Reconfiguration Costs in Analog Sensor Interfaces for Wireless Sensing Applications

Brandon Rumberg and David W. Graham
Lane Department of Computer Science and Electrical Engineering
West Virginia University, Morgantown, WV 26506
Email: brumberg@mix.wvu.edu, david.graham@mail.wvu.edu

***Abstract*—Analog sensor interfaces are used in wireless sensor nodes to perform sensor conditioning, event detection, and data reduction. The use of reconfigurable interfaces will enable applications developers to customize these sensor interfaces and to reconfigure them in the field. In this paper, we examine the energy cost of reconfiguring analog circuitry and how the requirements of wireless sensor nodes impact the design of reconfigurable analog systems.**

I. Introduction

One of the primary challenges in wireless sensing is to maximize the network's knowledge of its environment while using only the energy available within that environment. Among the many components of this challenge, including energy harvesting and network protocols, is the analog sensor interface. Since all environmental information passes through the sensor interface, the interface irrecoverably limits the bandwidth and precision of the information that can be collected with a given amount of energy. However, the sensor interface presents an opportunity to extract the necessary information early in the signal chain (e.g., by performing event detection to wake a sleeping microcontroller), thereby reducing the energy consumption of the sensor node while still obtaining the desired knowledge of the environment [1]. Thus, an application-optimized interface is crucial for maximizing the knowledge that can be collected with the available energy.

An analog sensor interface should include some amount of reconfigurability to enable application developers to customize the interface for their applications and to redefine the interface after deployment. This need can be met with field-programmable analog arrays (FPAAs), which have been applied to filtering, sensor interfacing [2], and large-scale signal processing [3]. Figure 1 shows the usage of a generic FPAA within a wireless sensor node. The optimal design and usage of FPAAs for wireless sensing depends upon the cost of reconfiguration, which raises tradeoffs such as the universality and power consumption of the FPAA. General FPAA design choices were previously outlined in [4]. This paper studies the circuit-level and node-level costs of FPAA reconfiguration and what these costs imply for using FPAAs within wireless sensing, where energy consumption is of paramount importance. To aid this study, we have fabricated two 1280-switch FPAAs in 0.35μm CMOS—one with volatile switches and one with nonvolatile switches—and measured the energy while reconfiguring with an off-the-shelf sensor platform. Throughout this paper, we assume a 3V supply voltage as is typical in battery-operated sensor nodes.

This material is based upon work supported by the National Science Foundation under Award No. 1148815.

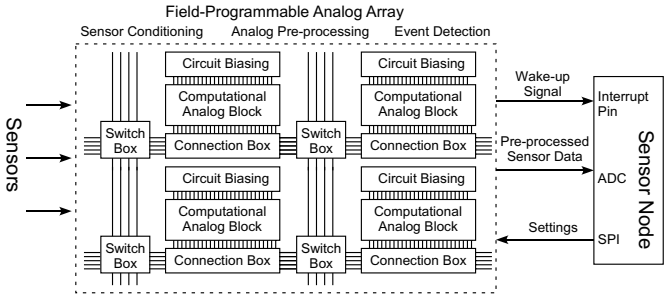

Fig. 1. A field-programmable analog array (FPAA) used as a reconfigurable sensor interface in a wireless sensor node. The FPAA may be used for conditioning a variety of sensor outputs, to perform low-power event detection to wake up the rest of the sensor node, or to extract relevant features of the signal to reduce the bandwidth requirements of the data converter and processor.

II. Cost of Analog Reconfiguration

In an FPAA, reconfiguration is achieved via programmable connections in the connection box and the switch box, which are used for local routing and global routing, respectively. Programmable connections can be created using unbuffered conductive switches (such as pass transistors) or buffered switches (such as current mirrors or voltage followers). We will focus on unbuffered switches since they are more versatile and have no run-mode power consumption. Two previously used unbuffered switches that achieve rail-to-rail operation are a transmission gate (T-gate) controlled by an SRAM cell [Fig. 2(a)] and a floating-gate (FG) pass transistor controlled by the nonvolatile charge that is stored on its gate [Fig. 2(b)] [3]. The FG transistor's gate is not constrained to be within the supply rails and so can be programmed with enough overdrive to achieve a low resistance across the range of operation [5]. In both FPAAs, an SPI block selects the column to write, and then data bits are either latched into the SRAM memory cells or are used to select the FGs to program.

A. Equivalent Switch Resistance

The simulated resistance of T-gate and FG switches is shown in Fig. 2(d). For both switch types, the resistance varies with the common-mode voltage due to the body effect. In the T-gate, the nFET is minimum size and the pFET is sized for symmetric drive strength. The pFET in the FG

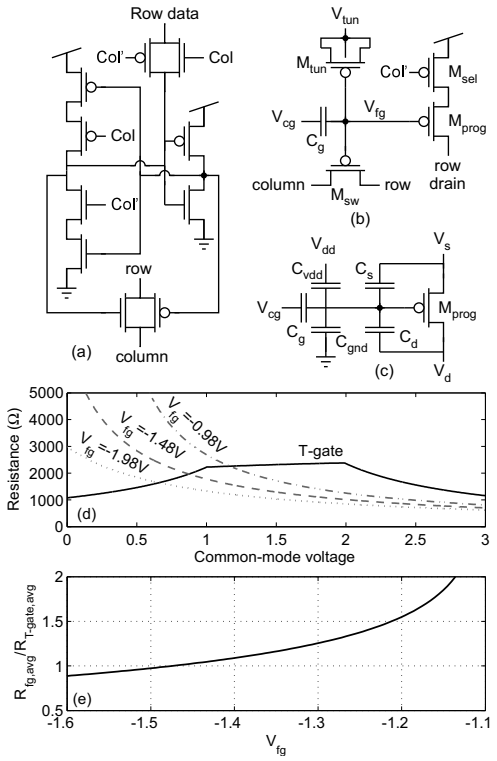

Fig. 2. (a) Example SRAM-controlled transmission-gate switch. *Col* (*Col'*) selects a column of switches to be rewritten. Switch on/off settings are loaded in parallel through *Row data*. *Row* and *column* are the analog routing paths which may be connected through the switch. (b) Example floating-gate transistor switch. C_g=45fF. M_{tun}=0.4μm x 0.4μm. (c) Capacitive coupling of terminals onto the floating gate. (d) Simulated comparison of the resistance of T-gate switches and FG switches. (e) Simulated comparison of the mean resistance of a T-gate switch to the mean resistance of an FG switch.

is the same size as in the T-gate. Note that the kΩ-range switch resistance does not create a significant voltage drop for the nA-range currents that are common in ultra-low-power analog computation systems. Wider switches may be used for resistance-sensitive circuitry; matching resistance between T-gates and FGs is not impacted by changes in dimension when their relative size remains constant.

We define the switches to have equivalent resistance when they have the same average resistance across the supply rails. Under the sizing conditions specified above, the T-gate has an average resistance of 1.8kΩ. The ratio of the FG's mean resistance to the T-gate's mean resistance is shown in Fig. 2(e); $V_{fg} = -1.48$V is required to match the T-gate's average resistance. Therefore, in this paper, we specify the FG voltage for an "on" switch to be $V_{fg,on} = -1.5$V and the voltage for an "off" switch to be $V_{fg,off} = 3$V. However, despite having equal mean resistance, the FG's sharply increasing resistance near ground may be unacceptable for some applications.

B. Erasing a Floating-Gate Switch Matrix

Since the majority of switches in an FPAA switch configuration will be "off," an efficient way to program an FG matrix is to globally erase all switches (by removing electrons from the FGs), and then write only the switches that must be turned on. In the cell in Fig. 2(b), elecrons are removed by raising V_{tun} to a sufficient voltage to cause electrons to tunnel through the thin oxide of M_{tun}. This mechanism is described by the Fowler-Nordheim tunneling equation

$$I_{tun} = \alpha W L e^{\frac{-\beta t_{ox}}{V_{tun} - V_{fg}}} \tag{1}$$

where I_{tun} is the tunneling current through M_{tun}, $\alpha = 688\frac{A}{\mu m^2}$ and $\beta = 34.3\frac{V}{nm}$ are parametric fits, t_{ox} (7.7nm for 0.35μm) is the oxide thickness, and W and L are the width and length of the tunneling junction. Using (1), we determined that $V_{tun} = 12.5$V is sufficient to tunnel all switches to the off state ($V_{fg,off} \approx 3$V) within 100μs. This high-voltage pulse was generated using an on-chip regulated charge pump; measurements of the voltage pulse and supply current are shown in Figs. 3(a)&(b), respectively. The total energy to erase the entire switch matrix is 183nJ. Since the power of the high-voltage generator significantly exceeds the power delivered to the tunneling junction, minimizing the duration of the pulse is crucial to minimizing the energy consumption.

C. Writing a Floating-Gate Switch

After tunneling has been used to remove electrons from all FGs, hot-electron injection is typically used to place electrons onto, and thus "turn on," selected FGs. Injection is commonly used to selectively program standard CMOS FGs because, unlike tunneling, the programming voltages are low enough that standard devices can be used to isolate FGs. Regardless, programming the switches to $V_{fg,on}$ presents a challenge since the FG must be programmed very far below ground.

Injection can be modeled using

$$I_{inj} = \alpha I_s (V_{gd} + V_T) e^{\frac{-\beta}{V_{gd} + V_T}} \tag{2}$$

where $\alpha = 9$ and $\beta = 80$ are parametric fits for 0.35μm, and V_T is the threshold voltage. A large V_{gd} ($\geq$4.5V) is necessary to achieve fast and efficient injection. To accommodate this high voltage, the supply voltage is typically raised during injection. With the drain connected to ground, the FG will need to be at 4.5V. After injecting, the FG must be shifted down 6V to reach $V_{fg,on} = -1.5$V. This FG voltage shift corresponds to a V_{cg} shift of $\frac{C_T}{C_g}$6V, which is approximately 8.5V for typical capacitance values. This number illustrates that raising the supply voltage for injection is an inefficient method to program negative FG values—to maintain safe supply voltages, we will be limited to low V_{gd} at the end of the injection cycle, and thus slow programming. For our estimate of the energy to write a switch, we will assume the use of negative drain voltages.

Using the FG switch capacitive-coupling model shown in Fig. 2(c), we can determine the FG terminal voltages during injection that will correspond to an "on" switch. In run mode, $V_{cg} = 0$V, $V_s = V_d = V_{dd}$, and $V_{fg} = -1.5$V. We can solve for the necessary program-mode FG voltage,

$$V_{fg,p} = \frac{V_{fg,on} + V_{dd}(\frac{C_g}{C_T} - \frac{C_s}{C_T} - \frac{C_d}{C_T}) + \frac{C_s}{C_T} V_{sg,p} - \frac{C_d}{C_T} V_{gd,p}}{1 - \frac{C_s}{C_T} - \frac{C_d}{C_T}} \tag{3}$$

were $V_{sg,p}$ and $V_{gd,p}$ are the program mode source-to-gate and gate-to-drain voltages. If we use $V_{gd} = 5$V and want to

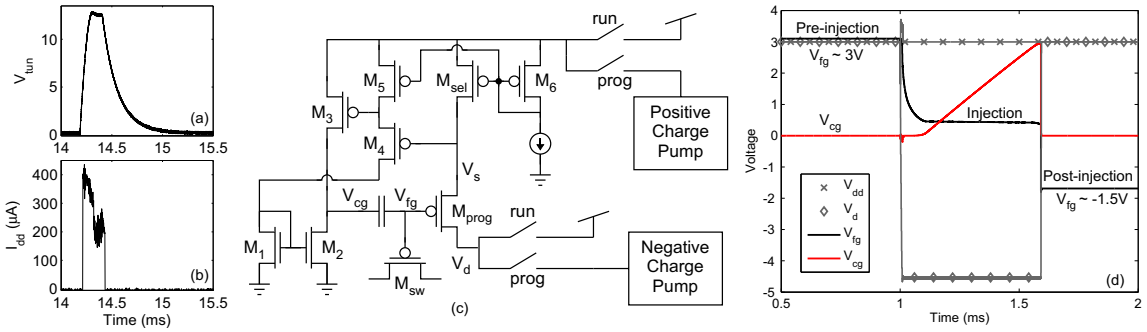

Fig. 3. (a) Measured high-voltage pulse from the on-chip tunneling charge pump. (b) Measured supply current during the high-voltage pulse. The total erase energy was 183nJ. (c) Illustrative injection circuit for "turning on" FG switches. Transistors M_1–M_5 implement negative feedback from V_s to V_{cg}, thus holding V_s and V_{fg} at the desired voltages during injection. (d) Simulation of the illustrative injection circuit. The simulation was performed with device-level implementations of the charge pumps. The total program energy, including the charge pumps' ring oscillators and regulation circuitry, was 152nJ.

program the FG within 1ms, then (2) gives $I_s \approx 7\mu$A (which yields $V_{sg,p} \approx 1$V). Using (3) with $\frac{C_g}{C_T}$=0.7, $\frac{C_d}{C_T}$=$\frac{C_s}{C_T}$=0.02, we obtain the following program-mode approximate terminal voltages: $V_{fg,p} = 0.41$V, $V_{d,p} = -4.59$V, and $V_{s,p} = 1.41$V.

Figure 3(c) shows a complete demonstrative circuit for writing switches using this method. The circuit was designed to hold the FG terminals at the above-mentioned values during injection. A regulated negative charge pump generates the drain voltage. Since the FG is initially "off" at 3V, a positive charge pump is used to generate a short pulse (10μs) on the supply line to start up injection. A full transistor-level simulation for this switch-writing scheme is shown in Fig. 3(d). The total energy to turn on a single switch, including the voltage generation circuits, is 152nJ.

D. Energy Costs of Volatile and Nonvolatile Switches

To determine the reconfiguration cost of an SRAM-based FPAA, we measured the supply current of an FPAA while it was being reconfigured by a sensor platform. Figure 4(a) shows this measurement. The total energy is 80.4nJ, or ≈1nJ/column. This energy is primarily from the SPI block that decodes the incoming serial stream. The FG FPAA will incur the same cost to interpret the serial stream.

In the previous section, we determined the cost of erasing an FG switch matrix and the cost of writing a single switch. The total cost to reconfigure an FG switch matrix depends upon the number of "on" switches. The percentage of switches that are "on" depends upon the complexity of the circuit being synthesized and the design of the FPAA. In our 1280-switch FPAA, we have found that few configurations use more than 5% of the switches. Additionally, although the overhead of generating high injection voltages can generally be amortized through parallel programming, the sparse distribution of "on" switches within a switch matrix confounds the energy reduction of parallel programming. The energy costs for the switches are summarized in Table I. In Section III, we will interpret these results in the context of wireless sensing.

E. Other Considerations Regarding Switches

Density: Although the SRAM cell has more devices in the cell, the FG cell requires a dedicated n-well for M_{tun} (which consumes space), and also requires that C_g is large enough to dominate the capacitance on the floating node. Consequently, our layouts for the cells were the same size (20.4μm x 8.8μm).

Scaling: Charge leakage is a concern for FGs in deeply scaled standard CMOS, particularly when the FG voltages exceed the supply rails. Thick-oxide devices may retain charge longer, but lose the benefits of scaling. Low-leakage SRAM circuits will be needed for deeply scaled SRAM FPAAs.

Reliability: Much more stress is placed upon FGs in switch matrices than in Flash memory or analog circuit trimming. Much more charge is passed through the oxide on each programming iteration and the "on" switches have a high electric voltage across the oxide in run mode (4.5V).

Computation: The tunable conductance of FG switches allows them to be used as computational elements, thus improving the die utilization of FPAAs [6].

Capacitance: Since the T-gate's nFET is much smaller than the pFET, an equivalent pFET FG switch does not have significantly less capacitance than an equivalent T-gate. However, an equivalent nFET-based FG switch would achieve significantly less capacitance. The problem with nFET-based FG switches is that tunneling/injection turns them on/off. So we have to program all of the off switches, which is a larger number than the on switches, and so has high energy cost.

Summary: SRAM FPAAs have clear advantages in terms of reliability, CMOS scaling, and reconfiguration energy (≈123x less than FG FPAAs). However, we will show in the next section that the switch reconfiguration cost is a small part of the system's overall reconfiguration cost, meaning that FG switches are viable when nonvolatility and/or switch computation are beneficial.

III. System-Level Implications of Reconfiguration

To place the FPAA's configuration energy into the system context, we measured the energy of a standard low-power wireless sensing platform (Telos mote [7]) as it received a 1280-bit FPAA configuration over the radio and then programmed the configuration into an SRAM-based FPAA. The results are shown in Fig. 4(b). The energy for the mote to wirelessly receive the configuration was 5.3mJ and the energy for the mote to transfer the configuration serially into the FPAA was 0.331mJ ($E_{ser} = 4.1\mu$J/column). The node-level reconfiguration energy is dominated by the receive energy and

TABLE I
SUMMARY OF RECONFIGURATION COSTS

# Switches	Typ. Sw. Usage	Mote Serial Trans.
1280	5%	E_{ser}=4.1μJ/column
FG Erase	FG Write	SRAM Write
$E_{fg,tun}$=183nJ	$E_{fg,w}$=152nJ/sw	$E_{sram,w}$=1nJ/column
FPAA Type	Total Reconfiguration Energy	
	FPAA-only	FPAA w/ Mote Serial
FG	9.9μJ	276μJ (only prog. "on" columns)
SRAM	80.4nJ	331μJ (prog. all columns)

by the serial transfer energy. The FG FPAA has lower serial costs in Table I since only the "on" switches (typically 5%) require a serial transfer after the global erasure. Similarly, a global reset would reduce the serial cost for an SRAM FPAA.

Since the primary energy cost is wireless reception of the configuration, we developed an entropy-coding algorithm to compress the configuration file [8]. For typical FPAA configurations, we achieve a compression factor of >4. Since many large-scale analog signal-processing systems have identical parallel channels, higher levels of compression are achievable in larger systems. Figure 4(c) shows the measured supply current while receiving the compressed configuration (1.8mJ) and then decompressing the configuration in the mote (34.1μJ). The energy was reduced by approximately 65%

Many wireless sensors are powered by unreliable energy sources, thus volatile FPAAs will incur a cost for restoring configurations. In contrast to SRAM-based switches, FG switches are nonvolatile and therefore do not need to be reprogrammed after a power loss. The lower cost of reprogramming SRAM compared to FGs implies that volatile switches are preferable when the frequency of power outages, f_p, is rare compared to the frequency of fresh reconfigurations, f_r. Assuming that the energy to write SRAM ($E_{sram,w}$) is much smaller than the energy to write a floating-gate ($E_{fg,w}$) and assuming block erasure for both types, SRAM will be lower cost when $\frac{E_{fg,w}}{E_{ser}} > \frac{f_p}{f_r}$. For our implementations, the SRAM FPAA is lower energy when power drop-outs are at least 27-times less likely than fresh reconfigurations. However, a dedicated SPI port should reduce the serial transfer cost to E_{ser}≈120nJ/column [9], which would favor SRAM FPAAs when no more than 1.2 power dropouts occur for each fresh reconfiguration.

It is important to know how much reconfiguration can be done on a given energy budget. Using the SRAM values in Table I, we can project the maximum frequency for reconfiguration. We assume that the system should last 5 years on AA batteries, that only 5% of the system's energy budget is available for reconfiguration, and that compression is not used. For a 1k-switch FPAA, we can receive and program 2.6 configurations per hour. For a 100k-switch FPAA, the allowable frequency of configurations reduces to approximately once every two days, which illustrates the importance of compressing the configuration for large FPAAs. Excluding the communication costs, the allowable frequency of reconfiguration for a 1k-switch FPAA is over two per minute, which is sufficient to allow local adaptation of the FPAA settings.

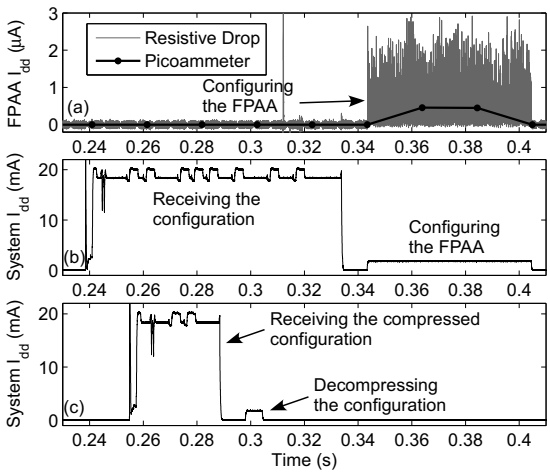

Fig. 4. Measurements of reconfiguration energy. The supply currents were determined by measuring the voltage drop across suitably sized series resistors. (a) Supply current of the SRAM-based FPAA while being reconfigured. (b) Supply current of the sensor node while receiving and writing a configuration to an SRAM-based FPAA. (c) Supply current of the sensor node while receiving and decompressing a compressed FPAA configuration file.

IV. CONCLUSION

In this paper, we have examined the costs of reconfiguring analog circuits in wireless sensors. We have investigated the sources of energy expenditure for both SRAM- and FG-based switch matrices and have introduced a new lower-energy mechanism for programming FG switches. SRAM-based switches are preferable in terms of speed, reconfiguration energy, device reliability, and scaling. However, FG switches offer lower reconfiguration costs when power drop-outs are more frequent than user-scheduled reconfigurations. Either way, our measurements of reconfiguration energy at the system level highlight the importance of optimizing node-to-node and IC-to-IC communications to reduce the total energy. We have shown that a simple method of compressing the FPAA configuration reduces the total energy by 65%.

REFERENCES

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

[2] E. Mackensen and C. Müller, "Implementation of reconfigurable microsensor interfaces utilizing FPAAs," in *IEEE Sensors*, Oct. 2005, pp. 1064–1067.

[3] A. Basu *et al*, "A floating-gate-based field-programmable analog array," *IEEE JSSC*, vol. 45, no. 9, pp. 904–922, Sept. 2010.

[4] D. D'Mello and G. Gulak, "Design approaches to field-programmable analog integrated circuits," *Analog Integrated Circuits and Signal Processing*, no. 17, pp. 7–34, June 1998.

[5] J. Gray, C. Twigg, D. Abramson, and P. Hasler, "Characteristics and programming of floating-gate pFET switches in an FPAA crossbar network," in *IEEE ISCAS*, vol. 1, May 2005, pp. 468–471.

[6] C. Twigg, J. Gray, and P. Hasler, "Programmable floating gate FPAA switches are not dead weight," in *IEEE ISCAS*, 2007, pp. 169–172.

[7] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling ultra-low power wireless research," in *IPSN*, 2005.

[8] B. Kelly, B. Rumberg, D. Graham, and V. Kulathumani, "Reconfigurable analog signal processing for wireless sensor networks," in *IEEE MWSCAS*, 2013.

[9] G. Mathur, P. Desnoyers, P. Chukiu, D. Ganesan, and P. Shenoy, "Ultra-low power data storage for sensor networks," *ACM Trans. Sen. Netw.*, vol. 5, no. 4, pp. 33:1–33:34, Nov. 2009.