# Indirect Programming of Floating-Gate Transistors

David W. Graham, Ethan Farquhar, Brian Degnan, Christal Gordon, and Paul Hasler
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
Email: phasler@ece.gatech.edu

***Abstract*— Floating-gate transistors are useful for precisely programming a large array of current sources. Present floating-gate programming techniques require disconnection of the transistor from the rest of its circuit to be programmed. We present a new method of programming floating-gate transistors indirectly that does not require this disconnection. Two transistors share a floating gate allowing one to exist directly in a circuit while the other is reserved for programming. Since the transistor does not need to be disconnected from the circuit to program it, the switch count is reduced, resulting in fewer parasitics and better overall performance.**

Floating-gate (FG) transistors have been shown to be very useful acting as precise current sources when directly programmed with a combination of hot-electron injection and Fowler-Nordheim tunnelling [1,2,3,4]. Programming these FGs has previously required using a T-gate switch to disconnect the transistor from its circuit for a programming phase and then reconnecting it for a run-time phase [5]. However, the addition of a T-gate switch for every FG to be programmed can be costly. The process of disconnection can decrease the maximum speed of operation and overall accuracy while also increasing the required real estate and necessary supply overhead. To circumvent the problems associated with detaching the FG transistor, we introduce a new, non-invasive method of programming that eliminates the need for disconnection and instead uses an indirect method of programming.

The concept of indirect programming of floating-gate transistors is illustrated in Fig. 1 (a-b). With this indirect programming technique, multiple MOSFETs share a common floating gate. One pFET is connected to the programming structure while the source and drain of the other FET are connected to the respective circuit. The first pFET is programmed in the fashion of [6] using hot-electron injection and tunnelling. Since the charge on this "programmer" pFET is modified, the current of the other transistor (the "agent") will also be set.

We present techniques for programming accurate currents with indirect programming and have fabricated several circuits for verification. All data presented in this paper were obtained from $0.5\mu m$ processeses available through MOSIS using a $V_{dd}$ of $3.3V$.

## I. Motivation for Indirect Programming

To illustrate the usefulness of this indirect programming method, Fig. 2 (a) shows the FG current mirror introduced in [7] for perfectly matching the two leg currents. The full schematic of this current mirror is actually given by Fig. 2 (b), and the increase in complexity is clearly evident. The additional resistances and capacitances introduced by the eight T-gates, used to break the FG transistors out of the mirror for programming, seriously hamper the performance of the current mirror, especially at high frequencies. The simple two-transistor current mirror becomes a complex 18-transistor circuit.

The use of indirectly programmed FG transistors simplifies the pFET current mirror to that of Fig. 2 (c). Now, only a minimal amount

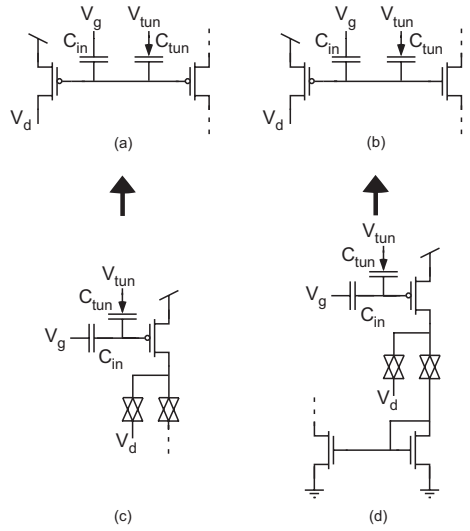


Fig. 1. (a) Programming structure of a pFET indirectly programming another pFET. The left transistor is connected to the external programming structure and is actively programmed. The transistor on the right is connected to its circuit (shown by the dotted lines) and is passively programmed. (b) Programming structure of a pFET indirectly programming an nFET. (c) Direct method of programming a pFET. Direct programming requires disconnecting the pFET from the rest of the circuit with T-gates. This schematic represents a best-case scenario in which only two T-gates are required. For some applications, two T-gates each at the source and gate would also be required. (d) Direct method of programming an nFET. Direct programming requires programming the current in a pFET and then mirroring that current into the nFET that is connected to the circuit. In all cases, $V_{tun}$ is set to a constant DC voltage in run mode equal to the voltage used when measuring the current.

of disconnects need to be included. Only two cascoding transistors and a single T-gate are now used, and the cascoding transistors serve the dual purpose of isolating the FG transistor and enhancing the current response of the mirror.

Precise programming of nFETs with hot-electron injection is virtually impossible due to process-control techniques that specifically work to avoid nFET injection [8]. When an nFET is to be used as a precise current source with FGs, a pFET is programmed, and that current is mirrored into the nFET current source, as shown in Fig. 1 (d). Therefore, creating a programmable nFET current mirror with the direct method of programming is no simple task.

The process of programming an nFET is more explicit with indirect programming. Since an nFET and pFET can share the same floating gate, the nFET current is set by programming the pFET. This technique allows the construction of an nFET current mirror that is completely analogous to the pFET version of Fig. 2 (c).

Therefore, this new indirect method of programming has several distinct advantages over previous methods, and these advantages are

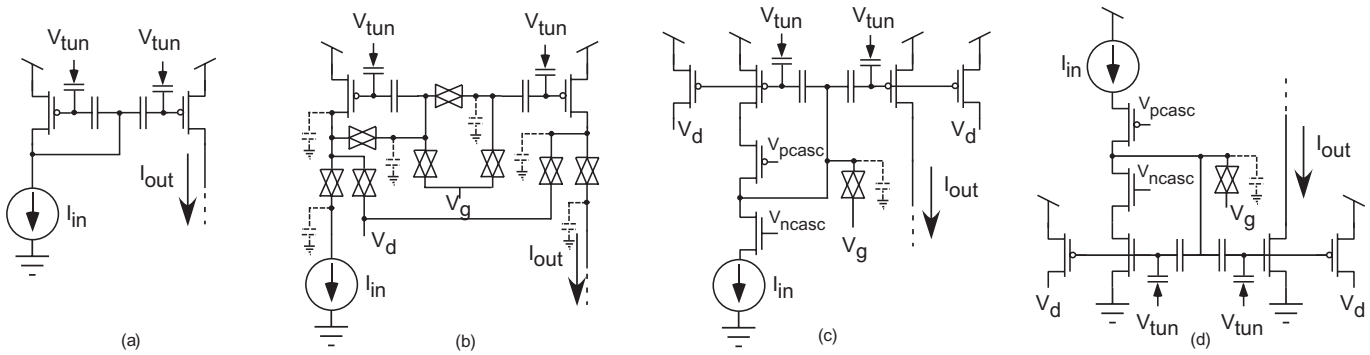


Fig. 2. (a) Floating-gate transistors for offset removal in a current mirror. (b) Implementation of the FG current mirror using direct FG programming techniques. To allow complete disconnection of each FG transistor for programming, many T-gate switches must be used which add parasitic capacitances (shown in dashed lines) and resistances. These switches increase the required area and supply headroom while concurrently degrading the operational performance. (c) Implementation of the FG current mirror with the indirect-programming technique. The use of indirectly programmed transistors greatly reduces the complexity of the circuitry and minimizes the parasitics. The two cascode transistors are included for both improved performance and also for isolation of the gate voltage for programming. (d) Implementation of an nFET FG current mirror with indirect programming. This current mirror is a simple design (the same as the pFET version), whereas the construction of an nFET programmable current mirror is virtually impossible.

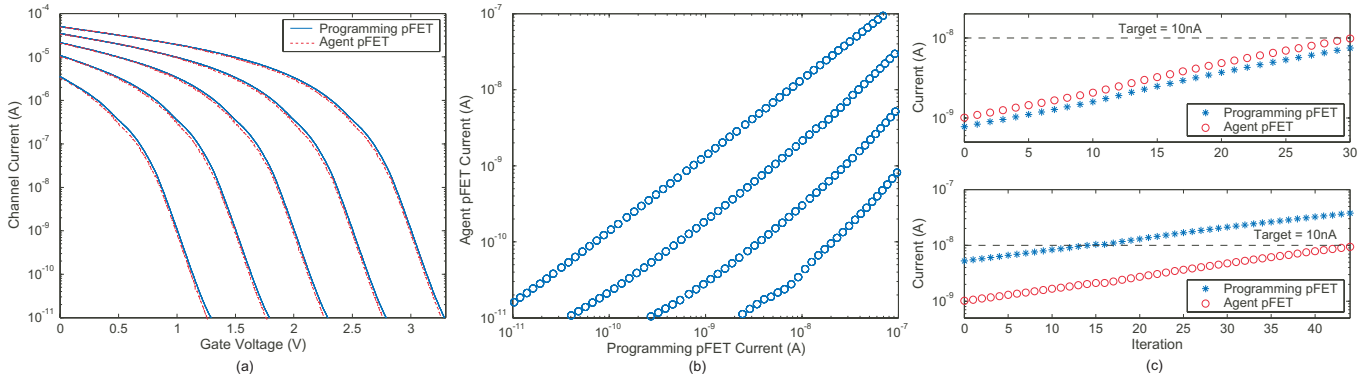


Fig. 3. (a) I-V characteristics of an indirectly programmed pFET ($\frac{W}{L} = 2$) and its programming pFET ($\frac{W}{L} = 2$). (b) Ratio of the programming pFET to the operational pFET for various values of $V_s$. The slope of each trace begins to differ from unity at low current levels due to measurement limitations. At high current level, the slope differs since the programming pFET leaves subthreshold sooner than the operational pFET for smaller values of $V_s$. (c) Programming the operational pFET to a target. (Top) Programming when the sources are at similar potentials. (Bottom) Programming when the operational pFET has a higher source potential than the programming pFET.

summarized as follows.

- Allows nFET programming
- Decreases the number of poles / parasitic capacitances for faster operational speeds
- Decreases resistance
- Decreases minimum supply headroom
- Permits run-time time programming / calibration
- Reduces transistor count / real estate

## II. Indirect Programming of pFET Transistors

The most basic method of indirect programming uses injection in the programming pFET to set the current in the agent pFET (Fig. 1 (a)). The programming pFET can be connected in a large FG array similar to [5]. The output of the agent will be a scaled version of the programmer, assuming the drain and source potentials of the two devices are similar. Scaling is due to $\frac{W}{L}$ ratios and any mismatch between the two devices. Figure 3 (a) shows the I-V characteristics for a gate sweep of two identically sized devices ($\frac{W}{L} = 2$).

Assuming that the sources and drains of the two transistors are at similar potentials is not always valid. Figure 3 (b) shows the effects of varying the source potential of the agent. With both transistors in the subthreshold regime, varying the programmer current yields approximately a $1:1$ change in the agent current.

Measuring the programmer current is used to predict the agent current, with the relationship shown in Fig. 3 (b). Figure 3 (c) shows that this technique can be used to accurately set the current in the agent within tolerance for two different values of the agent's $V_s$. Only the current through the programmer is observable during programming.

## III. Indirect Programming of nFET Transistors

As was stated previously, an important advantage of indirect programming is that it provides a simple mechanism for programming an nFET (Fig. 1 (b)), whereas process-control parameters make direct nFET programming difficult. However, certain design issues must be considered since the programming pFET and the agent nFET share a common FG.

Figure 4 (a) shows the I-V characteristics of both the nFET and pFET. If the transistors are not properly sized, then these curves will be significantly skewed. Unlike the pFET-pFET case, a direct relationship between the two transistors is not easily obtained. When

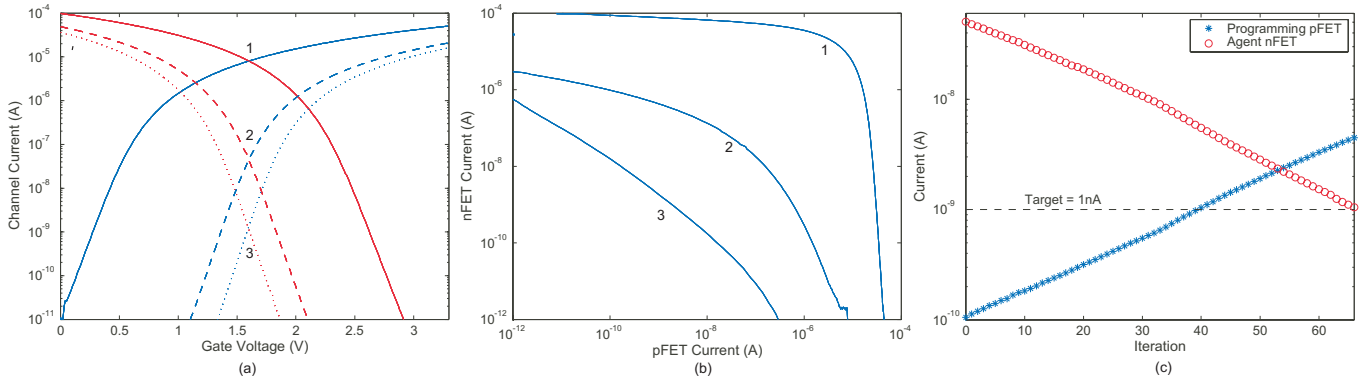


Fig. 4. (a) I-V characteristics of an indirectly programmed nFET ($\frac{W}{L} = 2$) and its programming pFET ($\frac{W}{L} = 2$). Curves 1-3 show the I-V relationships attained by increasing ($V_s - gnd$) for the nFET and ($V_{well} - V_s$) for the pFET. (b) Current-to-current relationships for each of the three curves shown in (a). As the current crossover point moves down, the current-to-current relationship becomes more linear, simplifying the programming process. (c) Programming the operational nFET to a target.

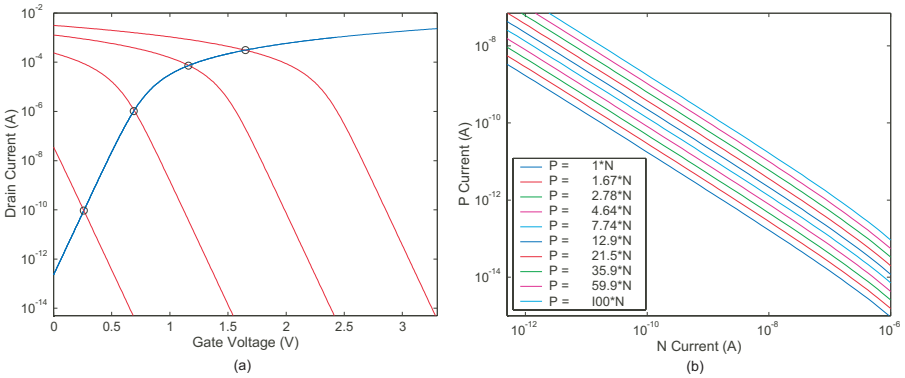


Fig. 5. (a) Simulation data of nFET and pFET I-V curves. The pFET's $V_{well}$ is being lowered, shifting the curve to the left. When $V_{well}$ is low enough, both transistors will operate in the subthreshold regime simultaneously. (b) Simulated current-to-current curves. In each of these cases both transistors are operating in subthreshold. $P \equiv \frac{W_P}{L_P}$ and $N \equiv \frac{W_N}{L_N}$. Changing the $\frac{W}{L}$ ratio does not change the linearity but increases the amount of current available.

the two transistors are not in subthreshold simultaneously, a current-to-current relationship like that in curve 1 of Fig. 4 (b) is the result. Small changes in pFET current yield large changes in nFET current. Therefore, restricting the operation to strictly subthreshold is desirable because it linearizes the current-to-current ratio.

Two methods are available to ensure that both transistors are simultaneously in subthreshold. The first requires moving the sources of both transistors. Decreasing the pFET source (reference to $V_{well}$) and increasing the nFET source (referenced to $V_{bulk}$) reduces the current in each transistor. This moves the threshold voltages to a point in which it is possible to operate both transistors in subthreshold at the same time (Fig. 4 (a)). Figure 4 (b) relates the pFET-to-nFET current for each set of curves in Fig. 4 (a). Lowering the crossoever point increases the linear range of the current-to-current ratio.

A linear current-to-current relationship makes predicting the agent current trivial. However, any reasonable current-to-current relationship (like curve 2 Fig. 4 (b)) allows accurate programming of the nFET. Fig. 4 (c) shows that the nFET can, indeed, be programmed to a desired value when only observing the pFET current.

As the sources of the transistors may not always be accessible, the previous method may not be possible. The second method of ensuring that both transistors are in subthreshold requires that the programming pFETs be in a well isolated from the operational circuit and that those wells can be accessed.

The current crossover point is a characteristic of process parameters and the $\frac{W}{L}$ ratios. Using parameters derived from a commercially available $0.5\mu m$ process, simulations determined that a $\frac{W_P}{L_P}$ ratio of the pFET to a $5\frac{W_N}{L_N}$ ratio of the nFET causes the current crossover point to be in the middle of the $V_{DD}$ rail. To make the crossover point occur in the subthreshold region for both transistors, the source and well potentials of the pFET are lowered to the threshold voltage of the nFET during the measurement phase of programming [6]. This has the effect of shifting the pFET curve to the left in Fig. 5 (a).

The gate voltage of both transistors is limited by $V_{gate} \leq V_{well}$, and $V_{well} \leq V_{Tn}$ (the threshold voltage of the nFET), thus ensuring the two transistors will always be in subthreshold. This makes programming the nFET a simple transform from the pFET (Fig. 5 (b)). The $\frac{W}{L}$ ratio does not change the linearity of the curve if the voltages are restricted to subthreshold voltage levels but simply alters the amount of current (shifting the curves).

## IV. Drain Characteristics with Indirect Programming

As has been shown previously, the difference between source potentials of the programming pFET and the agent transistor need to be taken into account when programming so that the correct current flows through the agent. The drain potentials of the two transistors are also of concern, especially the drain of the agent since the operation of its connected circuit can affect the potential at the drain. The drain of the programming pFET is held constant when not programming, thus eliminating all transient coupling effects from it.

The voltage on any FG node is set by a combination of the FG charge and a sum of the inputs to the gate through capacitive dividers [9]. The extension of the the FG voltage for the indirect programming

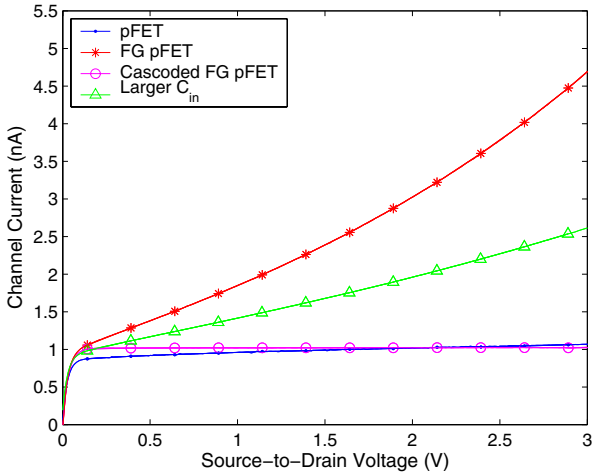


Fig. 6. Transistor drain sweeps. Due to capacitive coupling through $C_{gs}$, $I_{sat}$ in the FG pFET increases exponentially for larger $V_{ds}$ values. Increasing $C_{in}$ reduces the exponential current increase. Cascoding the agent transistor eliminates the exponential current increase and flattens $I_{sat}$ more than the $I_{sat}$ of the identically sized standard pFET.

case is described by

$$\begin{aligned} V_{FG} &= \frac{Q_{FG}}{C_T} + \frac{C_{in}}{C_T}V_g + \frac{C_{tun}}{C_T}V_{tun} \\ &+ \frac{C_{gdp}}{C_T}V_{dp} + \frac{C_{gsp}}{C_T}V_{sp} + \frac{C_{oxp}}{C_T}\psi_p \\ &+ \frac{C_{gda}}{C_T}V_{da} + \frac{C_{gsa}}{C_T}V_{sa} + \frac{C_{oxa}}{C_T}\psi_a \end{aligned} \quad (1)$$

where $C_T$ is the total capacitance connected to the FG node, the $p$ and $a$ subscripts indicate the programmer and the agent, respectively, and the $\psi$ represents the surface potential of each transistor (constant $\psi$ in subthreshold). Since $C_{gda}$ is a small parasitic capacitance, the drain of the transistor acts as an input to the gate. As the drain voltage is swept, a subthreshold current through the device changes exponentially, as is shown in Fig. 6. This is a significant alteration from the small slope due to the Early voltage of an identically sized transistor, which is also shown.

If supply headroom issues are important, then the drain-coupling effect can be minimized by increasing the input gate capacitance. Increasing $C_{in}$ increases $C_T$, thereby reducing the effect of $\frac{C_{gda}}{C_T}$. While the saturation current still has an exponential increase with drain potential, the range over which the current changes becomes smaller, as is shown in Fig. 6. Increasing the input capacitance further will decrease the exponential change even further.

If supply headroom issues are not a concern, then this drain-coupling effect can be completely removed by adding a cascode transistor at the drain of the agent. The saturation current received by the circuit is flatter than even a standard transistor, as is shown in Fig. 6.

## V. Conclusion

Direct programming of floating gates has been proven to be a highly accurate tool for analog designers. Difficulties with programming nFETs and the parasitics associated with the isolation circuitry exist with the direct programming method but can be overcome by the indirect programming method we have just introduced.

An early, ad hoc method of indirectly programming floating gate transistors has been shown to be a useful means of tuning a circuit [10]. Additionally, in this paper we have presented a systemmatic approach to programming both pFETs and nFETs indirectly. This systemmatic approach can easily be extended to large arrays of FG devices so that a large number of current sources can be programmed without invasively disconnecting them. In fact, directly and indirectly programmed FG transistors can coexist in the same large array so that each might be used for its particular advantage.

Indirect programming also allows certain circuits to be transformed into a programmable version that would not have been previously possible. The aforementioned programmable nFET current mirror is now possible, and a neuron circuit [11] that cannot properly operate due to the parasitics and voltage drops of the isolation circuitry can now be made.

New possibilities with floating-gate programming also exist. Since the agent transistor is never removed from its circuit, indirect programming removes the necessity of a separate programming phase and a operational phase. This allows the possibility of run-time recalibration and adaption to be carried out by the programming pFET. A circuit which uses a similar concept is described in [12].

Indirect programming offers solutions to many of the problems of direct programming while also providing new and unique capabilities to augment the analog designer's toolbox.

## References

[1] P. Hasler, C. Diorio, B. Minch, and C. Mead, "Single transistor learning synapses," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. Touretzky, and T. Leen, Eds. Cambridge, MA: MIT Press, 1995, pp. 817–824.

[2] W. P. Millard, Z. Kalayjian, and A. G. Andreou, "Calibration and matching of floating gate devices," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. IV, Geneva, Switzerland, June 2000, pp. 121–124.

[3] A. J. Montalvo and J. J. Paulos, "Improved floating-gate devices using standard CMOS technology," *IEEE Electron Device Letters*, vol. 14, no. 8, pp. 372–374, 1993.

[4] J. Killens, "Utilizing standard CMOS process floating gate devices for analog design," Master's thesis, Mississippi State University, 2001.

[5] P. Smith, M. Kucic, and P. Hasler, "Accurate programming of analog floating-gate arrays," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 5, Scottsdale, AZ, May 2002, pp. 489–492.

[6] G. Serrano, P. D. Smith, H. J. Lo, R. Chawla, T. S. Hall, C. M. Twigg, and P. Hasler, "Automatic rapid programming of large arrays of floating-gate elements," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 1, Vancouver, BC, Canada, May 2004, pp. 1373–1376.

[7] F. Adil, G. Serrano, and P. Hasler, "Offset removal using floating-gate circuits for mixed-signal systems," in *Proceedings of the IEEE Southwest Symposium on Mixed-Signal Design*, Las Vegas, NV, February 2003, pp. 190–195.

[8] P. Hasler, A. G. Andreou, C. Diorio, B. A. Minch, and C. A. Mead, "Impact ionization and hot-electron injection derived consistently from boltzmann transport," *VLSI Design*, vol. 8, no. 1-4, pp. 455–461, 1998.

[9] K. Yang and A. G. Andreou, "Subthreshold analysis of floating-gate MOSFET's," in *Proceedings of the Tenth Biennial University/Government/Industry Microelectronics Symposium*, Research Triangle Park, NC, May 1993, pp. 141–144.

[10] A. Low and P. Hasler, "Basics of floating-gate low-dropout voltage regulators," in *Proceedings of the IEEE Midwest Symposium on Circuits and Systems*, vol. 3, 2000, pp. 1048–1051.

[11] E. Farquhar and P. Hasler, "A bio-physcially inspired silicon neuron," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 1, Vancouver, BC, Canada, May 2003, pp. I–309 – I–312.

[12] J. Dugger and P. Hasler, "Supervised learning in a two-input analog floating-gate node," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 5, Vancouver, BC, Canada, May 2004, pp. 756–759.