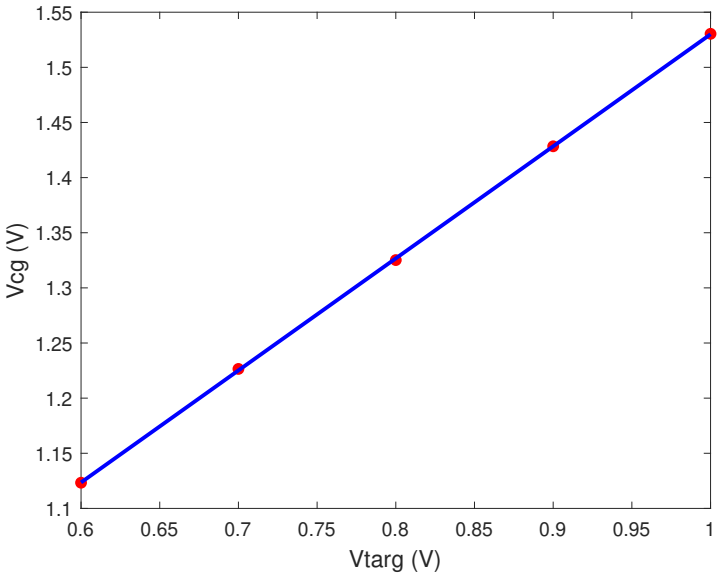

Figure 4.7: Linear injection results using below-ground injection with current-conveyor configuration with a slop of 1.011

those switches are not exposed to the negative voltage which makes below-ground injection of FG array is viable.

4.3.3 Accuracy Measurements

I examined the presented memory cell and programmer circuit to measure injection linearity and repeatability. Testing methodology used to take measurements are similar to the methodologies used for above-ground injection in chapter 3. I administered those test on a fabricated FG array and on-chip programmer fabricated in $0.35\mu m$ standard CMOS process available through MOSIS. First measurements were taken to confirm the linear injection characteristic of FG cell. Fig. 4.8 illustrates the linear relation between V_{cg} and V_{targ} with line-fit slope of 1.011.

Secondly, I ran a test to measure repeatability accuracy in term of the negative injection voltage. In this test, different below-ground voltages were used. For each V_{blw} the FG cell was injected to a range of V_{targ}, and for each V_{targ} injection process was repeated 100 iteration. Then current measurements were mapped back to voltage using reference cell. The circuit gave repeatability accuracy of 0.95mV at worst case, on the other hand the accuracy measurement were as low as 0.73mV. It is important to note that different V_{blwgnd} do not

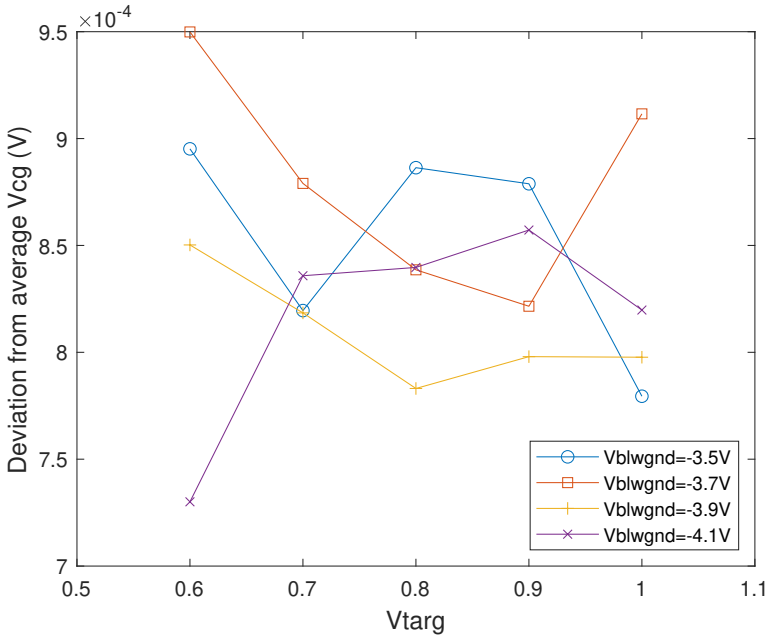

Figure 4.8: Linear injection results using below-ground injection with current-conveyor configuration with a slop of 1.011

affect accuracy, unlike above-ground injection. This is considered a very desirable quality for programming, as injection voltage is supplied by the charge-pump and with the ripple at the output of the charge-pump and the different applications FG could be used with, it is more functional to get target current that is independent on injection parameters.

4.4 OTA Programmer Circuit

A second iteration of the below-ground circuit from 4.3 is presented here. This circuit is an improvement of the original one and is expected to provide more accurate programming results. Instead of using a single transistor to implement negative feedback between the source and control gate of the FG transistor, an Operational Transconductance (OTA) is used. The need for this improvement stems from the fact that the FG transistor source voltage could be controlled externally using the negative feedback characteristic of virtually connecting the two inputs. Additionally, with adequately designed OTA characterized with high gain, a better accuracy should be achieved.

4.4.1 FG Memory Cell

The improved memory cell structure is shown in Fig. 4.9. Similar to the previous configuration presented in 4.3.1, this configuration utilizes the indirect programming of FG, however, instead of including the negative feedback common-source amplifier within the FG cell itself, the negative feedback amplifier is placed with the programmer circuit. Consequently, the newer version is introducing a smaller and more compact memory cell than the original cell. Originally as shown in Fig. 4.2, the FG cell consists of: the two indirect transistors M_{inj} and $M_{circuit}$, the negative feedback common-source amplifier M_1, a current mirror transistor represented by a current source I_1, current sink transistor represented as the current source I_2 and finally all the switching circuitry needed to switch between different operating modes.

The new cell consists of only the two indirect FG transistors M_{inj} and $M_{circuit}$ with the switching selection circuit at their terminals, which would significantly reduce the die area of the whole FG array on chip specially with hundreds of FGs are used on a single chip. As previously discussed, the transistor M_{inj} is used in program-mode to inject the floating-gate node while transistor $M_{circuit}$ is used in read-mode to bias programmable analog application.

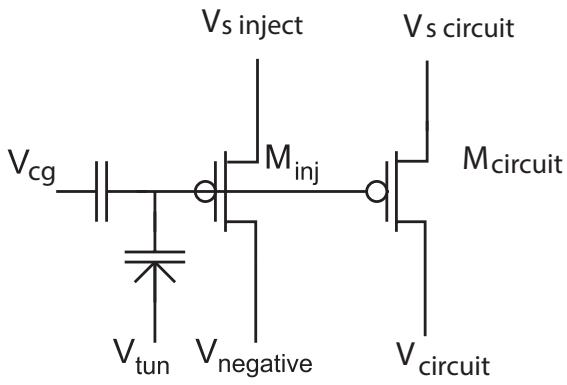

Figure 4.9: FG cell structure used in below-ground injection with OTA feedback

4.4.2 FG Array and Programmer Circuit

A programmer circuit is needed to stop injection when a target voltage is reached. Similar to the original programmer circuit used in 4.3.2, this programmer mainly consists of the comparator OTA that compares V_{cg} with V_{targ} as injection progresses and provides a current through FG channel that is proportional to the difference between the two voltages. One

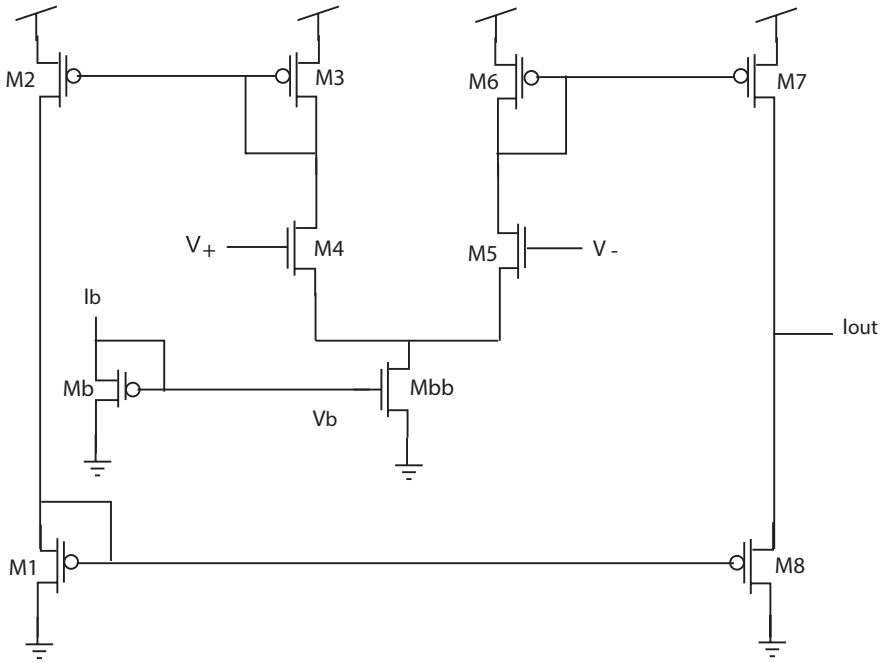

Figure 4.10: 9-transistors negative feedback OTA structure used in the programmer circuit between the source and the control-gate.

more distinguished element added in this newer programmer circuit is an additional OTA that serves as a neagtive feedback amplifier between the source and control-gate of the FG transistor. The negative feedback OTA used in this circuit is a 9-transistor OTA, a schematic of the OTA is shown in Fig. 4.10.

Fig. 4.11 illustrates the new configuration with OTA negative feedback. The noninverting input of the OTA is pinned out while the inverting input is connected to FG source terminal, this creates negative feedback path between the source and control-gate. This configuration allows the user to externally control the source voltage of the FG transistor by just setting the voltage V_{inp}. The programmer OTA used in this configuration is similar to the one used in the previous below-ground configuration and works here similarly as a comparator.

To illustrate how the new version is used in an array configuration, a two-by-two array is shown in Fig. 4.12. As I previously mentioned, the transistor M_{inj} is used in program-mode to inject the floating-gate node and only one feedback OTA is shared among all the FG cells in the array. From the array figure the selection process to connect one of the FG cell to the programmer circuit in program-mode is accomplished as following:

1. S_{w1} is opened while switch S_{w2} is closed to connect V_{cg} to the programmer OTA.

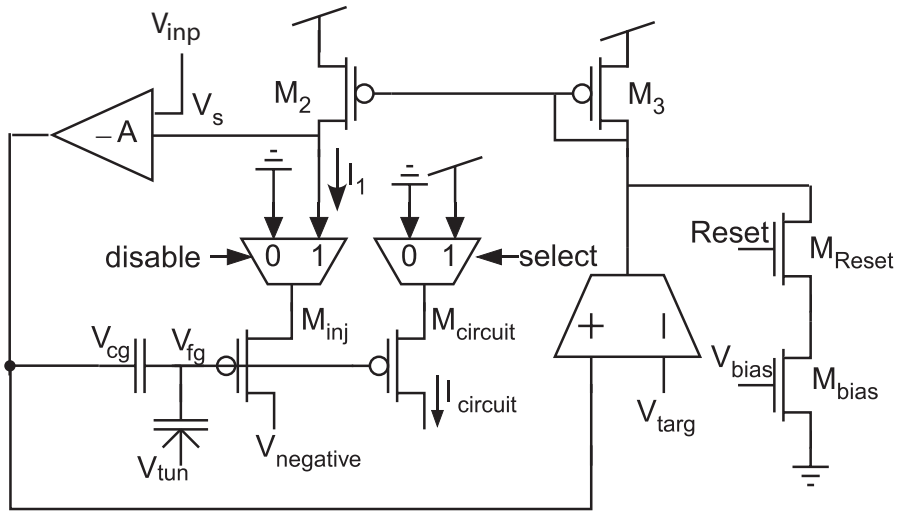

Figure 4.11: Complete negative voltage programming structure with OTA for negative feedback

2. S_{w5} is set at position 2 to connect the current mirror M_{I2}-M_{I3} to the source of the FG.

3. Finally, switches S_{w3} and S_{w4} at the source and drain of $M_{circuit}$ of the selected cell are set at positions 2 and 4 respectively, to place $M_{circuit}$ in dormant state.

On the other hand, unselected FG cells in program-mode are configured as following:

1. S_{w1} and S_{w2} are opened to disconnect the unselected cells' V_{cg} form any voltage.

2. S_{w5} is set at position 3 to connect the FG source to ground, resulting in small V_{sd} not enough for injection.

3. For all $M_{circuit}$ in unselected cells, S_{w3} is at position 2, while S_{w4} is at position 4 to neutralize trnasistor $M_{circuit}$.

In read-mode, V_{cg} is supplied externally to bias the injected FG transistor and the result biasing current is read through the drain of $M_{circuit}$. The selected FG cell is configured as following:

1. S_{w1} and S_{w2} are closed to connect V_{cg} to the external voltage to bias $M_{circuit}$.

2. S_{w5} is set at position 1 to set the source of M_{inj} at V_{dd}. Consequently, the FG V_{sd} will not be ≈ 0.

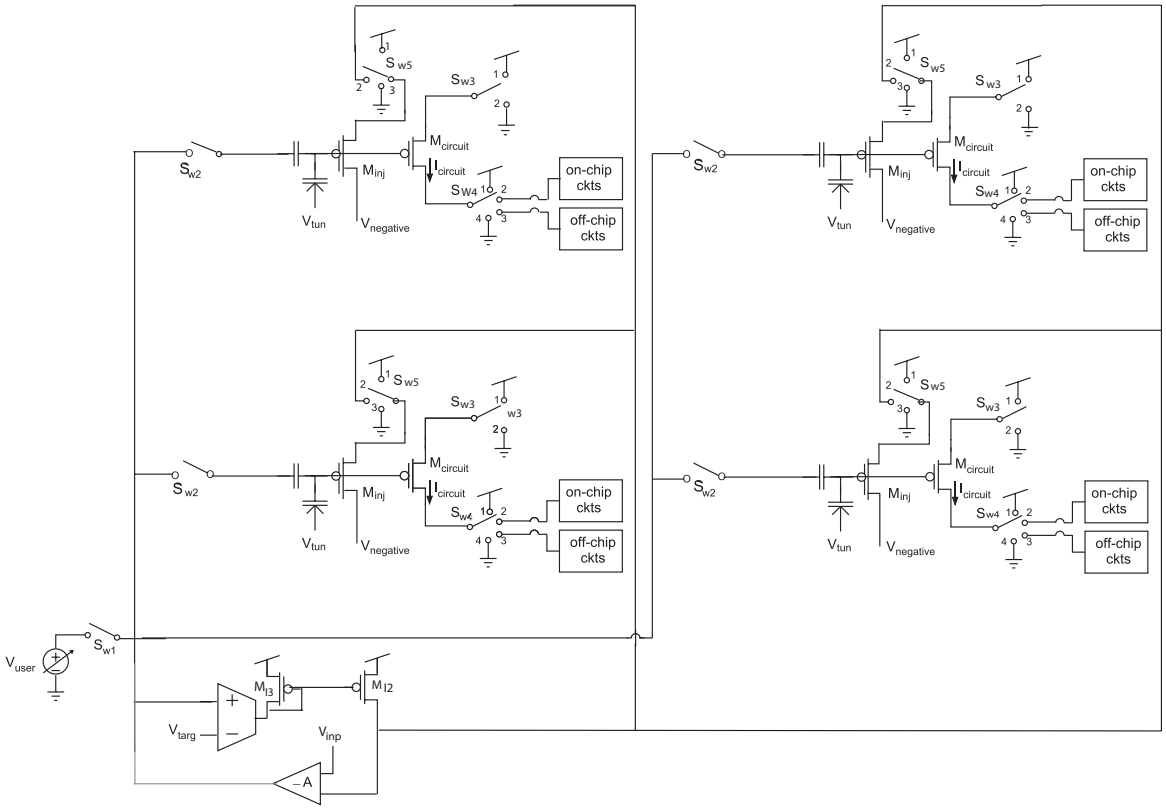

Figure 4.12: Two-by-two array architecture for the memory cell and programmer for below-ground injection.

3. S_{w3} is at position 1 to supply V_{dd} at the source of $M_{circuit}$. While S_{w4} is at either position 2 or 3 to connect $M_{circuit}$ transistor to either off-chip or on-chip application circuit.

4.4.3 Accuracy Measurements

To test the new circuit accuracy and repeatability few tests were conducted. Those tests were administered on a fabricated circuit (memory cell and programmer) in $0.35\mu m$ standard CMOS process available through MOSIS. Firstly, current measurements were taken while applying different values for different injection parameters such as V_{targ}, V_{blwgnd}, different biasing for programmer OTA and different V_{inp} at the FG source. Those measurements clarifies how injection accuracy would be affected whenever any of those parameters change.

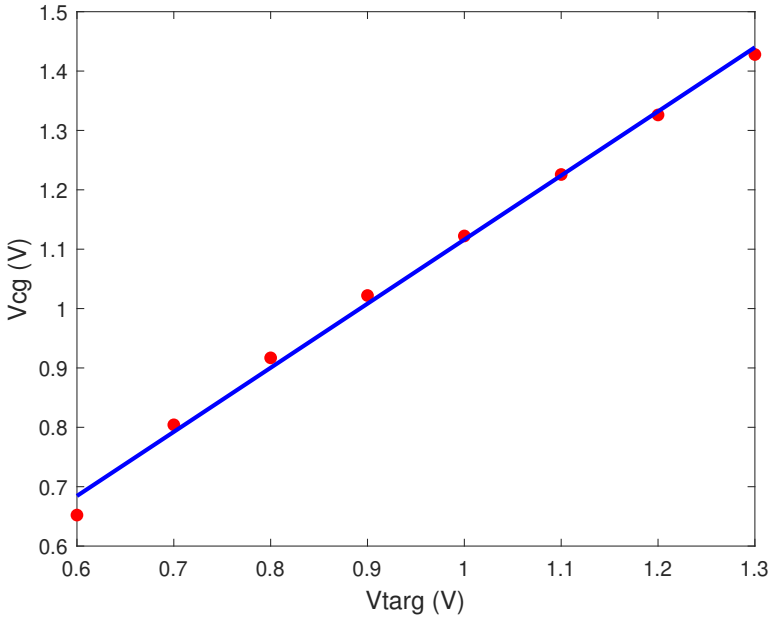

Figure 4.13: Linear injection results using below-ground injection with OTA configuration with a slope of 1.079

Additionally, a comparison between the two below-ground injection techniques is conducted to determine the best configuration in term of overall circuit size, repeatability and accuracy. Finally, I compare between the winner below-ground programmer and the above-ground programmer circuit. The best programmer of the three topologies will be used in the next iteration of the RAMP.

The methodology I used to conduct accuracy measurements here is similar to the one I used for above-ground and older below-ground programmers, where a reference IV-curve is used to map current measurements back to V_{cg}. This methodology makes it possible to not only test the performance of a one topology itself but also to compare the measurements from different topologies.

To confirm the effectiveness of using OTA as negative feedback amplifier to linearize injection process, V_{cg} vs. V_{targ} relation is shown in Fig. 4.13. These measurements were taken by finding V_{cg} that would provide a target current of $100\mu A$ for different target voltages. The slope of the line is 1.079. I believe I could get better slope if a bigger target voltage range is used.

To study the effect of the below-ground voltage V_{blwgnd} on injection accuracy and repeatability, different values for below-ground voltages were used. Since the rated supply

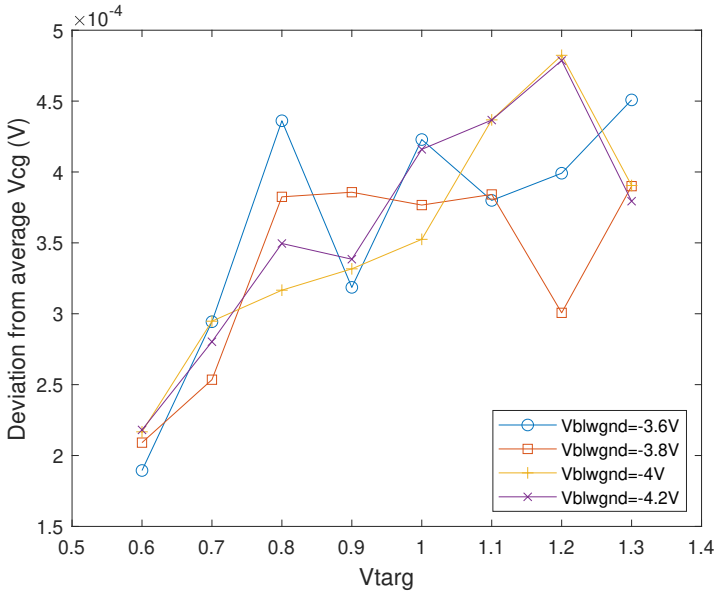

Figure 4.14: Measured repeatability accuracy for different below-ground voltages using OTA feedback programmer

voltage for $0.35\mu m$ process is 2.5V, the least V_{blwgns} used was -3.6V to create a V_{sd} of 6.1V enough to progress with injection process. Fig. 4.14 shows repeatability measurements for 100 iterations using 4 different V_{blwgnd}. These measurements were taking under the following conditions: V_{inp} = 2V, biasing current for comparator OTA = $2\mu A$, V_{blwgnd} = -3.6V, -3.8V, -4V and -4.2V, the target current = 10 μA. As demonstrated on the figure, deviation form average V_{cg} is not dependant on V_{blwgnd}. The results does not have specific pattern that follows specific change in the below-ground voltage. I believe this is because the drain voltage does not directly affect the injection rate resulting in measurements that does not reflect any change in it. This is congruous to the results from belwo-ground programmer with current-conveyor feedback.

Programming accuracy for below-ground circuit was calculate. The FG memory cell was programmed for 100 iteration for a target voltage range of 0.7V (0.6V to 1.3V). A max V_{cg} standard deviation of $314\mu V$ was observed. This yields to a programming accuracy of 11.12 bits.

Other measurements were taken to study the effect of different injection parameter on programming accuracy. One experiment was to test the affect of the programmer OTA biasing current on programming accuracy. I found out that biasing current does not enormously affect programming accuracy, however, better accuracy is achieved when biasing OTA in

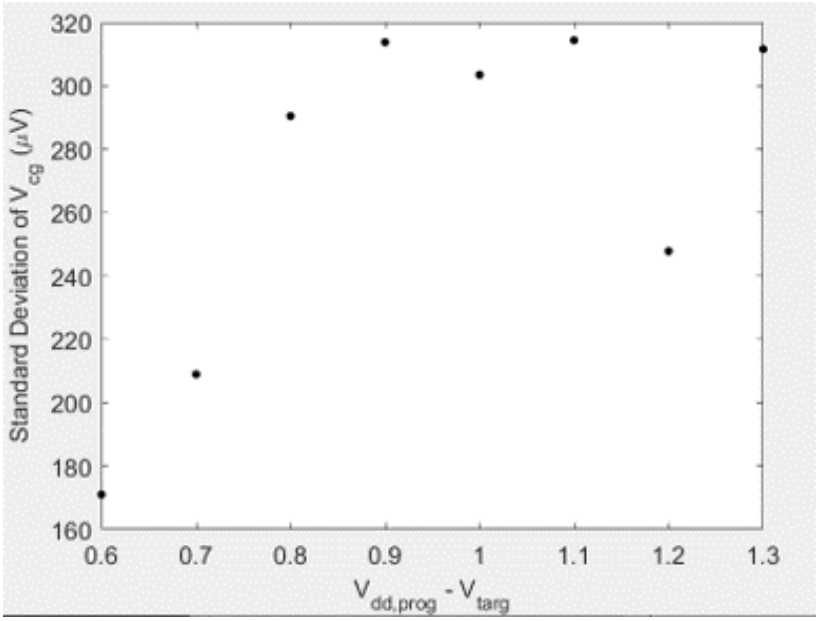

Figure 4.15: Measured repeatability accuracy.

subthreshold. Additionally, the other parameter that has been tested is the source voltage that is controlled through the feedback OTA V_{inp}. It was found that even though this voltage would affect the IV-curve of the FG transistor and affects the threshold voltage, it does not have a major impact on accuracy.

4.4.4 System Application: C^4 Programming

To test the new programmer we used the memory array to bias Capacitively-Coupled Current Conveyor (C^4) used in 3.6. The C^4 filterbank was designed and fabricated on chip with the FG memory array on a 0.35μm CMOS process. A die photograph of the chip is shown in Fig. 4.16. The FG memory array were used to tune 8 bandpass filter bank. Each column (two FG cells) in the array were used to determine the high and low frequency of the bandpass filter.

Fig. 4.17 shows the results performing frequency decomposition for various bandwidth and filter spacing by tuning C^4 bandpass filter bank. Quality factor (Q) for each configuration was selected so that the filters cross at their -3dB points. Fig. 4.17 (a) shows the results of octave spacing with Q $\sim$ 1.4 and a center frequency starting at $88Hz$, (b) shows the results of half-octave spacing with Q $\sim$ 2.9 and a center frequency starting at $300Hz$, (c) shows the results of third-octave spacing with Q $\sim$ 4.3 and a center frequency starting at $445Hz$.

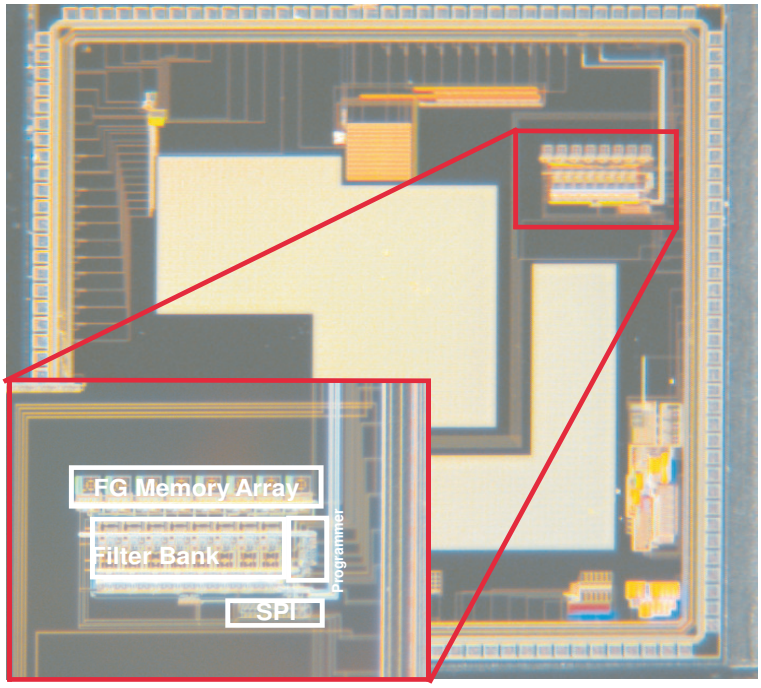

Figure 4.16: A die photograph of the circuit showing the FG memory array, the programmer circuit, the C^4 filterbanck and an SPI used to control programming signal.

4.5 Conclusion

Below-ground injection of FG transistor is made possible with the use of indirect programming which employs two transistors sharing the same floating-gate node. One transistor (injection transistor) is used during program-mode to inject charge on the floating-gate while the other transistor (circuit transistor) is disabled. On the other hand, the circuit transistor is used in read-mode while injection transistor is disabled. Two different below-ground configurations presented in this chapter. The first configuration utilizes a common-source single transistor amplifier as a negative feedback path to linearize injection process. Conversely, in the second configuration, a 9-transistor OTA is used to implement the negative feedback.

The two presented FG cell and programmer circuit were used to successfully design and build FG memory array. Only one programmer circuit is needed per each array. However, for the OTA feedback version, the feedback OTA is included within the programmer circuit yielding to more compact FG cell which is more adequate for FG array applications.

Accuracy measurements form both circuits showed that both circuits provides similar results. Overall and after comparing the three different programming topology, I concluded that all three circuits provide similar repeatability results that is accurate enough for most of

analog applications. However, the following specific findings are concluded: a) above-ground injection provides the best accuracy results, a little bit better than below-ground with OTA, while below-ground with current-conveyor presents the worst accuracy. b) even though above-ground is more accurate, its accurate is affected by the injection voltage FGV_{dd}, while below-ground injection is independent on the injection voltage V_{blwgnd}. c) finally, below-ground with OTA utilizes more compact cell and programmer configuration which makes this approach as the most suitable methodology for FG-dense analog application.

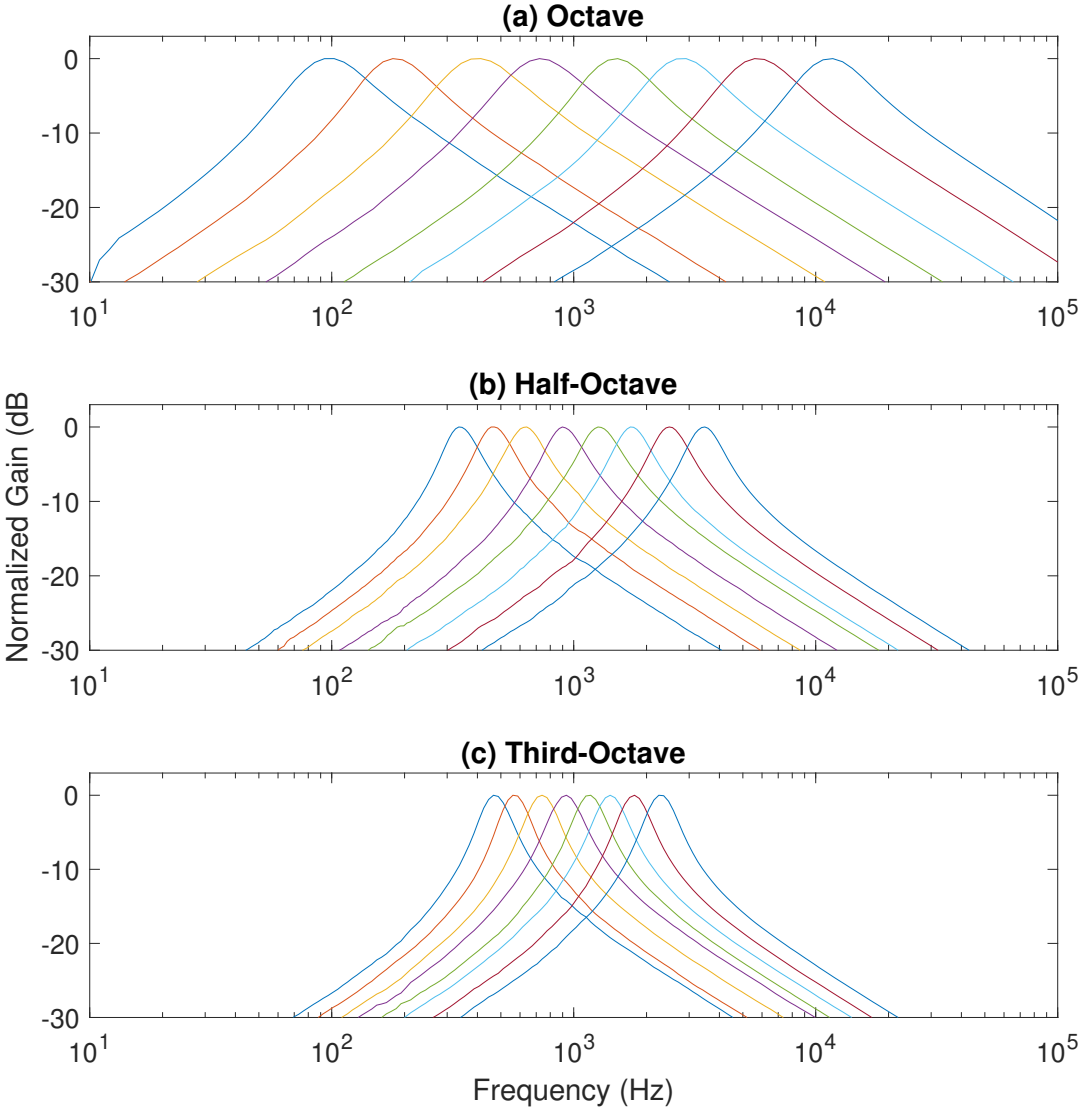

Figure 4.17: Programmed $\mathbf{C^4}$ array frequency responses for below-ground programming. (a) octave spacing starting at $\mathbf{f_c = 88Hz}$, (b) half-octave spacing starting at $\mathbf{f_c = 300Hz}$, and (c) third-octave spacing starting at $\mathbf{f_c = 445Hz}$.

Chapter 5

Programmer Circuit with Operational Amplifier Feedback

The programmer circuit discussed in the previous chapters share the same main configuration, the negative feedback amplifier. It is observed that the programmer circuit with OTA shows better performance than the one with the single transistor. Therefore, in this chapter, a new yet-similar programmer circuit is introduced that is expected to excel and is anticipated to show much better accuracy performance. In this programmer circuit, a high-gain operational amplifier is used as the negative feedback. Two different operational amplifiers were designed. One prototype programmer circuit was fabricated on $0.5\mu m$ CMOS process. The programmer circuit was designed to work with both injection mechanisms—above and below ground.

5.1 Programmer with opamp

The three programmer circuits presented in the previous chapters used either a single transistor or an OTA at the negative feedback of the programming configuration. They provided good feedback characteristics and were successful to linearize injection. However, using an amplifier with much higher gain would promote the negative feedback characteristics and consequently improve the performance and the overall accuracy of the circuit. Here, I present a programmer circuit that utilizes a high-gain opamp as the negative feedback.

A prototype programmer circuit with two-stage opamp was designed to be tested with both above and below ground injection. The programmer circuit that includes the com-

parator OTA and the feedback opamp with the current mirror was provided with switching multiplexers. Two models of the FG cell were included on the chip for testing purpose and to be connected externally to the programmer circuit. The programmer circuit with opamp is shown in Fig. 5.1. When connecting FG cell to the programmer, the source of the FG is to be connected to V_S and the control gate is to be connected to V_{cg}. The two multiplexers at the source and control-gate switches between the different operation modes (read/program). The two stage opamp used within the programmer circuit is shown in Fig. 5.2.

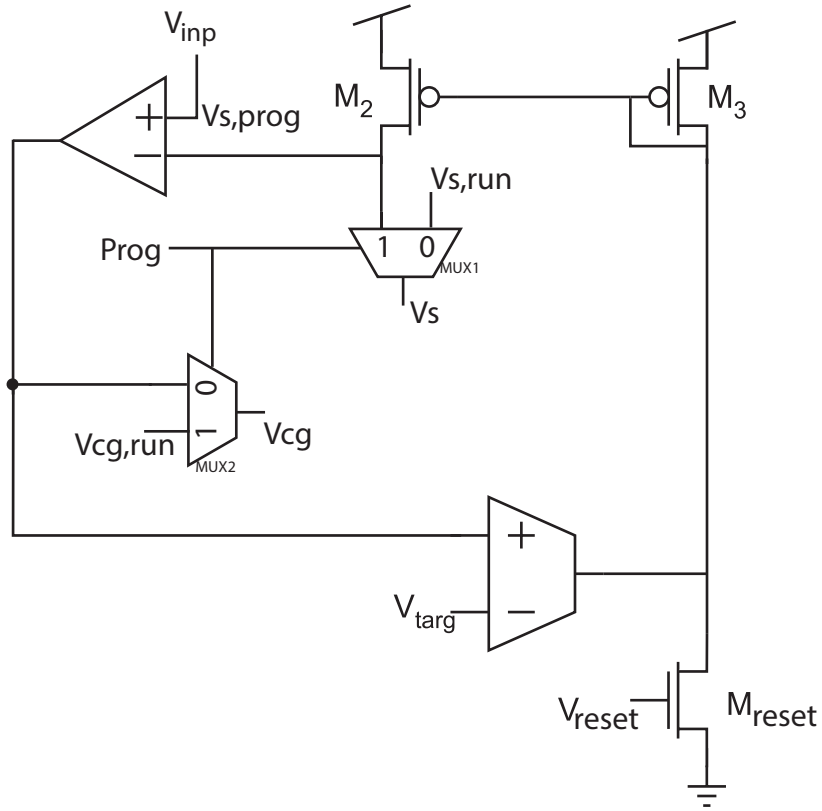

Figure 5.1: Prototype Programmer Circuit with Opamp

From Fig. 5.1, in program mode, the prog signal is set to high to connect the FG source to the negative feedback opamp and the programmer circuit. Additionally, the FG control-gate is connected through the other multiplexer to the output of the feedback opamp and the programmer's comparator OTA. This configuration is similar to the below-ground programmer circuit presented in Fig. 4.11. Similarly, V_{inp} here is used to set the voltage at the source of the FG. When operating the circuit in above-ground injection, the V_{dd} of the circuit is raised to a high enough voltage to induce injection. The voltage V_{inp} is set very close to V_{dd} while the drain of the FG is held at ground. On the contrary, when operating

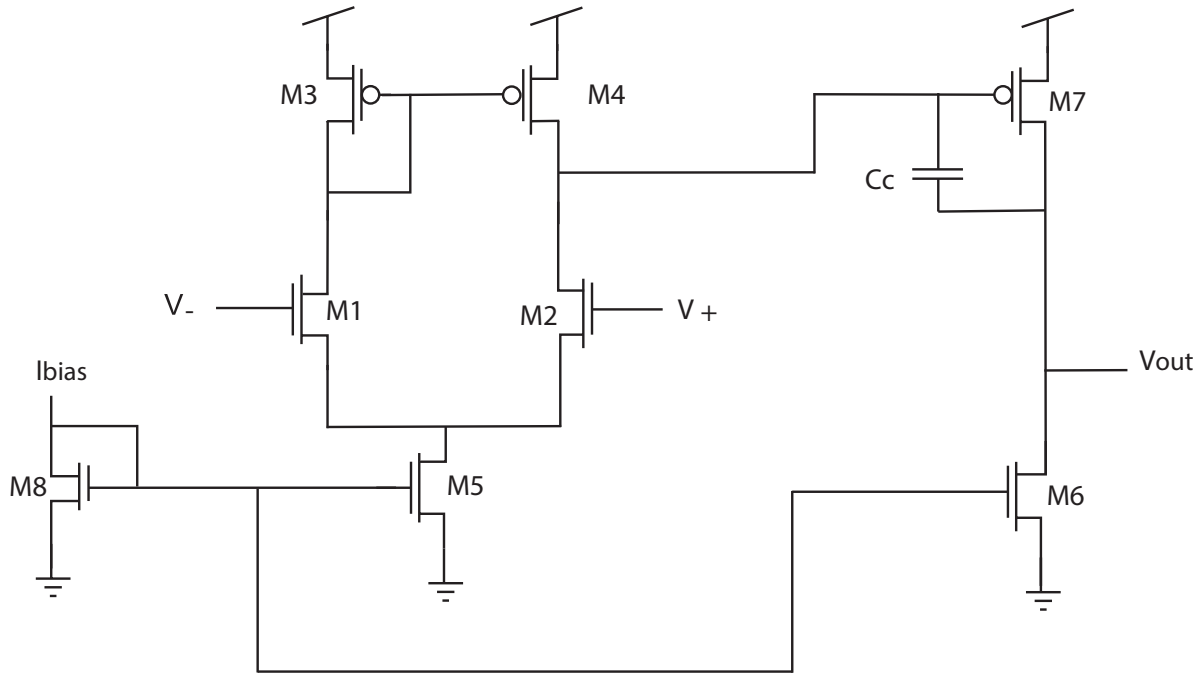

Figure 5.2: Two-stage opamp schematic

in below-ground injection, the V_{dd} is held at the process supply voltage (3.3V for $0.5\mu m$), while the drain of the FG transistor is lowered to a negative voltage. A transient analysis simulation of the programmer circuit when operated in above-ground injection is shown in Fig. 5.3.

In read mode, the prog signal is set at ground which will connect the FG source to the external voltage, $V_{s,run}$, that is set at the V_{dd} of the process. Likewise, the FG control-gate is connected to the external voltage, $V_{cg,run}$, through the other multiplexer. This setup will place the FG cell in current-read mode, and the application circuit can be placed at the drain of the FG transistor.

5.2 Folded Cascode Opamp Design

Since the opamp is used in the feedback path of the programmer circuit which, in addition to linearizing injection, can provide an external control for the source node. It was important to design an opamp with a very good ICMR so we can get a source voltage that is as close to V_{dd} as possible. Therefore, a folded-cascode opamp was designed to improve the overall performance of the programmer circuit.

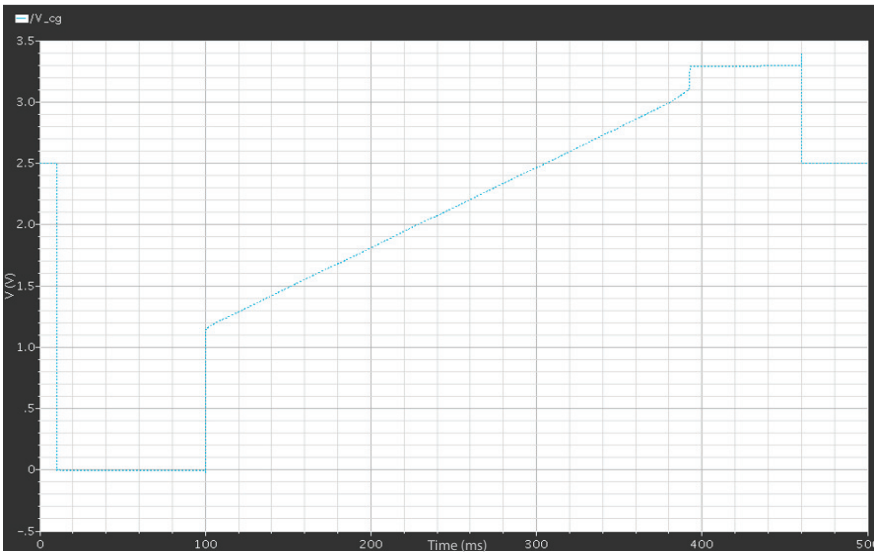

Figure 5.3: Transient simulation of the programmer circuit with opamp

A folded-cascode opamp is a differential opamp that utilizes cascoding of an n-fet and p-fet transistors. This configuration allows more headroom voltage and results in a wider operating range. The folded-cascode opamp offers a gain approximately equal to that of a two stage opamp, a good ICMR, and self-compensation [54]. Fig. 5.4 shows an n-fet input folded-cascode opamp. In this work, the folded-cascode opamp was designed to achieve the following characteristics:

1. A $|gain| \geq 10k$
2. Input Common Mode Range of $0V - 3.3V$.
3. Slew Rate $= 2V/\mu s$.
4. a Gain Bandwidth Product $= 5MHz$

5.2.1 Folded-cascode opamp Design Procedure

In this work, I used the n-fet input opamp configuration shown in Fig. 5.4 and followed the design approach presented in [54]. To meet the aforementioned opamp requirements, the following design is calculated:

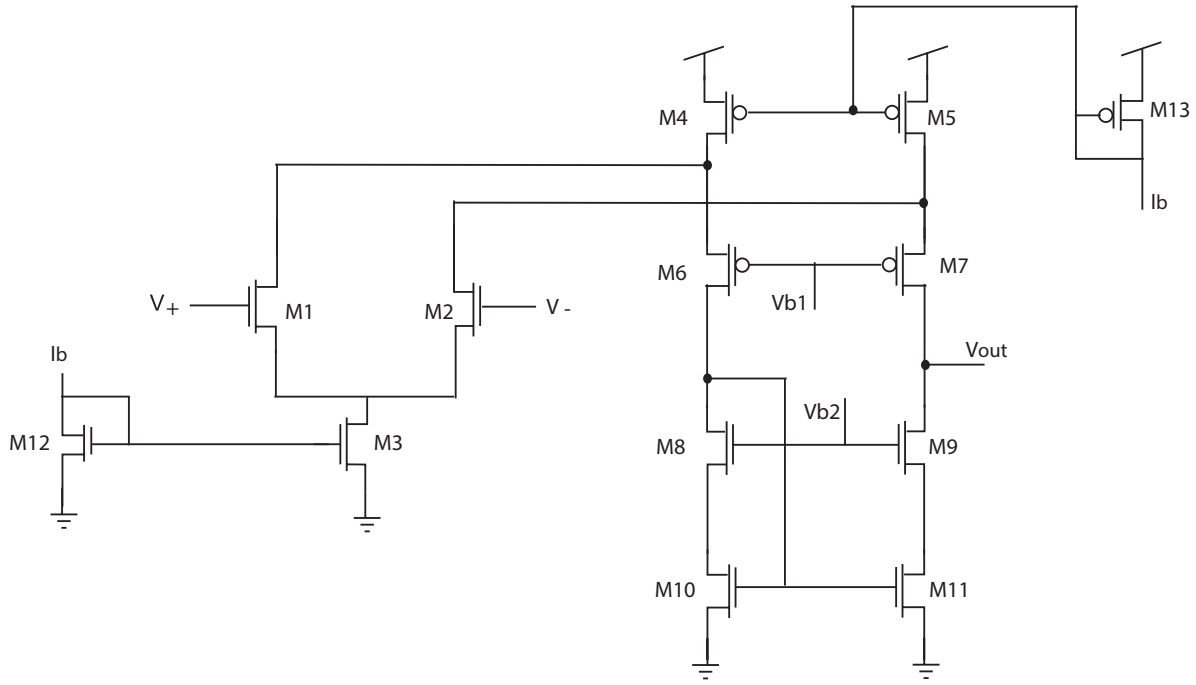

Figure 5.4: Folded-cascode opamp schematic

1. Using slew-rate requirement, and assuming that the load capacitance is $30pF$, the biasing current I_3 is calculated:

$$I_3 = SR.C_L \tag{5.1}$$

This yields to a biasing current I_3 of $60\mu A$ for transistor M_3 and biasing currents I_4 and I_5 of $80\mu A$ for the cascode mirror.

2. Since this opamp is designed to be fabricated for a $0.5\mu m$ process and giving that the supply voltage is $3.3V$, transistors M_4, M_5, M_6 and M_7 were designed:

$$S_4 = S_5 = S_6 = S_7 = \frac{2I_5}{\kappa'_P V^2_{SD5}} \tag{5.2}$$

Solving Equ. 5.2 yields to transistor ratio of at least 320, which meets the ratio required for maximum input CM from the equation below:

$$S_4 = S_5 = \frac{2I_5}{\kappa'_P(V_{DD} - V_{in}(max) + V_{T1})} \tag{5.3}$$

3. Similarly, the following equation is used to solve for transistors M_8, M_9, M_{10}, M_{11}:

$$S_8 = S_9 = S_{10} = S_{11} = \frac{2I_9}{\kappa'_N V^2_{DS9}} \tag{5.4}$$

Table 5.1: Folded-cascode opamp Design

Component	W (μm)	L (μm)
M_1 - M_2	270	2
M_3	220	10
M_4 - M_7	700	2
M_8 - M_{11}	290	2
M_{12}	20	10
M_{13}	40	2

This yields to transistor ratio of at least 145.

4. Using gain-bandwidth requirement, knowing the biasing current I_3 from step 1, transistor ration for M_1 and M_2 is calculated:

$$S_1 = S_2 = \frac{GB^2 C_L^2}{\kappa'_N I_3} \tag{5.5}$$

which yields to a transistor ration of 134.45.

5. Finally, from the minimum input CM and the biasing current I_3, I calculated transistor M_3 ratio:

$$S_3 = \frac{2I_3}{\kappa'_N (V_{in}(min) - V_{ss^-} \sqrt{\frac{I_3}{\kappa'_N S_1}} - V_{T1})^2} \tag{5.6}$$

which yields to transistor M_3 with the ration of 22.

Based on the above calculations and values, a folded-cascode opamp circuit was designed and simulated. Table 5.2 shows the transistor parameters for the final design for the folded-cascode opamp.

5.2.2 Simulation Results

The opamp designed in the previous subsection was simulated in LTspice to confirm that it meets the design requirements. The following simulation results were observed.

Table 5.2: Folded-cascode opamp Biasing Parameters

Biasing Parameter	Value
$Ibias$ (μA)	5
Vb_1 (V)	2
Vb_2 (V)	1.2

Table 5.3: Folded-cascode opamp DC Parameters

Parameter	Measurement
Gain	10k
Gain (dB)	79.5
High ICMR (V)	4
Output Range (V)	400m - 2.93

DC Parameters

Table 5.4 shows all the DC parameters that were measured for the folded-cascode opamp. A gain of 10k and a wider ICMR were achieved, which expected to improve the accuracy of the FG programmer circuit.

Frequency Response and Slew-Rate

Since the folded-cascode opamp configuration is self compensated, there was no need for Miller compensation. A phase margin of at least 85^0 was observed for the proposed design to ensure stability during the expected operating conditions. The frequency response of the opamp is shown in Fig. 5.5. The gain bandwidth of the opamp was measured to be 3MHz.

Since this opamp is going to be used as the negative feedback amplifier for the programmer circuit, it was important to ensure that the opamp has a good slew-rate that fits the circuit requirements. A slew-rate of at least 2.3 $V/\mu s$ was observed which meets the requirement for the intended programmer application.

Table 5.4: Folded-cascode opamp AC Parameters

Parameter	Measurement
Unity-Gain Frequency (Hz)	3M
Phase Margin	85.3ˆ0
Slew-Rate ($V/\mu s$)	2.3

5.3 Conclusion

To improve the performance of the programmer circuit, a higher-gain opamp is to be used as the negative feedback amplifier. A prototype programmer circuit with an opamp is designed to work with both above-ground and below-ground injection. A folded-cascode opamp is designed to provide a better input common-mode ratio (ICMR), so during injection the source can be placed to a voltage that is very close to the supply voltage. The programmer circuit with the folded-cascode opamp is to be designed and tested.

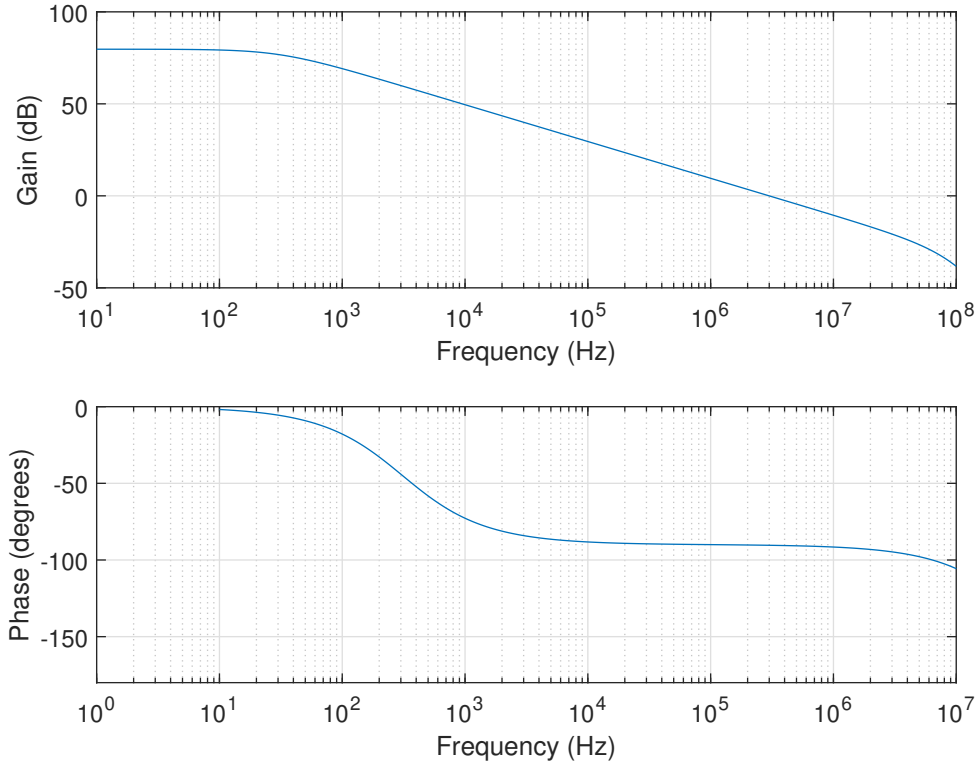

Figure 5.5: Bode Plot Output from AC analysis.

Chapter 6

Reconfigurable Analog Mixed-Signal Platform

Since 1984, when Altera introduced its first programmable array logic (PAL), many reconfigurable circuits have been released. Nowadays, the Field Programmable Gate Array (FPGA) is the most well-known reconfigurable platform. FPGAs are integrated circuits designed to be configured depending on the application they are used for. Field-Programmable Analog Arrays(FPAA) are the analog equivalent of an FPGA. FPAAs are used to provide application-driven hardware pre-processing circuits which maintain minimal power consumption [55, 56]. An FPAA allows for post-fabrication analog circuit design and implementation. Early examples of FPAAs are presented in [57, 58, 59].

The Reconfigurable Analog Mixed-Signal Processor (i.e. RAMP) [2] is a field-programmable analog array (FPAA) developed by the Computational Electronic Systems laboratory (CESLAB) and is used to develop custom, low-power applications. In this chapter, I will briefly explain what the RAMP is and how to program it to synthesize analog applications. Additionally, I will explain why FG transistors are crucial components for reconfigurable analog systems, and specifically how FG transistors are integrated in the RAMP.

6.1 RAMP System Architecture

The two main characteristics of the RAMP are: 1) low-power consumption and 2) reconfigurability to build different circuits. The RAMP IC consists of different analog and digital components that can be connected in any configuration to synthesis general-purpose

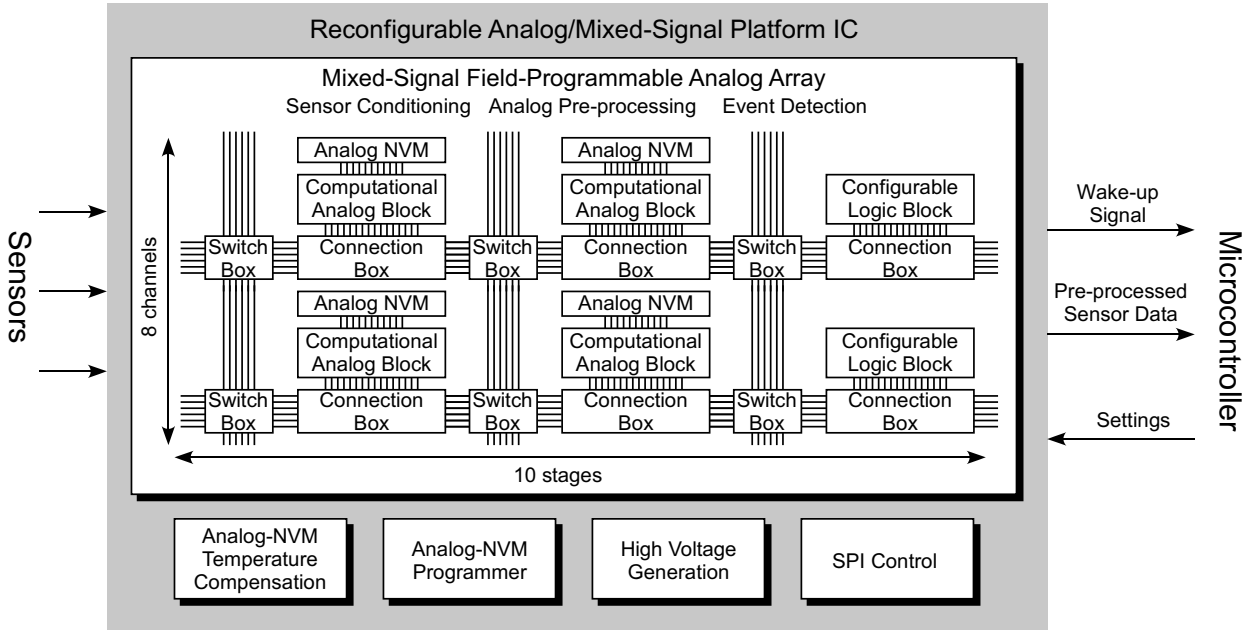

Figure 6.1: Architecture of the reconfigurable analog/mixed-signal platform (RAMP) integrated circuit.

or customized analog signal processing. The RAMP platform consists of two fundamental parts: reconfigurable platform and software infrastructure. The former is the actual hardware and components that the are used to synthesis different applications while the later is the programming infrastructure that allows the user to reprogram and reconfigure their application circuit.

6.1.1 Reconfigurable Platform

The main purpose for developing the RAMP is for users/ circuit designers to be able to synthesis analog signal processing circuits depending on the application requirements. Sensor interfacing applications can exploit the RAMP to create common analog signal processing circuits such as filters. Some of the components included on the RAMP are amplifiers, single transistors, inverters, comparators and current mirrors.

As shown in Fig. 6.1, the RAMP consists of an array of the following components:

1. Computational analog blocks (CABs) that consist mainly of different circuit building blocks such as resistors, capacitors and transistors, and analog signal processing blocks such as programmable filters, opamps and sample-and-hold circuits.

2. Configurable logic blocks (CLBs) to provide control signals for the analog parts. CLBs

Table 6.1: Componentson the RAMP 1.0.

8 BPFs	56 OTAs	8 Inverters	16 Envelope Detectors
8 LPFs	8 Multipliers	32 Comparators	48 Current Sources/Sinks
56 Caps	9 Op-Amps	8 Bump Circuits	16 Pulse Generators
8 PNPs	16 Resistors	8 Time-to-Voltage	16 Asymmetric Integrators
16 S/Hs	144 FETs	32 JK Flip Flops	16 6-input 2-output LUTs

represent a mini FPGA inside the RAMP and consist of digital components such as filp-flops and look-up tables.

A list of the computational components used on the RAMP are shown in Table 6.1. Those CABs and CLBs are arranged in 10 stages (columns) and 8 identical channels (rows) to realize parallel aspect in signal processing. The 10 stages are consists of 8 analog stages and 2 digital stages. This would bring the total to 80 CABs/CLBs on a single RAMP. To allow the user to connect specific component to create a specific configuration, matrix of a total of 20,380 of switches is used. Those switches are implemented as an SRAM-controlled T-gate. Those switching circuits are divided into two types:

1. A connection box is a crossbar switch that permits the selection between elements within the same CAB/CLB.

2. A switch box is a 4-way switch to permit the connection from one CAB/CAB to another CAB/CLB.

An SPI (serial Peripheral Interface) is used on-chip to load the circuit design of which switch is turned on/off into the RAMP. Each stage is designed for a specific function and different components are placed on its CAB/CLB accordingly to give the use flexibility to implement various applications. A die photograph of the RAMP chip is shown in Fig. 6.2.

Reconfigurable analog application required programmable analog components which require biasing voltage or current to precisely set their parameters. However, with a platform as big as the RAMP and with all those circuit building blocks, it is complicated to provide those biasing values while keeping power consumption and die area minimal. This is where analog nonvolatile memory (NVM) 6.1 (i.e. FG transistors) play a major rule and make reprogramability both possible and feasible. By accurately programming floating-gate transistors, an exact biasing current or voltage can be generated which allow the user to

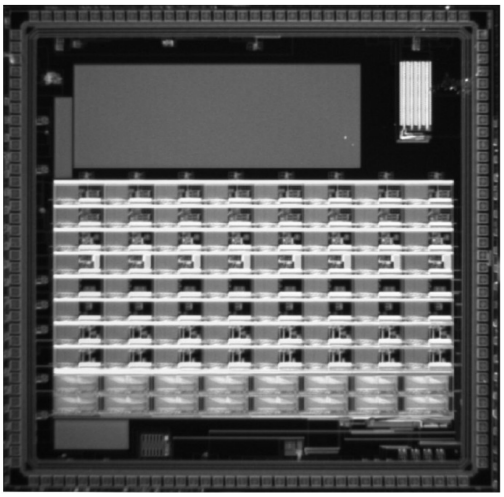

Figure 6.2: A die photograph of the reconfigurable analog mixed-signal Platform (RAMP)

precisely set the programmable parameters such as providing a biasing current to set an amplifier transconductance. An above-ground programmer similar to the one presented in chapter 3 is used for injecting Analog NVM on the RAMP, which is represented as analog-NVM programmer in Fig. 3. The RAMP contains a total of 296 programmable parameters to be set by analog NVM. All the user has to do is to design their circuit and specify their programmable parameter, the RAMP will do the rest of the work such as translating those parameters into a specific amount of FG charge or such as setting the appropriate switches to be on/off.

6.1.2 Programming Infrastructure

To build a specific application circuit using the RAMP, the user needs to specify what components they need to build their application circuit ans what programmable parameter to set to. To specify circuit components, the RAMP utilizes netlist representation of the circuit where the user needs to specify which component and what nodes this component is connected to. The user could use either Matlab or Octave as their interface to program and compile their netlist using the compilation platform developed for the RAMP. This development board works with Arduino and TinyOS platforms.

The netlist created by the user defines the input signal and its pin, the circuit components and which stage and channel this component in on the RAMP, the programmable parameter for the component chosen and finally define the output signal pin. Once the user programmed

the application's netlist, a back-end software checks the netlist for syntax error and perform placing and routing routine. The place routine optimizes the placement of the selected components for best electronic performance. The routing routine optimizes the physical connection between the selected components by setting switching circuitry (connection box and switch box) to be on/off. This back-end software makes the RAMP a very user friendly platform as the user does not need to worry about the placement and routing of components which reduce any error, also the RAMP provides the optimal synthesised circuit with minimal routing between the components.

6.2 RAMP Version 1.1

Since the RAMP was a success to provide a user-friendly reconfigurable platform to synthesis analog processing circuits, a newer and improved version was developed. The new RAMP, RAMP 1.1, is similar to RAMP 1.0 in that it consists of 10 stages and 8 channels. However, an additional eleventh stage (ninth analog stage) was added. This new stage was named "Unique Stage". As its name implies, the CABs in this stage contains different application-specific components. Unlike the other stages where all the 8 CABs in one stage are identical to enable parallelism. Those 8 unique CABs are summarized in Table 6.2 and explained below:

1. Temperature Compensation CAB: based on the work in [60], to provide temperature compensation for all FG current sources.

2. Subthreshold Neurons CAB: consists of two complete neuron circuits operating in the subthreshold current-mode [61]. These neuron are in the form of nonvolatile analog memory that need the ability to learn. These neurons are the basis for neural-network on the RAMP in the future.

3. A programmable BiQuad filter CAB [62]: utilizing C^4 bandpass filters. This newer filter and can operate as low-pass filter, high-pass filter or band-pass filter.

4. Automatic Gain Control CAB: used as a sensor-interfacing circuitry for audio sensing application (i.e. MEMs microphone input). It consists of 11 programmable biases, 4 OTAs and 3 opamps. Synthesizing this circuit with CAB elements would have exploited a lot of CABs and routing line.

Table 6.2: RAMP 1.1. Unique Stage Circuit

Channel	Circuit
0	Temperature Compensation
1	Subthreshold Neurons
2	BiQuad Filter
3	Automatic Gain Control
4	Bandgap Reference with Temperature Compensation
5	Wheatstone Bridge
6	Ring Oscillator
7	Neural Amplifier

5. Bandgap Reference with Temperature Compensation CAB: similar to channel 4 Bi-Quad filter but with temperature compensation.

6. Wheatstone Bridge CAB: typically to be used in sensor-interfacing circuitry.

7. A current-controlled ring oscillator CAB [63, 64]: that is commonly used as a clock source. It is current controlled by a floating-gate current source.

8. Neural Amplifier CAB: to permit real-time neural recording activity.

6.3 Conclusion

In this chapter, I discussed the RAMP, field-programmable analog array, that utilizes floating-gate transistor to set programmable analog components such as filters. Currently, in my research lab, there are two versions of the RAMP, RAMP 1.0 and RAMP 1.1. The newer version offers improved selection circuitry and infrastructure. The RAMP was design to allow a programming platform similar to the typical FPGA design. Using Matlab or Octave software interfaces, the end-user could easily program any prototype design. The RAMP is designed so that all routing procedures are performed automatically for the user which makes the RAMP a convenient platform for circuit designers. One of the main goals of this research is to explore different methods to program FG memory cell that can be used

on the RAMP. Therefore, plans for the next iteration of the RAMP is to redesign the RAMP system to utilize the below-ground injection of FGs.

Chapter 7

Conclusion and Future Work

FG transistors were first introduced in 1967 and since then, their applications became unlimited. Floating-gate memory cells can be used for both digital and analog applications. One of the analog applications of FGs is non-volatile analog memory where a precise charge is stored on the FG. In this research work, different FG programming techniques were introduced and discussed. These techniques are to be used to program an FG memory cell in FG-dense analog applications.

My research has focused on the development of various programmer circuits for floating-gate arrays that can be utilized in FG-dense analog applications such as Field-Programmable Analog Arrays (FPAAs). Developing FG programmer circuit contributes to the infrastructure-level of larger analog applications. The general approach I have chosen to work on for programming is hot-electron injection. Much of my work focused on developing a compact floating-gate cell that could be used in FG-dense analog applications along with a continuous-time programmer circuit. One conventional above-ground programmer and two novel below-ground injection programmers were presented in this work.

Finally, all proposed research work has been tested and validated with analog applications (i.e field-programmable analog arrays (FPAA) and filter banks) to prove their applicability in providing accurate, repeatable results with the least infrastructure and least possible power consumption.

Table 7.1 presents a comparison of the programming methods presented in this dissertation and other programming techniques. From the table, the programming technique with the highest resolution [32] suffers a longer programming time, this is due the small injection rate used over short injection period to achieve better resolution. The continuous-time

Table 7.1: Performance Comparison

Ref.	Range	Accuracy	Programming Time	Level of Integration
Above-Ground	2.32V	13 bits	100ms	Fully on-chip
Below-Ground	0.7V	11.12 bits	18ms	Fully on-chip
[65]	0.13V	< 8 bits	-	Off-chip
[47]	1.4V	9.45 bits	120ms	Fully on-chip
[32]	4 V	13.4 bits	50ms $\times$ 3.3 $\times$ 10^4	Fully on-chip
[13]	0.63V	9.5 bits	50ms	Fully on-chip
[66]	1V	6.5 bits	75 μs	Fully on-chip
[41]	0.3V	9 bits	-	Only I-V on-chip
[67]	2V	10 bits	50ms	Off-chip

above-ground programming method presented here provided a resolution of 13 bits with injection time of 100ms while the below-ground method shows a resolution of 11.12 bits with injection time of 18ms. The injection time for the work presented in this dissertation varies depending on injection condition. A faster injection time could be achieved by using higher V_{sd} or bigger I_s current during injection.

7.1 Contribution

The goal of the work presented in this dissertation was to develop an FG programming technique that is compact, fast, accurate and suitable for low-power analog systems. Additionally, this programming technique should be easily scaled to work with an array of FGs. I achieved this goal by:

- evaluating and testing the original and traditional above-ground programming technique used in the Reconfigurable Mixed Signal Platform (RAMP) [2].
- designing and testing a novel below-ground FG programming technique that uses negative voltages to program an FG array. This below-ground circuit is similar to the original above-ground programming. This circuit made below-ground injection possible for field-programmable analog arrays (FPAAs).
- developing, designing, fabricating and testing a newer version of below-ground FG

programming. This scheme employs a compact FG cell with a compact programmer circuit and provides more control on the injection process.

- testing the newer below-ground programming circuit with an analog application. The FG was programmed and used as programmable current source to tune the corner frequencies of a C^4 bandpass filter bank.

- designing a prototype programmer circuit with a high-gain two-stage opamp used as the negative feedback amplifier for the programmer circuit. This prototype programmer circuit was designed to work with both above and below ground injection and is expected to provide improved performance than the previous programmer circuits.

7.2 Future Work

Similar to all other research work, there are elements in this work that need to be taken further. My future plans focus on three directions: 1) developing a below-ground programmer circuit with folded-cascode opamp to program an array of FGs, 2) testing and characterization of the new programmer with a simple application, i.e. C^4, 3) scaling the below-ground programming of the FGs to work with a larger array of FGs and adopting below-ground injection to program FGs on next iteration of the RAMP system.

The folded-cascode opamp designed and analyzed in Chapter 5 provides high gain and good input-common range higher than V_{dd}. However, biasing circuitry need to be designed and integrated with the opamp circuit. Additionally, and most importantly, the proposed folded-cascode opamp is to be integrated with the below-ground programmer to inject an array of the FGs. Finally, testing for repeatability and characterizations need to take place to determine proper programming and operation for the programmer circuit with an FG cell.

Secondly, once the folded-cascode opamp is integrated into the programmer circuit, the newer programmer needs to be tested with a proof-of-concept analog application. This is significant to prove the effectiveness of the programmer circuit before scaling the FG array to work with larger analog applications such as the RAMP.

Finally, even though the work done in Chapter 4 showed the effectiveness of below-ground programming with a programmable analog application circuit, it was just for a proof-of-concept application. Therefore, the work that deserves the most attention and raises the biggest challenge is to develop system-level below-ground programming that could be

successfully scaled to work on the next iteration of the RAMP chip. It is expected that using this technique will provide more consistent results among all the FGs on the RAMP.

References

[1] M. M. Navidi and D. W. Graham, "A regulated charge pump for injecting floating-gate transistors," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2017, pp. 1–4.

[2] B. Rumberg, D. W. Graham, S. Clites, B. M. Kelly, M. M. Navidi, A. Dilello, and V. Kulathumani, "Ramp: accelerating wireless sensor hardware design with a reconfigurable analog/mixed-signal platform," in *Proceedings of the International Conference on Information Processing in Sensor Networks*. ACM, 2015, pp. 47–58.

[3] B. Rumberg and D. Graham, "A low-power and high-precision programmable analog filter bank," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 59, no. 4, pp. 234–238, April 2012.

[4] M. M. Navidi, "Integrated circuits for programming flash memories in portable applications," Ph.D. dissertation, West Virginia University, 2018.

[5] M. M. Navidi, D. W. Graham, and B. Rumberg, "Below-ground injection of floating-gate transistors for programmable analog circuits," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2017, pp. 1–4.

[6] S. Ravindran, P. Smith, D. Graham, V. Duangudom, D. Anderson, and P. Hasler, "Towards low-power on-chip auditory processing." *EURASIP J. Adv. Sig. Proc.*, vol. 2005, pp. 1082–1092, 01 2005.

[7] G. Frantz, "Digital signal processor trends," *IEEE Micro*, vol. 20, no. 6, pp. 52–59, 2000.

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

[9] R. Sarpeshkar, "Efficient precise computation with noisy components: extrapolating from an electronic cochlea to the brain," 1997.

[10] P. Pavan, R. Bez, P. Olivo, and E. Zanoni, "Flash memory cells-an overview," *in Proceedings of the IEEE*, vol. 85, no. 8, pp. 1248–1271, Aug., 1997.

[11] C. G. O. P. Micheloni, Rino, *Memories in Wireless Systems*. Springer, 2008.

[12] Micron Semi., *NAND Flash 101: An Introduction to NAND Flash and How to Design It In to Your Next Product*. https://www.micron.com/resource-details/fea5cfd9-ee93-47f4-b2af-cd494d3291c3, 206.

[13] A. Basu and P. E. Hasler, "A fully integrated architecture for fast and accurate programming of floating gates over six decades of current," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, pp. 953–962, 2011.

[14] 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.

[15] B. K. Ahuja, Hoa Vu, C. A. Laber, and W. H. Owen, "A very high precision 500-na cmos floating-gate analog voltage reference," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 12, pp. 2364–2372, 2005.

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

[17] S. Lai, "Flash memories: where we were and where we are going," in *International Electron Devices Meeting 1998. Technical Digest (Cat. No.98CH36217)*, 1998, pp. 971–973.

[18] ——, "Non-volatile memory technologies: The quest for ever lower cost," in *2008 IEEE International Electron Devices Meeting*, 2008, pp. 1–6.

[19] A. Benvenuti, A. Ghetti, A. Mauri, H. Liu, and C. Mouli, "Current status and future prospects of non-volatile memory modeling," in *2014 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, 2014, pp. 5–8.

[20] P.Hasler, B. Minch, and C. Diorio, "Floating-gate devices: they are not just for digital memories any more," in *IEEE International Symposium on Circuits and Systems*, vol. 2, Jul 1999, pp. 388–391 vol.2.

[21] C. Mead, *Analog VLSI and Neural Systems*, ser. Addison-Wesley VLSI system series. Addison-Wesley, 1989.

[22] T. Shibata and T. Ohmi, "A functional mos transistor featuring gate-level weighted sum and threshold operations," *IEEE Transactions on Electron Device*, vol. 39, no. 6, pp. 1444–1455, Jun 1992.

[23] M. Holler, S. Tam, H. Castro, and R. Benson, "An electrically trainable artificial neural network (etann) with 10240 'floating gate' synapses," in *International Joint Conference on Neural Networks*, 1989, pp. 191–196 vol.2.

[24] A. Thomsen and M. A. Brooke, "A floating-gate mosfet with tunneling injector fabricated using a standard double-polysilicon cmos process," *IEEE Electron Device Letters*, vol. 12, no. 3, pp. 111–113, March 1991.

[25] S. Shah and S. Collins, "A temperature independent trimmable current source," in *IEEE International Symposium on Circuits and Systems*, vol. 1, 2002, pp. I–713–I–716 vol.1.

[26] S. Jackson, J. Killens, and B. Blalock, "A programmable current mirror for analog trimming using single poly floating-gate devices in standard cmos technology," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 1, pp. 100–102, Jan 2001.

[27] Y. Wong, M. Cohen, and P. Abshire, "A 750-mhz 6-b adaptive floating-gate quantizer in 0.35um cmos," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 7, pp. 1301–1312, July 2009.

[28] T. Constandinou, J. Georgiou, and C. Toumazou, "An auto-input-offset removing floating gate pseudo-differential transconductor," in *Proceedings of the International Symposium on Circuits and Systems*, vol. 1, May 2003, pp. I–169–I–172 vol.1.

[29] 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.

[30] Y. L. Wong, M. Cohen, and P. Abshire, "On-line histogram equalization for Flash adc," in *Proceedings of the International Symposium on Circuits and Systems*, May, 2007, pp. 3598–3601.

[31] Y. L. Wong, M. H. Cohen, and P. A. Abshire, "A 1.2 GHz adaptive floating gate comparator with 13-bit resolution," in *Proceedings of the International Symposium on Circuits and Systems*, May, 2005, pp. 6146–6149 Vol. 6.

[32] C. Huang, P. Sarkar, and S. Chakrabartty, "Rail-to-rail, linear hot-electron injection programming of floating-gate voltage bias generators at 13-bit resolution," *IEEE Journal of Solid-State Circuits*, vol. 46, pp. 2685–2692, 2011.

[33] 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.

[34] R. R. Harrison, J. A. Bragg, P. Hasler, B. A. Minch, and S. P. 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, 2001.

[35] L. Carley, "Trimming analog circuits using floating-gate analog mos memory," *IEEE Journal of Solid-State Circuits*, vol. 24, no. 6, pp. 1569–1575, Dec 1989.

[36] A. Negut and A. Manolescu, "Analog floating gate approach for programmable current mirrors and current sources," in *International Semiconductor Conference*, vol. 02, Oct 2010, pp. 525–528.

[37] P. Hasler, B. A. Minch, and C. Diorio, "An autozeroing floating-gate amplifier," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 1, pp. 74–82, 2001.

[38] 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.

[39] M. Lenzlinger and E. Snow, "Fowler-Nordheim tunneling into thermally grown SiO_2," *IEEE Transactions on Electron Devices*, vol. 15, no. 9, p. 686, Sept., 1968.

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

[41] A. Bandyopadhyay, G. Serrano, and P. Hasler, "Adaptive algorithm using hot-electron injection for programming analog computational memory elements within 0.2 percent of accuracy over 3.5 decades," *IEEE Journal of Solid-State Circuits*, vol. 41, pp. 2107–2114, 2006.

[42] T. Ong, P. Ko, and C. Hu, "Hot-carrier current modeling and device degradation in surface-channel p-mosfets," *IEEE Transactions on Electron Devices*, vol. 37, no. 7, pp. 1658–1666, July 1990.

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

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

[45] C. Diorio, S. Mahajan, P. Hasler, B. Minch, and C. Mead, "A high-resolution nonvolatile analog memory cell," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 3, Seattle, WA, April 1995, pp. 2233–2236.

[46] K.-H. Kim, K. Lee, T.-S. Jung, and K.-D. Suh, "An 8-bit-resolution, 360-μs write time nonvolatile analog memory based on differentially balanced constant-tunneling-current scheme (DBCS)," *IEEE J. Solid-State Circuits*, vol. 33, no. 11, pp. 1758–1762, Nov. 1998.

[47] Y.-D. Wu, K.-C. Cheng, C.-C. Lu, and H. Chen, "Embedded analog nonvolatile memory with bidirectional and linear programmability," *IEEE Trans. Circuits Syst. II*, vol. 59, no. 2, pp. 88–92, Feb. 2012.

[48] A. Sedra and K. Smith, "A second-generation current conveyor and its applications," *IEEE Transactions on Circuit Theory*, vol. 17, no. 1, pp. 132–134, February 1970.

[49] B. Rumberg and D. Graham, "A low-power field-programmable analog array for wireless sensing," in *International Symposium on Quality Electronic Design*, March 2015, pp. 542–546.

[50] S. L. Clites, "A parallel Programmer for Non-Volatile Analog Memory Arrays," Master's thesis, West Virginia University, 2015.

[51] D. 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.

[52] A. Dilello, B. Rumberg, and D. Graham, "Multiplexing high-side load switch using adaptive well biasing," *Electronics Letters*, vol. 52, no. 12, pp. 1056–1058, June 2016.

[53] M. Navidi and D. Graham, "A regulated charge pump with extremely low output ripple," *Electronics*, vol. 8, p. 1293, 11 2019.

[54] P. Allen and D. Holberg, *CMOS Analog Circuit Design.* Oxford University Press, Inc., 2002.

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

[56] R. Ellis, H. Yoo, D. Graham, P. Hasler, and D. Anderson, "A continuous-time speech enhancement front-end for microphone inputs." in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 2, May 2002, pp. 728–731.

[57] E. K. F. Lee and P. G. Gulak, "A CMOS field-programmable analog array," *IEEE Journal of Solid-State Circuits*, vol. 26, no. 12, pp. 1860 –1867, Dec. 1991.

[58] 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, no. 9, pp. 1781–1794, Sept. 2010.

[59] T. Hall, C. Twigg, J. Gray, P. Hasler, and D. Anderson, "Large-scale field-programmable analog arrays for analog signal processing," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 52, no. 11, pp. 2298–2307, nov 2005.

[60] 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)*, May 2017, pp. 1–4.

[61] S. Liu, J. Kramer, G. Indiveri, T. Delbrück, R. Douglas, and C. A. Mead, *Analysis and Synthesis of Static Translinear Circuits.* MITP, 2002, pp. 177–227. [Online]. Available: https://ieeexplore.ieee.org/document/6290146

[62] 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.

[63] 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.

[64] 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.

[65] S. Chakrabartty and G. Cauwenberghs, "Fixed-current method for programming large floating-gate arrays," in *IEEE International Symposium on Circuits and Systems*, vol. 4, May, 2005, pp. 3934–3937.

[66] S. Kinoshita, T. Morie, M. Nagata, and A. Iwata, "A pwm analog memory programming circuit for floating-gate mosfets with 75-/spl mu/s programming time and 11-bit updating resolution," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 8, pp. 1286–1290, 2001.

[67] Y. Wong, M. Cohen, and P. Abshire, "A floating-gate comparator with automatic offset adaptation for 10-bit data conversion," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 52, pp. 1316 – 1326, 08 2005.

[68] Changsik Yoo, "A cmos buffer without short-circuit power consumption," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, no. 9, pp. 935–937, 2000.

Appendix A

$0.5\mu m$ Tapeout

During my work on this dissertation, I worked on other smaller testing cells that would be part of the newer RAMP system. Here is I present the circuits I designed, layedout and fabricated in $0.5\mu m$ standard CMOS process available through MOSIS.

A.1 FG Testing Cells

Two FG testing cells, were designed. The difference between the two cells is the size of the capacitor used at the gate. A schematic for the FG cells is shown in Fig. A.1.

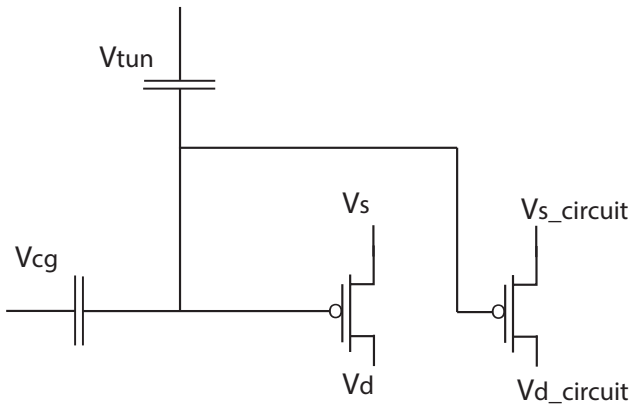

Figure A.1: FG transistor testing cell

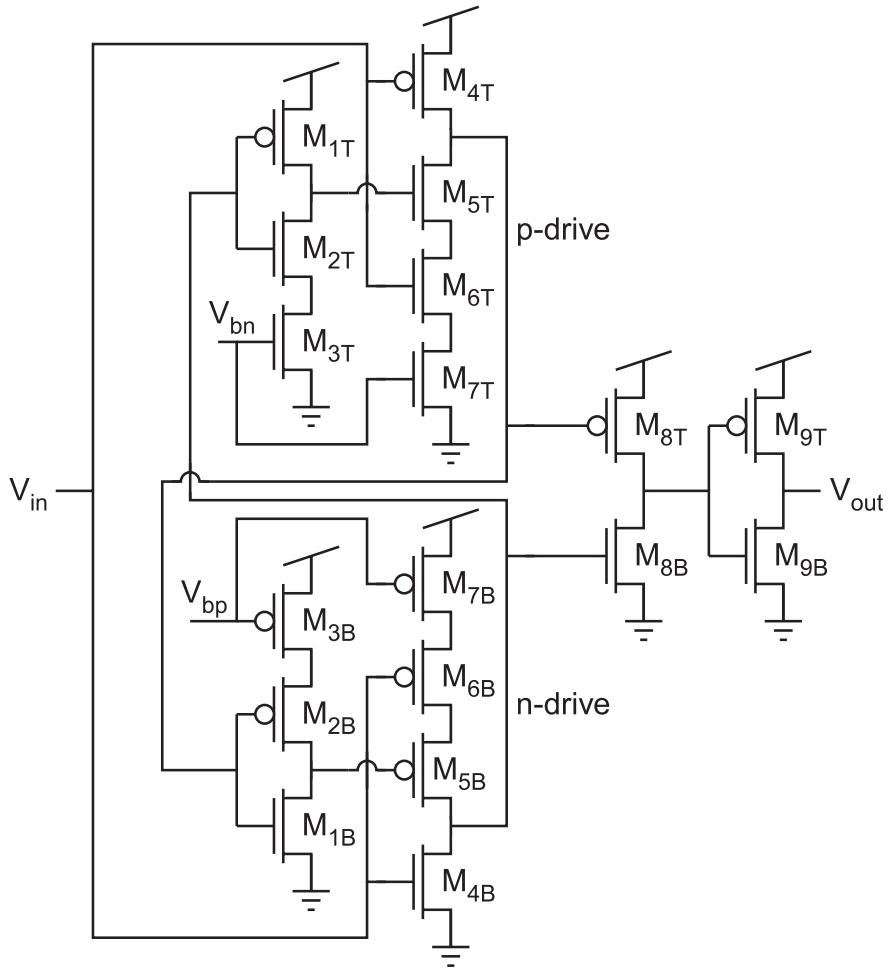

Figure A.2: Edgifier 1

A.2 Edgifier

Four different versions for the edgifier circuit were designed. Starting form the original edgifier circuit, based on [68], used on the charge pump to more simplified version using that utilizes diode-connected transistor instead of current starved one. The two of the designed edgifiers are shown in Fig. A.2 and Fig.A.3.

A.3 Tgate Testing Cells

Tgate is used on the RAMP chip as switching circuitry to select between different analog components. The circuit shown in Fig. A.4 is a tgate circuit utilizing a FG transistor to set the control signal of the tgate.

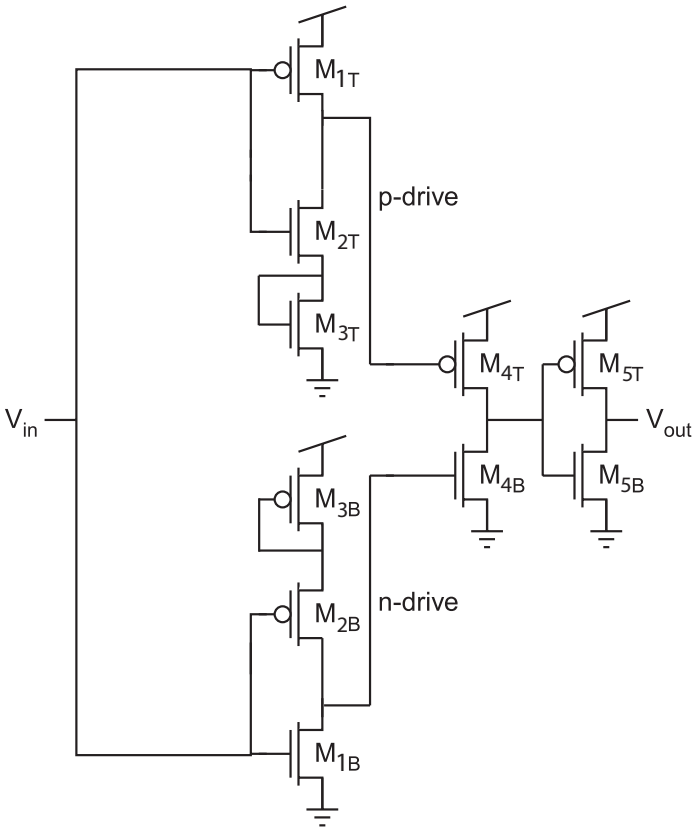

Figure A.3: Edgifier 2

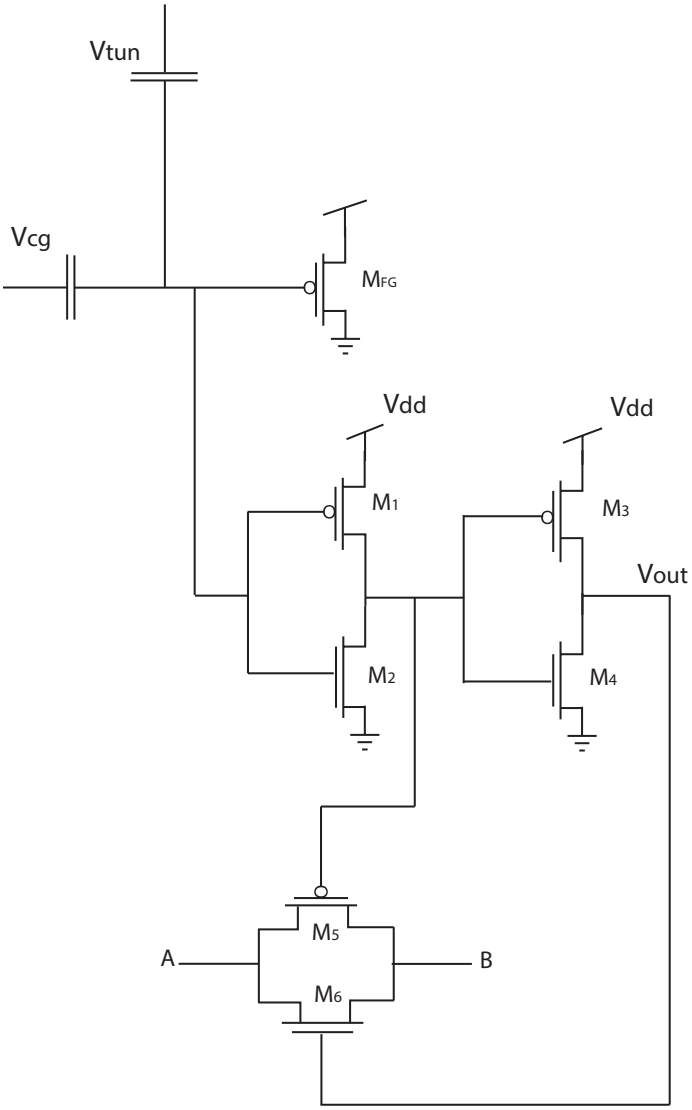

Figure A.4: Tgate Testing cell