APPLICATION PERFORMANCE OF ELEMENTS IN A FLOATING–GATE FPAA

Tyson S. Hall, Christopher M. Twigg, Paul Hasler, and David V. Anderson

Georgia Institute of Technology, Atlanta, GA 30332-0250, tyson@ece.gatech.edu

ABSTRACT

Field–programmable analog arrays (FPAAs) provide a method for rapidly prototyping analog systems. Currently available commercial and academic FPAAs are typically based on operational amplifiers (or other similar analog primitives) with only a few computational elements per chip. While their specific architectures vary, their small sizes and often restrictive interconnect designs leave current FPAAs limited in functionality, flexibility, and usefulness. In this paper, we explore the use of floating–gate devices as the core programmable element in a signal processing FPAA. A generic FPAA architecture is presented that offers increased functionality and flexibility in realizing analog systems. In addition, the computational analog elements are shown to be widely and accurately programmable while remaining small in area.

1. LOW–POWER SIGNAL PROCESSING

The future of FPAAs lie in their ability to speed the implementation of advanced, low–power signal processing systems. Growing demand for complex information processing on portable devices has motivated a lot of contemporary research in the design of power efficient signal processing systems. For analog systems to be desirable to the largely digital signal processing commmunity, they need to provide a significant advantage in terms of size and power and yet still remain relatively easy to use and integrate into a larger digital system.

Gene's law postulates that the power consumption in DSP microprocessors, as measured in mW/MIPS, is halved about every 18 months. These advances largely follow Moore's law, and they are achieved by using decreased feature size and other refinements, such as intelligent clock gating. Myriad applications only dreamed of a few years ago are possible because of these gains, and they have increased the demand for more advanced signal processing systems. What is needed now is a leap forward in power efficiency. Unfortunately, a problem looms on the horizon: the power consumption of the analog–to–digital converter (ADC) does not follow Gene's law and will soon dominate the total power budget of digital systems. While ADC resolution has been increasing at roughly 1.5 bits every five years, the power performance has remained the same, and soon, physical limits will further slow progress.
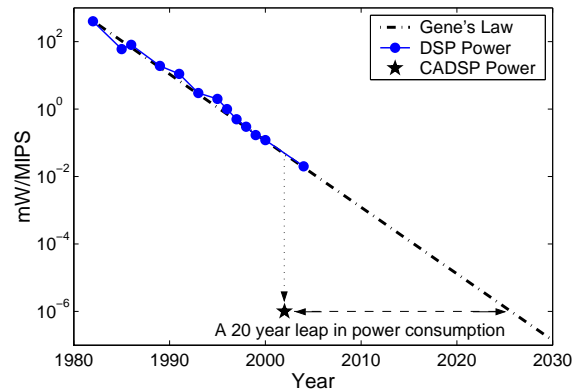

Fig. 1. Data from [1] showing the power consumption trends in DSP microprocessors along with data taken from a recent analog, floating–gate integrated chip developed by the CADSP team [2, 3, 4].

Most current signal processing systems that generate digital output place the ADC as close to the analog input signal as possible to take advantage of the computational flexibility available in digital processors. However, the development of large–scale FPAAs and the CAD tools needed for their ease of use would allow engineers the option of performing some of the computations in reconfigurable analog hardware prior to the ADC. This results in both a simpler ADC and a substantially reduced computational load on the digital processors that follow. Furthermore, the analog processor and ADC may be combined to form a specialized ADC tailored to the application at hand.

Recent advances in analog floating–gate technologies have shown it to be a viable alternative to traditional FPAA designs [5]. As shown in Figure 1, analog floating–gate circuits have shown tremendous gains in efficiency (a factor of as much as 10,000) compared with custom digital approaches for the same applications, and when used in the ADC, they result in more efficient biasing.

2. COMPUTATIONAL ANALOG BLOCKS

The computational logic is organized in a compact computational analog block (CAB) providing a naturally scalable architecture. CABs are tiled across the chip in a regular mesh–type architecture with busses and local interconnects in–between as shown in Figure 2. A sample FPAA with 64
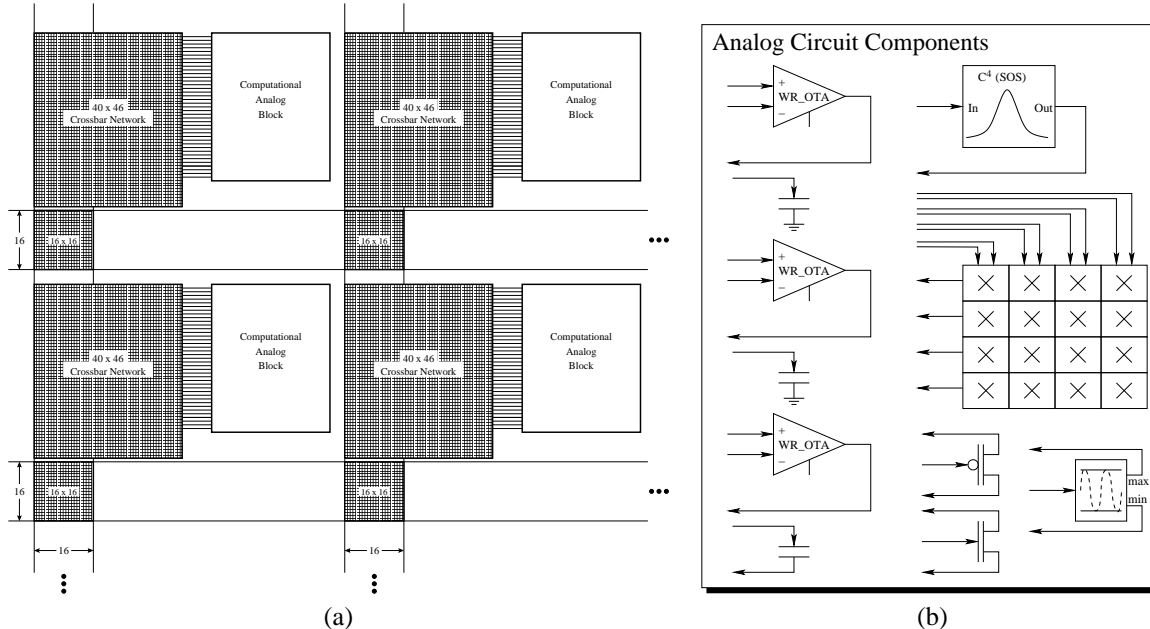
Fig. 2. (a) This is the overall block diagram for a large–scale FPAA. The switching interconnects are fully connectable crossbar networks built using floating–gate transistors. (b) This is a Computational Analog Block (CAB) for an FPAA based on floating–gate devices. Here, each CAB contains a four-by-four matrix multiplier, three wide–range operational transconductance amplifiers (OTAs), three fixed–value capacitors, a cascade of two capcatively coupled current conveyors (C^4), a peak detector, and two FET transistors. The component signals shown in this figure are routed to the rows of the switch matrix.

CABs on a single chip fabricated in TSMC 0.35-micron is estimated to cover an area of approximately 36 mm^2. Of course, commercially viable FPAAs are forseen that have 100s if not 1000s of CABs based on this same architecture.

The programmable elements in the FPAA are floating–gate transistors, and they are used for both the switch interconnects and the computational logic. In the switch network, floating–gate switches provide similar characteristics as a standard pass transistor without the need of a digital memory cell. In the computational logic, floating–gate transistors can be accurately programmed and used to set bias currents and store coefficients. This scheme allows a single programming infrastructure to be used since all programmable elements will be floating–gate transistors [6].

Many example CABs can be imagined using this technology. Figure 2 shows one example CAB, whose functionality is enhanced by a mixture of medium– and coarse–grained computational blocks similar to many modern FPGA designs. The computational blocks were carefully selected to provide a sufficiently flexible, generic architecture while optimizing certain frequently used signal processing blocks. For generality, operational transconductance amplifiers (OTAs), FETs, and fixed–value capacitors are included [7]. For specialization, a cascade of capacitively coupled current conveyors for band–pass filtering, 4 x 4 vector–matrix multiplier, and peak detector are added. These blocks will optimize operations such as sub-banding, fourier processing, and matrix transformations.

3. SYSTEM RESULTS

The testbed floating–gate FPAA was fabricated in a 0.5-micron, standard CMOS process. This FPAA contains two CABs with a floating–gate crossbar switch network connecting them [5]. The CAB design was slightly smaller than the one outlined in Section 2 having a C^4 bandpass filter module, 4 x 4 vector–matrix multiplier, and three wide–range OTAs. This design, however, is more than sufficient to test the concept of floating–gate FPAAs and characterize the elements of the CAB.

The resistance of the floating–gate switch in the "on" position proved similar to standard pFET switches with the measured on–resistance starting at 11 kΩ. As shown in Figure 3a, the architecture of our FPAA required that the resistance be measured through two devices in series. During these experiments, both of the transistors were programmed to the same position and the measured resistances shown here have been divided by two to report the resistance for a single transistor. The resistance of the floating–gate switch in the "off" position was not directly measured due to the extremely small currents. Instead, the saturation current of the device was determined as shown in Figure 3b. This data was acquired with $V_{DD} = 7.5\ V$ since this was the first reliable measurement voltage. The resistance can then be calculated to be $R = \frac{U_T}{I_{SAT}} = \frac{25.8\ mV}{145.4\ pA} = 177.5\ M\Omega$. At an operating $V_{DD} = 3.3\ V$, the off–resistance is higher. At this level, the worst case bias current is 70 pA, which
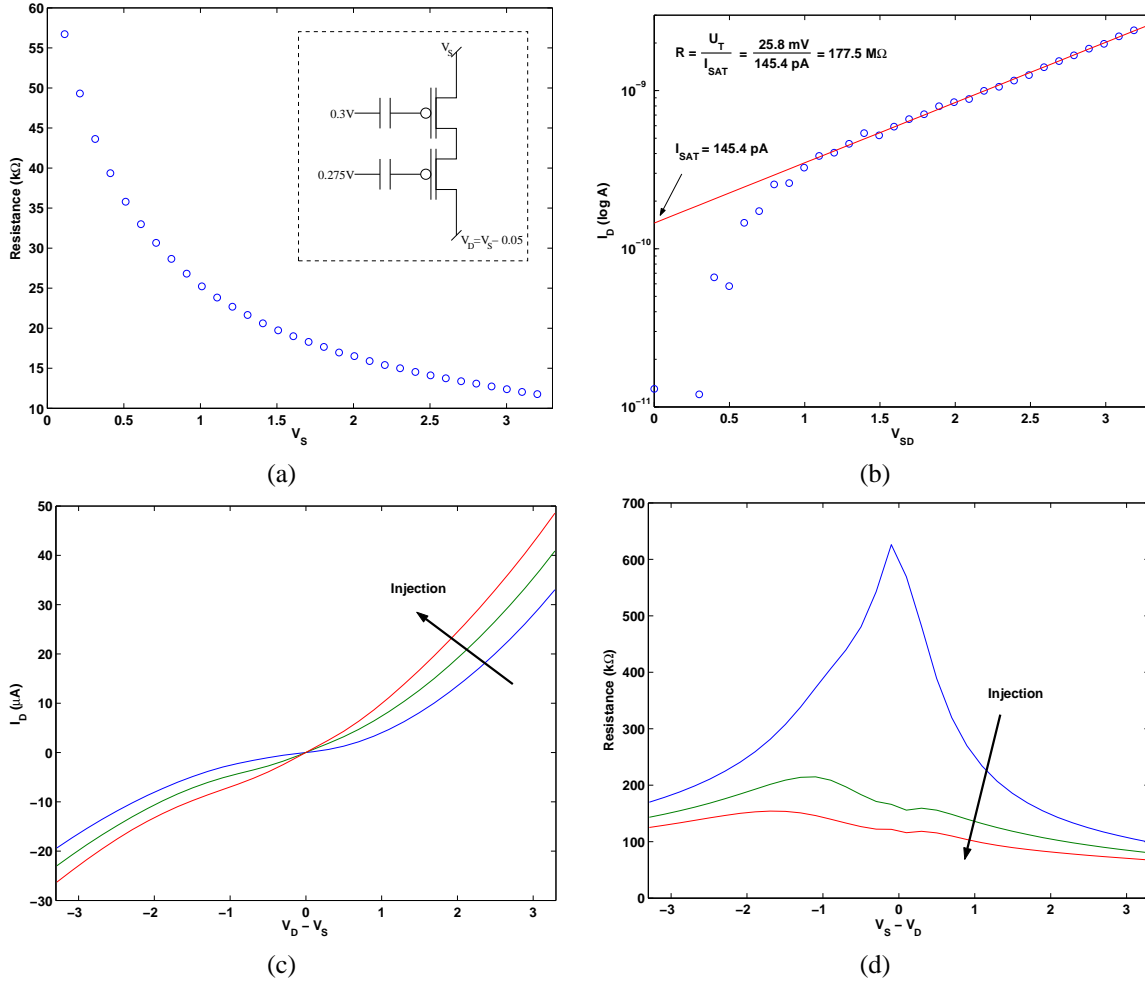
Fig. 3. **Switch Characteristics of Floating–gate Transistors:** (a) Resistance for an "on" switch was measured using two floating–gate devices in series, both programmed to the "on" position. Here, the experimental resistance has already been divided by two to represent a single floating–gate switch. (b) Currents for an "off" switch are too low for reliable picoammeter measurements. So, a conservative "off" resistance was determined by measuring the saturation current of a floating–gate transistor programmed to the "off" position while $V_{DD} = 7.5V$. (c) The floating–gate switch can also be programmed mid–position (between "on" and "off") to synthesize a variable resistance. Here, a sampling of the differential currents achievable with this programming scheme are shown. (d) The resistance plots correspond to the same injection levels as the differential currents shown in (c).

equates to an "off" resistance in the 1 GΩ range.

Figures 3c and d show the current and resistance of a switch as it is programmed mid–way between the "on" and "off" positions. As can be seen by the current measurements, these devices have been injected close to the "off" position. While this resistance is non-linear over the full operational range, given a sufficient constraint on the input, linearity can be achieved.

Using the programming method described in [6], the FPAA's floating–gate switches can be accurately programmed. Figure 4 shows a switch that was programmed to an arbitrary current of 8 nA at an operating gate voltage of 0.3 V. Thus, floating–gate switches can be used to accurately set a current level within the system (e.g., these devices can be used to implement a current source as well).

The OTAs in the Computational Analog Blocks (CABs) have biases that are set with a floating–gate current source. In Figure 5, a source–follower integrator is shown. Using the switch matrix, a single OTA from one of the CABs is configured and connected to the external pins via five floating–gate switches. Once configured, the biasing floating–gate transistor is programmed to vary the corner frequency of this first–order filter. The frequency response is shown for several programmed corner frequencies in Figure 5. The moderate gain in the lower frequencies is due to the switches that are in the feedback loop of the OTA. Ideally, the output node and the negative input node would be directly connected. However, this path must be routed via the switch network in the FPAA, which means that a minimum of two floating–gate switches will be in the feedback
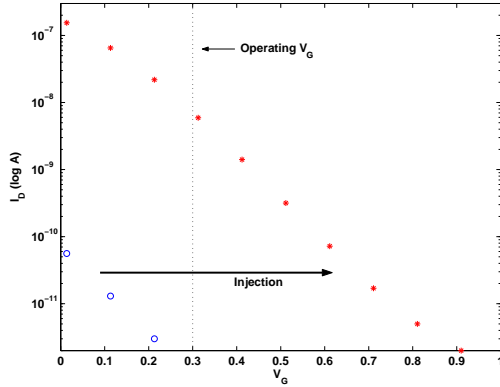
Fig. 4. This is a floating–gate switch programmed to 8 nA at an operating gate voltage of 0.3 V. Thus, current sources can be accurately defined within the switching network itself.

loop. The gain can be minimized by injecting the floating–gates of these switches to a lower charge, or if gain is desired for a given application, then it can be set by programming these switches to a higher charge.

One can imagine a wide class of systems that can be implemented and configured on this FPAA. In particular, differentiators, cascaded second–order sections, bandpass filters, matrix transforms, and frequency decomposition and processing are all well suited for this architecture. In addition, the systems implemented here used transistors that were all biased in the subthreshold region lending further support to floating–gate devices being used in ultra low–power design.

4. CONCLUSION

FPAAs based on floating–gate technologies are an emerging design concept that will increase the current state–of–the–art in the analog and mixed–signal prototyping. In particular, these FPAAs are well suited to facilitate the design of low–power signal processing systems based on analog floating–gate devices. In this paper, a novel FPAA architecture was presented that utilizes floating–gate transistors as programmable switches, in–circuit active elements, and the configurable device within the computational analog blocks. Several systems were implemented on a testbed FPAA, and they were shown to have a high degree of programmability via on–chip, floating–gate current sources that set the bias currents. Thus, the bandwidth and Q parameters of various OTA filter designs can be easily modified on–chip without the need for external biases.

5. REFERENCES

[1] G. Franz, "Digital signal processor trends," *IEEE Micro*, vol. 20, no. 6, pp. 52–59, Nov–Dec 2000.

(a)   (b)

(c)
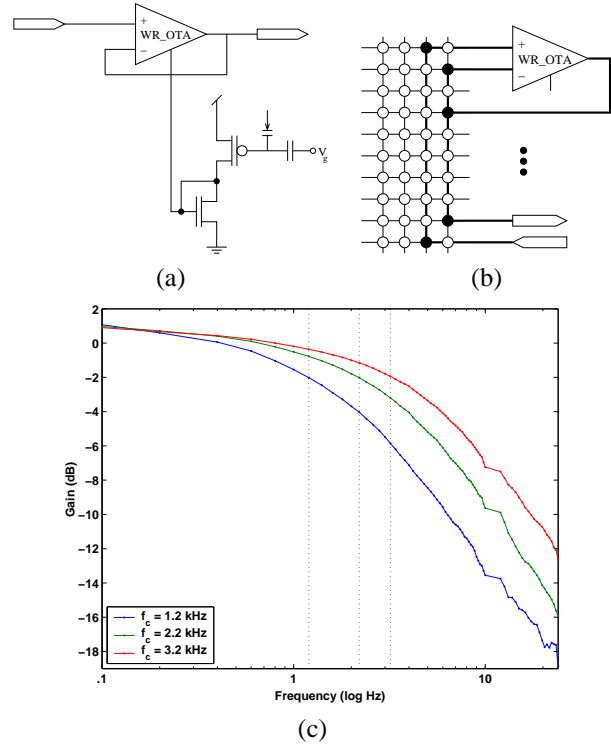
Fig. 5. (a) The source–follower configured using a floating–gate current source. By programming the floating gate charge, the current is set in the current mirror (the other half of the current mirror is internal to the wide–range OTA). Thus, the effective conductance can be modified for each of the OTAs on chip. (b) Using the switch matrix, an OTA located in one of the CABs is connected in a source–follower configuration, and two external pins are routed to the OTA as the input and output signals. The programmable biases illustrated in (a) are not shown here for simplicity, but each OTA has a current mirror and floating–gate current source that sets its bias. (c) The frequency response of the source–follower circuit is shown for several bias currents. An internal floating–gate transistor is used as a current source to set the OTA's bias. Injecting the floating–gate device, increases the current and thus the bandwidth of this first order filter.

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

[3] P. Hasler, P. Smith, R. Ellis, D. Graham, and D. V. Anderson, "Biologically inspired auditory sensing system interfaces on a chip," in *2002 IEEE Sensors Conference*, Orlando, FL, June 2002, Invited Paper.

[4] P. D. Smith, M. Kucic, R. Ellis, P. Hasler, and D. V. Anderson, "Mel–frequency cepstrum encoding in analog floating-gate circuitry," in *Proceedings of the International Symposium on Circuits and Systems*, Phoenix, AZ, May 2002, pp. IV–671 – IV–674.

[5] T. S. Hall, P. Hasler, and D. V. Anderson, "Field–programmable analog arrays: A floating–gate approach," in *Proc. 12th International Conference on Field Programmable Logic and Applications*, Montpellier, France, Sept. 2002, pp. 424–433.

[6] G. Serrano, P. Smith, H. J. Lo, R. Chawla, T. Hall, C. Twigg, and P. Hasler, "Automatic rapid programming of large arrays of floating–gate elements," in *ISCAS*, 2004.

[7] T. S. Hall, C. M. Twigg, P. Hasler, and D. V. Anderson, "Developing large–scale field-programmable analog arrays," in *Reconfigurable Architectures Workshop*, 2004.