Fault-Tolerant Logic Gates Using Neuromorphic CMOS Circuits

Neil Joye, Alexandre Schmid, and Yusuf Leblebici
Microelectronic Systems Laboratory
Swiss Federal Institute of Technology
Lausanne, Switzerland
neil.joye@epfl.ch

Tetsuya Asai and Yoshihito Amemiya
Graduate School of Information Science and Technology
Hokkaido University
Sapporo, Japan

***Abstract*—Fault-tolerant design methods for VLSI circuits, which have traditionally been addressed at system level, will not be adequate for future very-deep submicron CMOS devices where serious degradation of reliability is expected. Therefore, a new design approach has been considered at low level of abstraction in order to implement robustness and fault-tolerance into these devices. Moreover, fault tolerant properties of multi-layer feed-forward artificial neural networks have been demonstrated. Thus, we have implemented this concept at circuit-level, using spiking neurons. Using this approach, the NOT, NAND and NOR Boolean gates have been developed in the AMS 0.35 µm CMOS technology. A very straightforward mapping between the value of a neural weight and one physical parameter of the circuit has also been achieved. Furthermore, the logic gates have been simulated using SPICE corners analysis which emulates manufacturing variations which may cause circuit faults. Using this approach, it can be shown that fault-absorbing neural networks that operate as the desired function can be built.**

I. INTRODUCTION

The ITRS technology roadmap predicts that semiconductor device dimensions will shrink to lower than 30 nm by the end of this decade [1]. While technological feasibility of downscaling the device dimensions is not disputed from the manufacturing point of view, such scaling is expected to result in a multitude of serious challenges at the circuit and system levels, especially in terms of increased leakage currents, reduced power supply voltages, degraded reliability and increased power density. Therefore, actual fault-tolerant design methods for VLSI circuits which have traditionally been addressed at system level, involving algorithmic adaptation, block-level redundancy and majority voting as the main tools, will not be sufficient for future very-deep submicron CMOS devices where serious degradation of reliability is expected. Therefore, new design approaches at low level of abstraction will need to be considered in order to implement robustness and fault-tolerance into these devices.

The adaptability and generalization abilities of artificial neural networks offer a possible solution to this issue, which can be exploited in the fabrication of new fault-absorbent microelectronic circuits [2].

Using this approach, earlier research work has demonstrated that compensation of faults resulting from process variations or device mismatch can be achieved by implementing multi-layer feed-forward artificial neural networks (FFANNs) [3]. Figure 1 depicts a two-layer FFANN with analog inputs and output designed to perform a Boolean operation (NAND, NOR, …). If an appropriate training scheme is applied to it until a defined mean-square error is reached, followed by a second training sequence involving dynamic random suppression of connections, spreading the global information to the second layer units is allowed, thus providing the network with fault-tolerance ability. Figure 2 shows the simulation results of a FFANN composed of nine second-layer neurons, and performing the NAND Boolean function. The mean square error computed at the network output is depicted on the upper graph, where a first training phase consisting of regular error backpropagation followed by a second training phase where one random connection is selected as the faulty one can be identified.

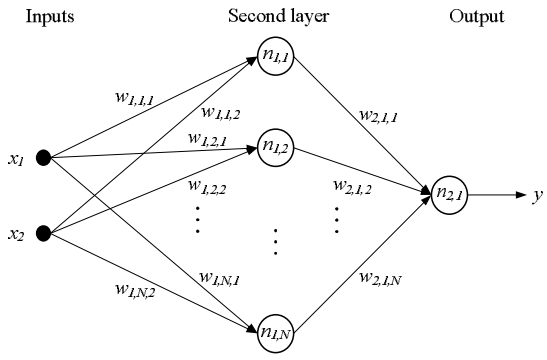

Figure 1. Two-layer FFANN implementing a Boolean function.

1-4244-1001-0/07/$25.00 ©2007 IEEE.

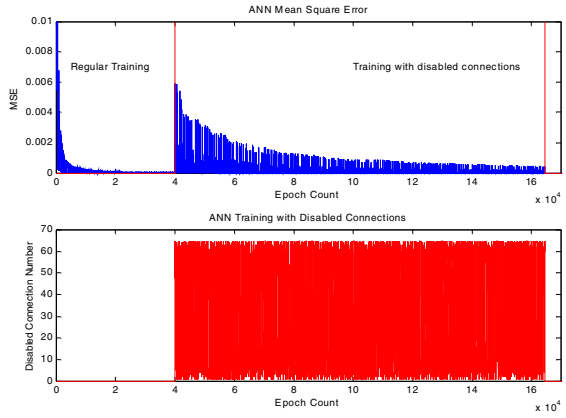

Figure 2. Training results of a NAND Boolean function using a standard error backpropagation learning algorithm.

II. PHYSICAL IMPLEMENTATION

The FFANN concept presented in the previous paragraph has been implemented at hardware circuit-level using spiking neurons. This type of neurons can easily encode the analog signals used by neural networks as mean frequencies of impulses. Moreover, two types of synapses have also been used, namely excitatory and inhibitory synapses.

A. Spiking Neurons

A classical spiking neuron circuit based on a model referred to as the integrate-and-fire neuron has been used. The circuit that has been implemented is taken from [4] and depicted in Figure 3. The input current I_i is integrated on the capacitor C, causing an increase of V_c. When V_c reaches the threshold voltage level of the amplifier, the output voltage V_o switches to Vdd. In this state, the discharge path of the state capacitor C is open, and the charging path of C is closed. Thus, V_c decreases and V_o is switched back to ground potential. This way, a spike is generated and fired at V_o. This specific architecture of the spiking neuron has been selected on the criteria of very compact design.

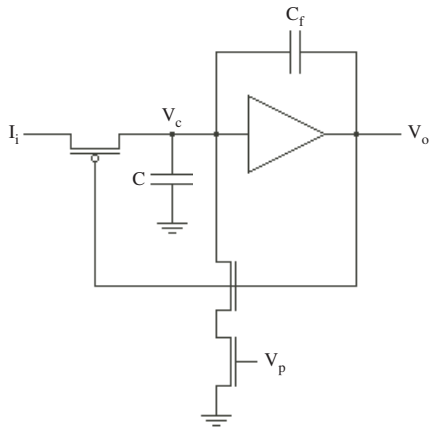

Figure 3. Circuit of the integrate-and-fire neuron.

Spiking neurons can easily encode the analog signals of FFANNs as mean frequencies of impulses. Therefore, the analog signals x are encoded as mean frequencies f, as described in Equation (1). Notice that the value of x ranges between 0 and 1.

$$f = x \ 100\text{MHz} \tag{1}$$

B. Synapses

Two types of synapses have been implemented, namely excitatory and inhibitory synapses (Fig. 4). The goal of these synapses is to control the input current of a spiking neuron. Excitatory synapses are responsible for current increase and inhibitory synapses for current decrease. Furthermore the inputs of these synapses are connected to the outputs of other spiking neurons. The inputs are therefore represented as voltages. Consequently, the synapses have to perform a voltage to current conversion.

Figure 5 depicts the hardware implementation that has been selected for an excitatory synapse having one input signal. It has been adapted from [5] considering one extra input access transistor.

C. Coding of the Synaptic Weight

Consider the circuit described in Figure 5 which implements an excitatory synapse. An important aspect of this circuit is the fact that the size of the input transistor T_{IN}, which controls the value of the current injected in the synapse, also determines the synaptic weight. In order to understand this point, consider the equation of the drain current of a transistor operating in saturation mode.

$$I_D = \frac{\beta}{2n}(V_G - V_{TO} - nVs)^2 \tag{2}$$

$$\beta = \frac{W}{L}\mu C_{ox} \tag{3}$$

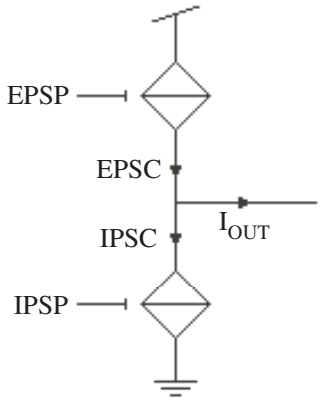

Figure 4. Excitatory and inhibitory synapses. EPSP = Excitatory postsynaptic potential. IPSP = Inhibitory postsynaptic potential. EPSC = Excitatory postsynaptic current. IPSC = Inhibitory postsynaptic current.

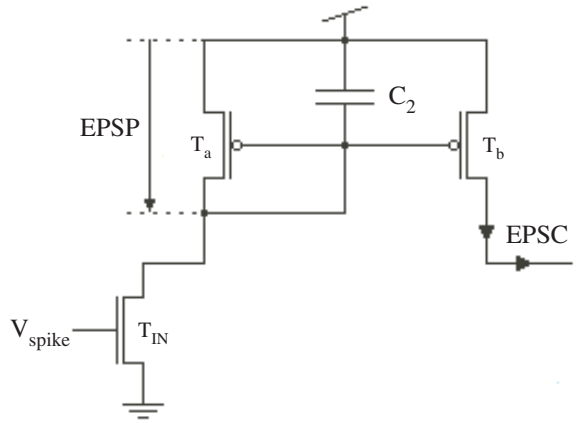

Figure 5. Circuit implementing an excitatory synapse with one input.

As described in Equation (2), the drain current depends on the gate and source voltage and the β factor of the transistor. Considering transistor T_{IN}, the source voltage is at ground potential and the gate voltage is equal to Vdd during an input spike (ground potential when there is not any input pulse). Therefore, the only way to control the drain current of the input transistor, and by the same way the current injected into the synapse, consists of varying the length or the width of T_{IN}. In our implementation, the gate length is used to control I_D in order to ease keeping the current mirror in saturation mode.

The minimum gate length L_{MIN} of the input transistor, which encodes a synaptic weight ω equal to 1, has been set to 2 µm. Smaller values would increase unwanted second-order effects such as channel-length shortening. Considering Equations (2) and (3), smaller synaptic weights can be implemented by increasing the gate length as described in Equation (4). Therefore a direct mapping between the synaptic weight and a physical parameter of the circuit, in this case the gate length of a transistor, is accomplished.

$$L_{MIN} = 2\ \mu m \rightarrow L = \frac{L_{MIN}}{\omega} \quad (4)$$

However, as presented in Equation (4), the gate length may reach very large values for small synaptic weights. Therefore, the maximum gate length has been set to 20 µm. Consequently the width of the input transistor has to be decreased in order to obtain weights that are smaller than 0.1.

All transistors in the circuit of Fig. 5 operate in strong inversion mode, which guarantees an optimal control of drain current. Weak inversion mode of operation would enable lowering power dissipation at the cost of a loss in drain current control.

III. Fault-Tolerant Boolean Gates

A. Faul-Tolerant Architecture

It has already been demonstrated in [6] that the architecture described in Figure 6 can successfully absorb a significant amount of defects affecting its second layer, depending on the number of redundant units.

Stuck-at faults in the logic layer have been considered, and a threshold equal to 0.5 has been chosen for the decision layer. Moreover, in order to obtain the probability of correct operation with respect to the number of faulty devices, Monte Carlo simulations have been performed using process parameters variations. Matching variations have not been considered because they lead to much smaller variations of the output level.

NOT, NOR and NAND Boolean gates implementing different levels of fault-tolerance have been built using spiking neurons. The following paragraphs only describe the NOT gate. However, similar results have also been found with NOR and NAND Boolean functions.

Ongoing research, based on software simulations, is focusing on various circuit topologies and parameters in a FFANN [7]. Moreover, future work will also implement learning algorithms in the circuits in order to adapt the correct synaptic values of FFANNs.

B. The NOT Boolean Gate

The NOT gate is implemented using an inhibitory synapse with a synaptic weight equal to one and an offset also equal to one connected to the input of a neuron (Fig. 7).

Figure 8 shows a fault-tolerant architecture used for the NOT gate with three redundant units. The averaging layer is implemented by a neuron with N excitatory synapses connected to its inputs, where N represents the number of redundant units. Moreover, the synaptic weights of the excitatory synapses are all identical and equal to N^{-1} in order to perform an average of the logic layer outputs.

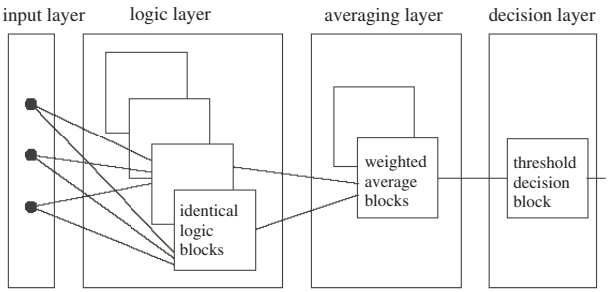

Figure 6. The proposed fault-tolerant architecture based on multiple layers.

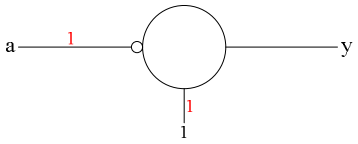

Figure 7. Symbolic representation of a NOT gate, consisting of one inhibitory synapse connected to a neuron.

C. *Simulation Results*

Figure 9 describes simulated results for the probability of correct operation obtained with the NOT Boolean function. For a given number of faulty neurons, fault immunity increases with the number of redundant units due to a benefit of the inherent neuron redundancy. However, the probability of correct operation of the gate does not depend on the density of faulty neurons in the logic layer. Similar results have been found for the NOR and NAND Boolean functions.

IV. CONCLUSION

In order to compensate the serious degradation of reliability of future very-deep submicron CMOS devices, a new design approach based on the implementation of multi-layer feed-forward artificial neural networks implemented in hardware has been developed. Spiking neuron models have been selected for the electronic integration, as they easily encode the analog signals used by FFANNs as mean frequencies of impulses.

CMOS circuit architectures implementing integrate-and-fire neurons, excitatory and inhibitory synapses have been elaborated. Moreover, a very straightforward mapping between the synaptic weights of FFANNs and a physical parameter of the circuits has been achieved. A basic library of fault-tolerant Boolean gates (NOT, NOR and NAND functions), based on a fault-tolerant architecture previously demonstrated [5], has been built in the AMS 0.35 µm CMOS technology.

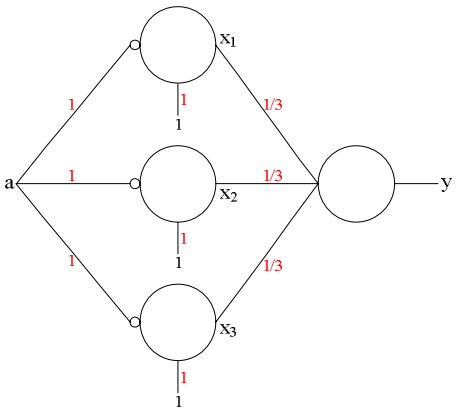

Figure 8. Fault-tolerant architecture of the NOT gate with three redundant units.

Considering logic errors such as stuck-at-one and stuck-at-zero, it has been shown that the Boolean gates are still able to operate correctly even with a number of faulty devices in the logic layer. Moreover, it has also been demonstrated that redundancy in this layer of the FFANN considerably improves the fault immunity of the gate.

The hardware overhead which is related to redundant units in the hidden layer is comparable to other fault-tolerant CMOS techniques based on redundancy.

Future work will consist of extending the library of logic gates to different Boolean functions and different degree of fault immunity. This library of fault-tolerant Boolean gates should also enable the synthesis of complex systems using the well-established conventional design-flow applied for digital CMOS architectures, while keeping all device-level development to enhance robustness and fault-tolerance largely transparent to the designer.

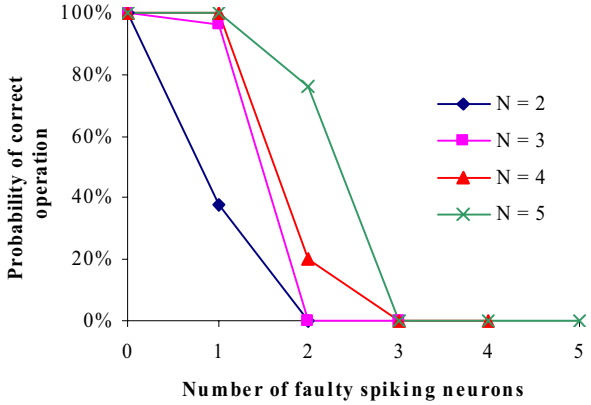

Figure 9. Probability of correct operation of the NOT gate against the number of faulty spiking neurons with two to five redundant neurons.

REFERENCES

[1] ITRS 2005 technology roadmap, http://public.itrs.net

[2] R. Eickhoff and U. Rückert, "Fault-tolerance of basis function networks using tensor product stabilizers," 2005 IEEE International Conference on Systems, Man and Cybernetics, vol.3, pp. 2144-2149.

[3] A. Schmid and Y. Leblebici, "A modular approach for reliable nanoelectronic and very-deep submicron circuit design based on analog neural network principles," in Proc. Third IEEE Conf. Nanotechnology, vol.2, pp.516-519, August 2003.

[4] J. Lazzaro, "Low-power silicon spiking neurons and axons," in Proc. IEEE ISCAS San Diego1992, vol.5, pp.2220-2224.

[5] T. Asai, Y. Kanazawa, and Y. Amemiya, "A subthreshold MOS neuron circuit based on the volterra system," IEEE Trans. Neural Netw., vol. 14, no.5, pp. 1308-1312, May 2003.

[6] A. Schmid and Y. Leblebici, "Robust and fault-tolerant circuit design for nanometer-scale devices and single-electron transistors," in Proc. IEEE ISCAS Vancouver 2004, vol. 3, pp. 685-688.

[7] M. Vural, A. Özgür, A. Schmid, and Y. Leblebici, "Fault tolerance of feed-forward artificial neural networl architecture targeting nano-scale implementations," IEEE MWSCAS Montreal 2007, in press.