EE451 Supervised Research Exposition

Topic : Analog/Mixed-Signal Circuits for Machine Learning Applications

Supervisor : Prof. Rajesh Zele

Mihir Kavishwar (17D070004)
Electrical Engineering, Dual Degree - Microelectronics
Indian Institute of Technology, Bombay

December 2020

Contents

1 Introduction

Over the last couple of decades there have been many advancements in the field of Machine Learning (ML) and today ML algorithms are used in a variety of applications. Given the computation intensive nature of these algorithms, designing hardware which can meet the power and performance requirements is very important. While efforts are being made to improve upon the existing digital processing architectures, there is emerging research which advocates using analog and mixed signal circuits for certain applications (such as Edge Devices) which have strict power and latency constraints. Analog/Mixed-Signal designs trade numerical accuracy for very high energy efficiency and performance, and are therefore well suited for Edge Devices.

This report summarises the work that was done during the course of one semester under the guidance of Prof. Rajesh Zele and his PhD student, Siva Elangovan, in Advanced Integrated Circuits and Systems (aiCAS) Lab.

2 Literature Survey

In this report I have discussed the following 5 papers from my survey:

1. J. Lazzaro, S. Ryckebusch, M. A. Mahowald and C. A. Mead, "**Winner-take-all networks of O(N) complexity**", Advances in Neural Information Processing Systems 1, Morgan Kaufmann Publishers, San Francisco, CA, 1989.
2. T. Delbruck, "**'Bump' circuits for computing similarity and dissimilarity of analog voltages**," IJCNN-91-Seattle International Joint Conference on Neural Networks, Seattle, WA, USA, 1991.
3. S. Peng, P. E. Hasler and D. V. Anderson, "**An Analog Programmable Multidimensional Radial Basis Function Based Classifier**," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 54, no. 10, pp. 2148-2158, Oct. 2007.
4. E. H. Lee and S. S. Wong, "**Analysis and Design of a Passive Switched-Capacitor Matrix Multiplier for Approximate Computing**," in IEEE Journal of Solid-State Circuits, vol. 52, no. 1, pp. 261-271, Jan. 2017.
5. J. Zhang, Z. Wang and N. Verma, "**In-Memory Computation of a Machine-Learning Classifier in a Standard 6T SRAM Array**," in IEEE Journal of Solid-State Circuits, vol. 52, no. 4, pp. 915-924, April 2017.

The circuits described in the first 2 papers form the backbone of many complex ML systems being implemented today which use Analog/Mixed signal processing. Therefore it is very useful to look at those papers first despite the fact that they were written almost 30 years ago. The next 3 papers are relatively recent,

and all of them use different approaches to realise an ML system.

Note: Many publications could not be included in this report but an interested reader may refer to the following comprehensive list of papers.

2.1 Winner-Take-All Circuits

Reference: J. Lazzaro, S. Ryckebusch, M. A. Mahowald and C. A. Mead, "**Winner-take-all networks of O(N) complexity**", Advances in Neural Information Processing Systems 1, Morgan Kaufmann Publishers, San Francisco, CA, 1989.

2.1.1 Introduction

Many classification algorithms in supervised learning use a winner-take-all (WTA) approach to produce the final outputs. The behaviour of a WTA block can be described as follows:
Given N input-output pairs, say

$$(input_k, output_k) \quad k \in \{1, 2, \ldots, N\}$$

Let

$$m = \underset{k \in \{1,2,\ldots,N\}}{\arg\max} \; input_k$$

Then

$$output_k = \begin{cases} 1, & if\ k = m \\ 0, & if\ k \neq m \end{cases}$$

That is, only the output corresponding to the largest input is high while all other outputs are 0.

2.1.2 Circuit Implementation and Analysis

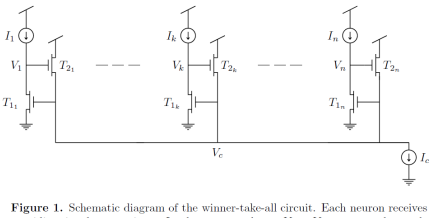


Figure 1. Schematic diagram of the winner-take-all circuit. Each neuron receives a unidirectional current input I_k; the output voltages $V_1 \ldots V_n$ represent the result of the winner-take-all computation. If $I_k = \max(I_1 \ldots I_n)$, then V_k is a logarithmic function of I_k; if $I_j \ll I_k$, then $V_j \approx 0$.

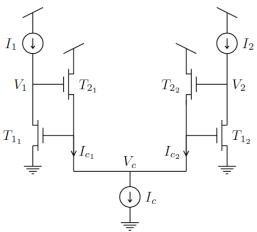


Figure 2. Schematic diagram of a two-neuron winner-take-all circuit.

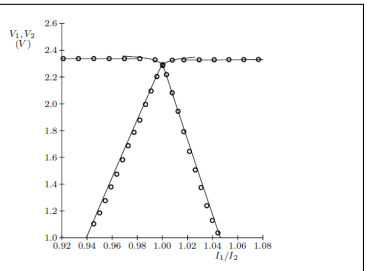


Figure 3. Experimental data (circles) and theory (solid lines) for a two-neuron winner-take-all circuit. I_1, the input current of the first neuron, is swept about the value of I_2, the input current of the second neuron; neuron voltage outputs V_1 and V_2 are plotted versus normalized input current.

To understand how the above circuit works, first let's look at a simpler version with just 2 inputs as shown in Figure 2. All 4 transistors are biased in **sub-threshold region**. The equation governing their behaviour is

$$I_{ds} = I_0 e^{\frac{\kappa V_{gs}}{V_T}} (1 - e^{-\frac{V_{ds}}{V_T}}) \tag{1}$$

where $V_T = \frac{KT}{q} \approx 26\, mV$ at room temperature. The factor $\kappa \approx 0.7$ accounts for back-gate or body effect. If we assume $V_{ds} \geq 3V_T$ then the effect of V_{ds} becomes negligible and transistor is said to operate in **sub-threshold saturation**

$$I_{ds} \approx I_0 e^{\frac{\kappa V_{gs}}{V_T}} \tag{2}$$

Consider the case when both inputs currents are equal $I_1 = I_2 \equiv I_m$. Transistors T_{1_1} and T_{1_2} have identical potentials at gate and source, and are both sinking I_m; thus, the drain potentials V_1 and V_2 must be equal, say V_m. Transistors T_{2_1} and T_{2_2} have identical source, drain, and gate potentials, and therefore must sink the identical current $I_{c_1} = I_{c_2} = \frac{I_c}{2}$. Assuming both T_1 and T_2 are in saturation and substituting in equation (2),

$$I_m \approx I_0 e^{\frac{\kappa V_c}{V_T}} \quad \Longrightarrow \quad \frac{\kappa V_c}{V_T} \approx ln\left(\frac{I_m}{I_0}\right)$$

$$\frac{I_c}{2} \approx I_0 e^{\frac{\kappa(V_m - V_c)}{V_T}} \quad \Longrightarrow \quad \frac{\kappa(V_m - V_c)}{V_T} \approx ln\left(\frac{I_c}{2I_0}\right)$$

$$V_m \approx \frac{V_T}{\kappa} ln\left(\frac{I_m}{I_0}\right) + \frac{V_T}{\kappa} ln\left(\frac{I_c}{2I_0}\right) \tag{3}$$

Thus, for equal input currents, the circuit produces equal output voltages; this behavior is desirable for a winner-take-all circuit. In addition, the output voltage V_m logarithmically encodes the magnitude of the input current I_m.

Now consider the case when $I_1 = I_m + \delta_i$ and $I_2 = I_m$. Transistor T_{1_1} must sink δ_i more current than in the previous example; as a result, the gate voltage of T_{1_1} rises. Transistors T_{1_1} and T_{1_2} share a common gate, however; thus, T_{1_2} must also sink $I_m + \delta_i$. But only I_m is present at the drain of T_{1_2}. To compensate,

the drain voltage of T_{1_2}, V_2, must decrease. For small δ_i, the Early effect serves to decrease the current through T_{1_2} , decreasing V_2 linearly with δ_i. For large δ_i, T_{1_2} must leave saturation, driving V_2 to approximately 0 volts. As desired, the output associated with the smaller input diminishes. For large δ_i, $I_{c_2} \approx 0$, and $I_{c_1} = I_c$. Therefore,

$$I_m + \delta_i \approx I_0 e^{\frac{\kappa V_c}{V_T}} \quad \Longrightarrow \quad \frac{\kappa V_c}{V_T} \approx ln\left(\frac{I_m + \delta_i}{I_0}\right)$$

$$I_{c_1} \approx I_c \approx I_0 e^{\frac{\kappa(V_1 - V_c)}{V_T}} \quad \Longrightarrow \quad \frac{\kappa(V_1 - V_c)}{V_T} \approx ln\left(\frac{I_c}{I_0}\right)$$

$$V_1 \approx \frac{V_T}{\kappa} ln\left(\frac{I_m + \delta_i}{I_0}\right) + \frac{V_T}{\kappa} ln\left(\frac{I_c}{I_0}\right) \quad ; \quad V_2 \approx 0 \tag{4}$$

The same logic can be extrapolated to a circuit with N inputs. The maximum current input sets the common gate voltage V_c, which forces other transistors to leave saturation as they are sinking smaller currents. The computational complexity is O(N) because the designs scales linearly with the number of inputs. The paper also briefly discusses the time response of this circuit and condition for stability. A local winner-take-all circuit is also discussed where the inhibitory action is limited to only adjacent neurons. I have not included the analysis here as the principle remains the same.

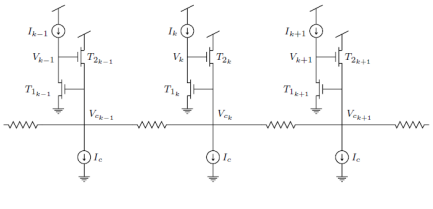


Figure 8. Schematic diagram of a section of the local winner-take-all circuit. Each neuron i receives a unidirectional current input I_i; the output voltages V_i represent the result of the local winner-take-all computation.

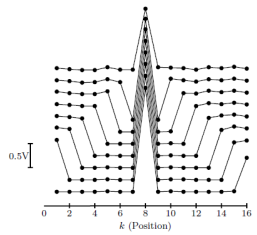


Figure 9. Experimental data showing the spatial impulse response of the local winner-take-all circuit, for values of I_x/I_c ranging over a factor of 12.7. Wider inhibitory responses correspond to larger ratios. For clarity, the plots are vertically displaced in 0.25 volt increments.

2.2 Bump Circuits

Reference: T. Delbruck, "**'Bump' circuits for computing similarity and dissimilarity of analog voltages**," IJCNN-91-Seattle International Joint Conference on Neural Networks, Seattle, WA, USA, 1991.

2.2.1 Introduction

Similarity measure or similarity function is a real-valued function that quantifies the similarity between two objects. These are in some sense inverse of distance metrics: **they take on large values for similar objects and either zero or a negative value for very dissimilar objects**. The notion of similarity is commonly used in many ML tasks:

- Clustering algorithms such as K-means clustering use Euclidean distance to compute similarity between two data points
- Radial Basis Function (RBF) Kernel which is commonly used in Support Vector Machines is actually a similarity function

Bump circuits implement a gaussian-like similarity function which "bumps" if the input voltages are close to each other and dies down if they are far apart. Anti-bump circuits implement a dis-similarity function which is like an inverse gaussian - it gives high output if input voltages are far apart and low output if they are close by.

2.2.2 Circuit Implementation and Analysis

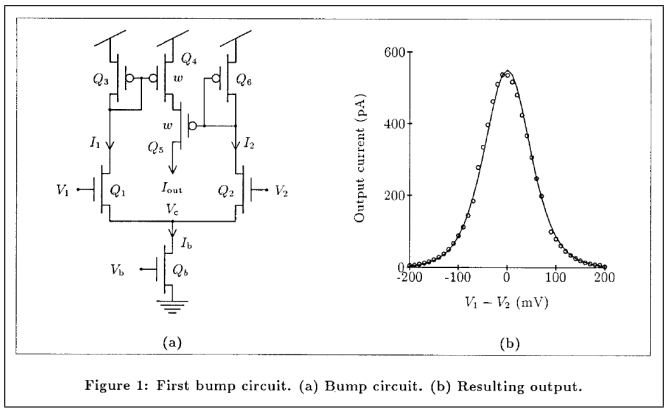


Figure 1: First bump circuit. (a) Bump circuit. (b) Resulting output.

The similarity output from the circuit, given as current, becomes large when the input voltages are close to each other. Bias current I_b is set by bias voltage V_b. Transistors Q_{1-6} are operating in subthreshold and Q_4 and Q_5 have their W:L ratios w times those of other transistors. Further, Q_1, Q_2 and Q_5 are assumed to be in subthreshold saturation, meaning that we can neglect the contribution of V_{ds} in drain current equation. First let's analyse the differential pair. Using the same subthreshold current equations (1 and 2) from the previous section with the assumption that **all voltages are in units of $\frac{\mathbf{KT}}{\mathbf{q}}$**,

$$I_1 = I_0 e^{\kappa(V_1 - V_c)} \quad ; \quad I_2 = I_0 e^{\kappa(V_2 - V_c)}$$

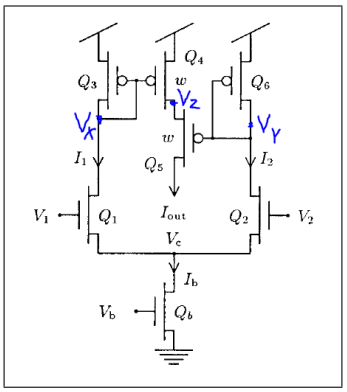
$$I_1 + I_2 = I_b$$

$$\implies I_1 = I_b \frac{e^{\kappa V_1}}{e^{\kappa V_1} + e^{\kappa V_2}} \tag{5}$$

$$and \quad I_2 = I_b \frac{e^{\kappa V_2}}{e^{\kappa V_1} + e^{\kappa V_2}} \tag{6}$$

From equations (5) and (6) we can conclude that **if $|\mathbf{\Delta V}| = |\mathbf{V_1} - \mathbf{V_2}|$ is larger than a few $\frac{\mathbf{KT}}{\mathbf{q}}$ then current in one of the two legs will shut off.**

Now let's look at the upper part of the circuit which is called a current correlator. The paper cites a reference which uses a slightly modified subthreshold current equation: (κV_{gs}) is replaced by $(\kappa V_g - V_s)$ when body is connected to appropriate supply rails. For κ close to 1, both give the same result and therefore I'll use the later as it simplifies the analysis.



$$I_{out} = wI_0 e^{V_{dd}-\kappa V_x}(1 - e^{-(V_{dd}-V_z)}) = wI_0 e^{V_z-\kappa V_y}$$
$$\implies e^{V_{dd}-\kappa V_x} - e^{V_z-\kappa V_x} = e^{V_z-\kappa V_y}$$
$$\implies e^{V_z} = \frac{e^{V_{dd}-\kappa V_x}}{e^{-\kappa V_x} + e^{-\kappa V_y}}$$

Substituting e^{V_z},

$$I_{out} = wI_0 \frac{e^{(V_{dd}-\kappa V_x)} e^{(V_{dd}-\kappa V_y)}}{e^{(V_{dd}-\kappa V_x)} + e^{(V_{dd}-\kappa V_y)}} \approx w \frac{I_1 I_2}{I_1 + I_2} \tag{7}$$

Intuitively, we can think about this circuit as an analog AND gate - for output to be high, both inputs need to be high. Substituting (5) and (6) in (7) gives,

$$I_{out} \approx w \frac{I_b}{2} sech^2 \left(\frac{\kappa(V_1 - V_2)}{2} \right) \tag{8}$$

where $sech(x) = \frac{2}{e^x + e^{-x}}$.

The following computes similarity as well as dissimilarity outputs as currents.

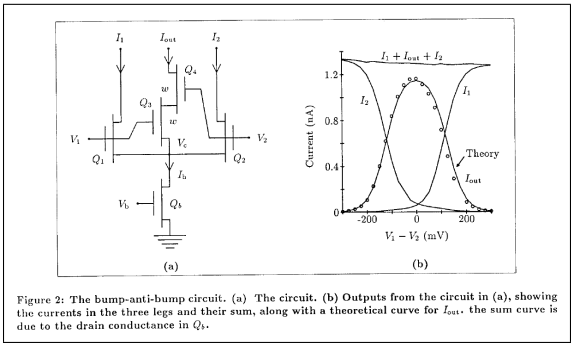


Figure 2: The bump-anti-bump circuit. (a) The circuit. (b) Outputs from the circuit in (a), showing the currents in the three legs and their sum, along with a theoretical curve for I_{out}. the sum curve is due to the drain conductance in Q_5.

Bias current I_b is set by V_b. Q_{1-4} are in subthreshold. Q_3 and Q_4 have their aspect ratio scaled by **w**. Q_1, Q_2 and Q_4 are assumed to be in subthreshold saturation.

$$I_1 = I_0 e^{\kappa V_1 - V_c} \quad ; \quad I_2 = I_0 e^{\kappa V_2 - V_c}$$

$$I_1 + I_2 + I_{out} = I_b$$

To solve for I_1 or I_2, first we will have to calculate I_{out} in terms of I_1 and I_2.

$$I_{out} = wI_0 e^{\kappa V_1 - V_c}(1 - e^{-(V_z - V_c)}) = wI_0 e^{\kappa V_2 - V_z}$$

$$\implies e^{\kappa V_1 - V_c} - e^{\kappa V_1 - V_z} = e^{\kappa V_2 - V_z}$$

$$\implies e^{-V_z} = \frac{e^{\kappa V_1 - V_c}}{e^{\kappa V_1} + e^{\kappa V_2}}$$

Substituting e^{-V_z},

$$I_{out} = wI_0 \frac{e^{(\kappa V_1 - V_c)} e^{(\kappa V_2 - V_c)}}{e^{(\kappa V_1 - V_c)} + e^{(\kappa V_2 - V_c)}} \approx w \frac{I_1 I_2}{I_1 + I_2} \tag{9}$$

Substituting this back into KCL equation we get,

$$I_{out} = \frac{I_b}{1 + \frac{4}{w} cosh^2\left(\frac{\kappa(V_1 - V_2)}{2}\right)} \tag{10}$$

where $cosh(x) = \frac{e^x + e^{-x}}{2}$ We get the anti-bump current by adding I_1 and I_2,

$$I_1 + I_2 = I_b - I_{out}$$

The following is an extension of bump-anti-bump circuit. It was designed to produce a transconductance element that would ignore small voltage offsets on the input voltage, ΔV,while retaining a monotonic yet saturating output characteristic for larger inputs.

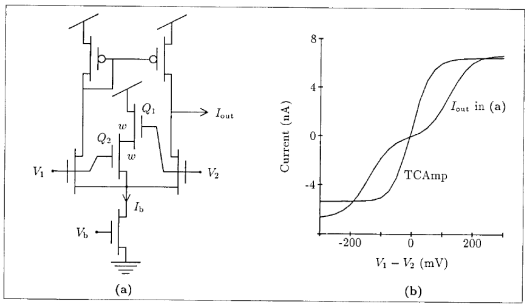


Figure 3: Modified transconductance circuit. (a) Bump circuit in Figure 2(a) modified to compute the difference $I_1 - I_2$. (b) Resulting output compared with the output from a transconductance amplifier (TCAmp).

2.3 Analog Programmable Multidimensional RBF based Classifier

Reference: S. Peng, P. E. Hasler and D. V. Anderson, "**An Analog Programmable Multidimensional Radial Basis Function Based Classifier**," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 54, no. 10, pp. 2148-2158, Oct. 2007.

2.3.1 Introduction

A real-valued function φ whose value depends only on the distance between the input and some fixed point, $\mathbf{c}$, called a center, so that $\varphi(\mathbf{x}) = \varphi(\|\mathbf{x} - \mathbf{c}\|)$ is called Radial Basis Function (RBF). A classifier which uses RBFs to determine decision boundaries is called an RBF classifier. A vector quantizer (multi-class classifier) compares distances or similarities between input vectors and the stored templates and classifies the input data to the most representative template. This paper discusses the development of an analog Gaussian response function (which is an RBF) having a diagonal covariance matrix and demonstrates its application to vector quantization.

2.3.2 System Overview

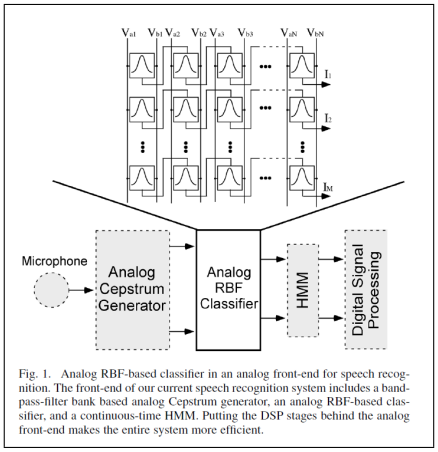


Fig. 1. Analog RBF-based classifier in an analog front-end for speech recognition. The front-end of our current speech recognition system includes a band-pass-filter bank based analog Cepstrum generator, an analog RBF-based classifier, and a continuous-time HMM. Putting the DSP stages behind the analog front-end makes the entire system more efficient.

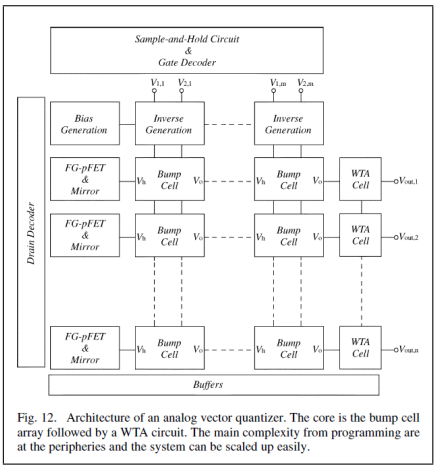
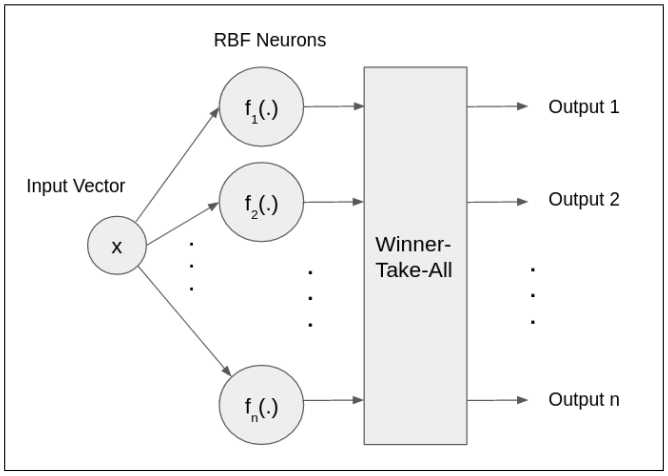


Fig. 12. Architecture of an analog vector quantizer. The core is the bump cell array followed by a WTA circuit. The main complexity from programming are at the peripheries and the system can be scaled up easily.

The classification algorithm uses multidimensional Gaussian function with diagonal covariance matrix Σ, mean vector μ , and scaling factor K as the RBF function

$$f_{\mu,\Sigma,K}(\mathbf{x}) = K.exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right)$$

The following block diagram explains the classification algorithm:



$f_i(.)$ are all multivariate gaussian functions with different values of μ, Σ and K. Therefore,

$$\text{class into which input is classified} = \underset{i}{argmax}\ f_{\mu_i,\Sigma_i,K_i}(\mathbf{x}) \qquad (11)$$

2.3.3 Circuit Implementation and Analysis

The function implemented by bump circuit can be approximated as a gaussian but we need to modify it so that we can control center, width and height of the curve (μ, Σ and K respectively). We also have to make it multi-dimensional.

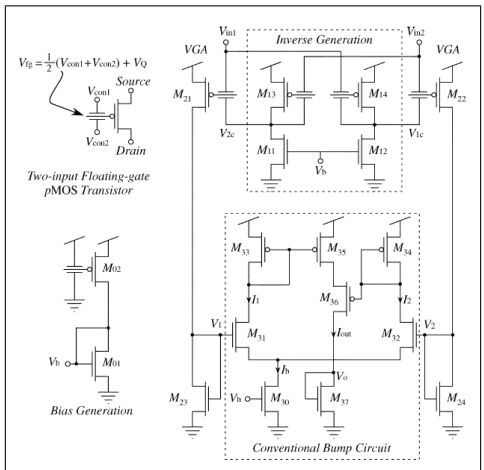


Fig. 2. Schematics of the new floating-gate bump circuit. All floating-gate transistors in the schematics have two inputs with equal weights and the floating-gate voltage can be expressed as $V_{\mathrm{fg}} = 1/2(V_{\mathrm{con1}} + V_{\mathrm{con2}}) + V_Q$, where $V_Q = Q_{\mathrm{fg}}/C_{\mathrm{T}}$, Q_{fg} is the charge on the floating gate and C_{T} is the total capacitance from the floating gate. The bump circuit is composed of an inverse generation block, a fully differential VGA, and a conventional bump circuit. The width and the center of the bell-shaped transfer function are set by the common-mode and differential charges on M_{21} and M_{22}. The height is controlled by the tail current I_{b}. All of them are independently programmable.

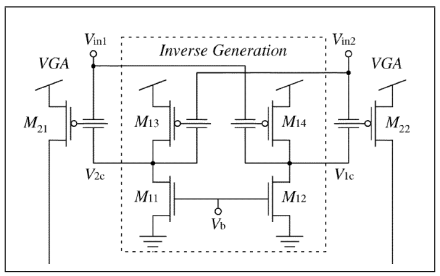
The paper implements a **Floating Gate Bump Circuit** as shown above. The circuit comprises of 2-input floating-gate transistors which can be programmed using Fowler–Nordheim tunneling and channel hot electron (CHE) injection mechanisms.

$$V_{fg} = \frac{1}{2}(V_{con1} + V_{con2}) + V_Q \quad ; \quad V_Q = \frac{Q_{fg}}{C_T} \tag{12}$$

The first step for realising desired function is additive inverse generation. Using the above equation for this circuit we can see,

$$V_{in1} + V_{1c} = V_{in2} + V_{2c} = V_{const}$$

$$\implies V_{1c} = V_{const} - V_{in1} \quad ; \quad V_{2c} = V_{const} - V_{in2}$$



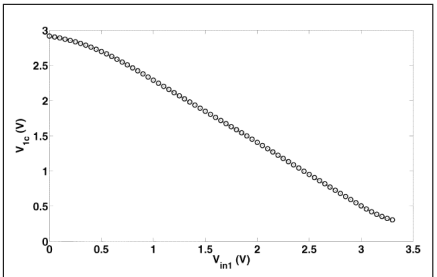


Fig. 3. Inverse generation transfer characteristics. A floating-gate summing amplifier generates a complementary input voltage. This outputs are fed to floating-gate transistors in the VGA so that the outputs of the VGA are independent of the input common-mode signals. With an appropriate bias voltage, the operating range is one V_{DSsat} away from supply rails.

To proceed, we first define 2 quantities,

$$V_{Q,cm} = \frac{V_{Q,21} + V_{Q,22} + V_{const}}{2} \quad ; \quad V_{Q,dm} = \frac{V_{Q,21} - V_{Q,22}}{2}$$

Now,

$$V_{fg,21} = \frac{V_{in1} + V_{2c}}{2} + V_{Q,21} = \frac{V_{in1} + V_{const} - Vin2}{2} + V_{Q,21} = \frac{\Delta V_{in}}{2} + \frac{V_{const}}{2} + V_{Q,21}$$

$$V_{fg,21} = \frac{1}{2}\Delta V_{in} + \frac{1}{2}V_{Q,dm} + V_{Q,cm} \tag{13}$$

$$V_{fg,22} = \frac{V_{in2} + V_{1c}}{2} + V_{Q,22} = \frac{V_{in2} + V_{const} - Vin1}{2} + V_{Q,22} = -\frac{\Delta V_{in}}{2} + \frac{V_{const}}{2} + V_{Q,22}$$

$$V_{fg,22} = -\frac{1}{2}\Delta V_{in} - \frac{1}{2}V_{Q,dm} + V_{Q,cm} \tag{14}$$

$$V_1 - V_2 = \eta.(V_{fg,21} - V_{fg,22}) = \eta.(\Delta V_{in} + V_{Q,dm}) \tag{15}$$

Where η is the gain of VGA and depends on $V_{Q,cm}$. Substituting (15) in (8) we get,

$$I_{out} \approx K.exp\left(-\eta'(\Delta V_{in} + V_{Q,dm})^2\right) \tag{16}$$

Here we have assumed $sech^2(x) \approx e^{-x^2}$. K is a function of bias current I_b and η' is a function of $V_{Q,cm}$. Therefore by programming I_b, $V_{Q,cm}$ and $V_{Q,dm}$ we can vary height, width and center of the curve respectively.

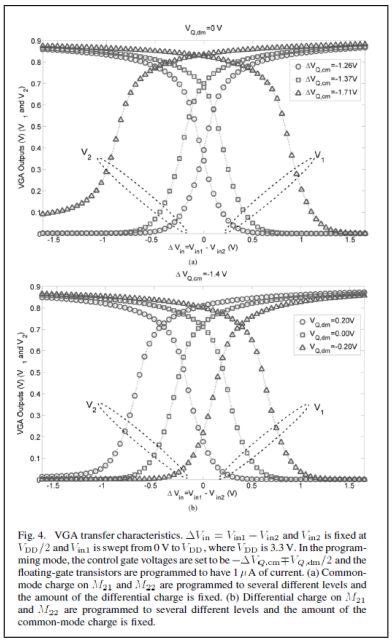


Fig. 4. VGA transfer characteristics. $\Delta V_{\rm in} = V_{\rm in1} - V_{\rm in2}$ and $V_{\rm in2}$ is fixed at $V_{\rm DD}/2$ and $V_{\rm in1}$ is swept from 0 V to $V_{\rm DD}$, where $V_{\rm DD}$ is 3.3 V. In the programming mode, the control gate voltages are set to be $-\Delta V_{Q,\rm cm} \mp V_{Q,\rm dm}/2$ and the floating-gate transistors are programmed to have 1 μA of current. (a) Common-mode charge on M_{21} and M_{22} are programmed to several different levels and the amount of the differential charge is fixed. (b) Differential charge on M_{21} and M_{22} are programmed to several different levels and the amount of the common-mode charge is fixed.

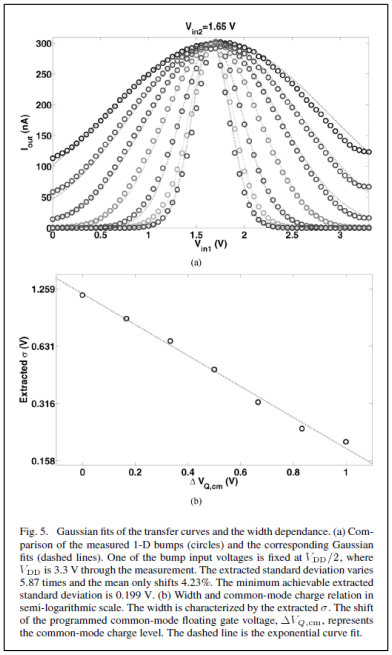


Fig. 5. Gaussian fits of the transfer curves and the width dependance. (a) Comparison of the measured 1-D bumps (circles) and the corresponding Gaussian fits (dashed lines). One of the bump input voltages is fixed at $V_{\rm DD}/2$, where $V_{\rm DD}$ is 3.3 V through the measurement. The extracted standard deviation varies 5.87 times and the mean only shifts 4.23%. The minimum achievable extracted standard deviation is 0.199 V. (b) Width and common-mode charge relation in semi-logarithmic scale. The width is characterized by the extracted σ. The shift of the programmed common-mode floating gate voltage, $\Delta V_{Q,\rm cm}$, represents the common-mode charge level. The dashed line is the exponential curve fit.

Now, to extend this to multiple dimensions we simple cascade many bump cells

$$
\begin{aligned}
I_{out2} &\approx w_2 \frac{I_{b2}}{2} exp\left(-\eta_2'(\Delta V_{in2} + V_{Q,dm2})^2\right) \\
&\approx w_1 w_2 \frac{I_{b1}}{4} exp\left(-\eta_1'(\Delta V_{in1} + V_{Q,dm1})^2\right) exp\left(-\eta_2'(\Delta V_{in2} + V_{Q,dm2})^2\right) \\
I_{out2} &\approx K'.exp\left(-\eta_1'(\Delta V_{in1} + V_{Q,dm1})^2 - \eta_2'(\Delta V_{in2} + V_{Q,dm2})^2\right) \qquad (17)
\end{aligned}
$$

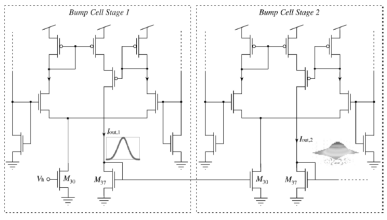


Fig. 6. Cascading bump circuits. By connecting the diode-connected output transistor to the tail transistor of the next stage bump cell, the resulting output current can approximate a multivariate Gaussian function with a diagonal covariance matrix.

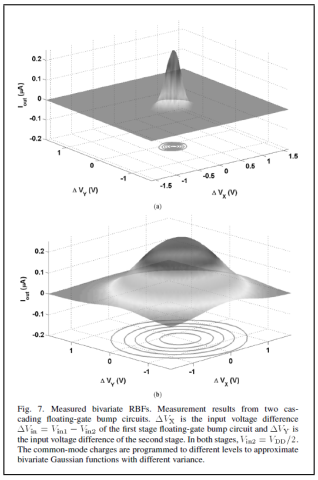


Fig. 7. Measured bivariate RBFs. Measurement results from two cascading floating-gate bump circuits. ΔV_X is the input voltage difference $\Delta V_X = V_{in1} - V_{in2}$ of the first stage floating-gate bump circuit and ΔV_Y is the input voltage difference of the second stage. In both stages, $V_{in2} = V_{DD}/2$. The common-mode charges are programmed to different levels to approximate bivariate Gaussian functions with different variance.

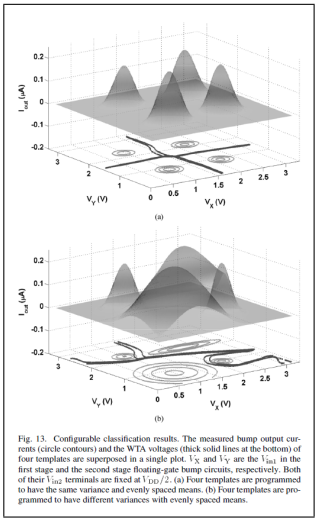


Fig. 13. Configurable classification results. The measured bump output currents (circle contours) and the WTA voltages (thick solid lines at the bottom) of four templates are superposed in a single plot. V_X and V_Y are the V_{in1} in the first stage and the second stage floating-gate bump circuits, respectively. Both of their V_{in2} terminals are fixed at $V_{DD}/2$. (a) Four templates are programmed to have the same variance and evenly spaced means. (b) Four templates are programmed to have different variances with evenly spaced means.

2.4 Switched-Capacitor Matrix Multiplier

Reference: E. H. Lee and S. S. Wong, "**Analysis and Design of a Passive Switched-Capacitor Matrix Multiplier for Approximate Computing**," in IEEE Journal of Solid-State Circuits, vol. 52, no. 1, pp. 261-271, Jan. 2017.

2.4.1 Introduction

This paper presents a switched-capacitor matrix multiplier for approximate computing and machine learning applications. Let x be the $n \times 1$ input vector, A be an $m \times n$ matrix whose elements are stored in memory and $y = Ax$ be the $m \times 1$ output vector; then

$$y[j] = \sum_{i=1}^{n} A[j,i].x[i]$$

x can be in analog or digital domain. If it's in digital we need a DAC to convert it to analog since the computation happens in analog domain. The output is again converted to digital using an ADC.

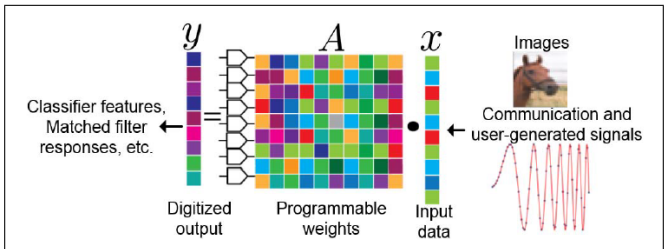


Fig. 1. Analog matrix multiplication for various signal processing applications.

2.4.2 Circuit Implementation and Analysis

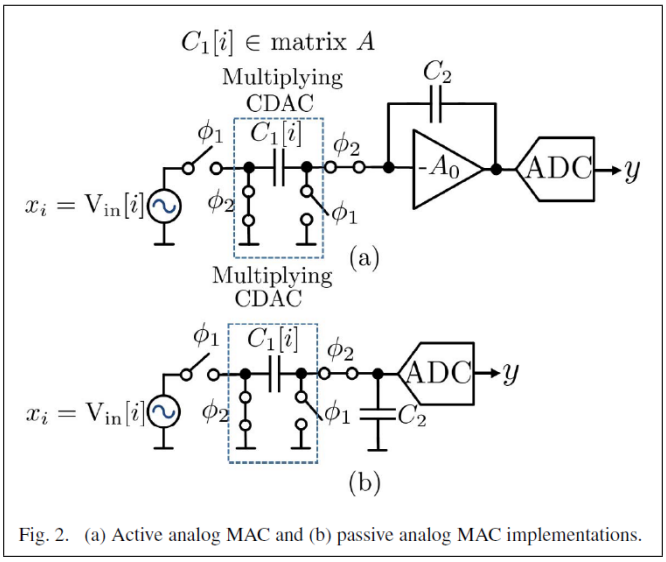
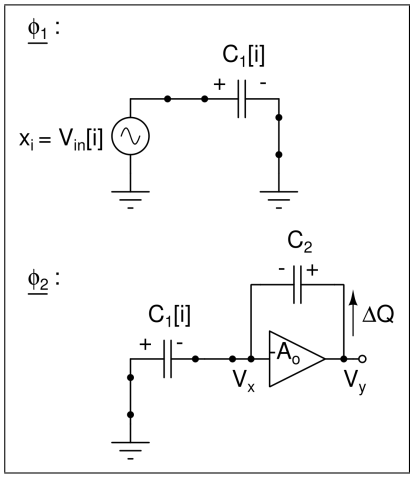
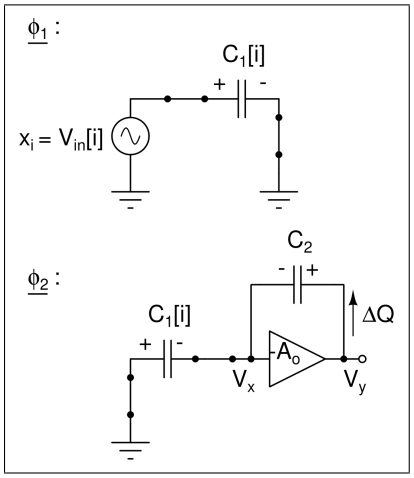


Fig. 2. (a) Active analog MAC and (b) passive analog MAC implementations.



(a) Active SC-MAC



(b) Passive SC-MAC

First let's analyse Active SC-MAC.

$$Q_{1,\phi_1}[i] = C_1[i].V_{in}[i] \quad ; \quad Q_{2,\phi_1}[i] = Q_{2,\phi_2}[i-1]$$

$$Q_{1,\phi_2}[i] = Q_{1,\phi_1}[i] - \Delta Q = C_1[i].V_{in}[i] - \Delta Q = -C_1[i].V_x[i] \tag{18}$$

$$Q_{2,\phi_2}[i] = Q_{2,\phi_1}[i] + \Delta Q = Q_{2,\phi_2}[i-1] + \Delta Q = C_2.(V_y[i] - V_x[i]) = -C_2.V_x[i].(1+A_0) \tag{19}$$

Eliminating ΔQ from above 2 equations,

$$C_1[i].V_{in}[i] + Q_{2,\phi_2}[i-1] = -\left(C_1[i] + C_2.(1+A_0)\right).V_x[i]$$

$$\implies V_x[i] = -\frac{C_1[i].V_{in}[i]}{C_1[i] + C_2.(1+A_0)} - \frac{Q_{2,\phi_2}[i-1]}{C_1[i] + C_2.(1+A_0)}$$

Substituting back in equation (19),

$$V_{C_2}[i] = \frac{C_1[i].(1+A_0)}{C_1[i] + C_2.(1+A_0)} V_{in}[i] + \frac{C_2.(1+A_0)}{C_1[i] + C_2.(1+A_0)} V_{C_2}[i-1] \tag{20}$$

Let $k[i] = \frac{C_2(1+A_0)}{C_1[i]+C_2(1+A_0)}$ and $\mu[i] = \frac{C_1[i]}{C_2}$ then,

$$V_{C_2}[i] = \mu[i].k[i].V_{in}[i] + k[i].V_{C_2}[i-1]$$

We only care about the final result after n cycles, therefore expanding the recursive equation gives us

$$V_{C_2}[n] = \sum_{i=1}^{n} \left(\mu[i].V_{in}[i]. \prod_{j=i}^{n} k[j] \right) \tag{21}$$

We can write $V_{C_2}[n]$ as dot product of 2 vectors,

$$V_{C_2}[n] = \begin{bmatrix} \mu[1] \prod_{j=1}^{n} k[j] & \mu[2] \prod_{j=2}^{n} k[j] & \dots & \mu[n]k[n] \end{bmatrix} \begin{bmatrix} V_{in}[1] \\ V_{in}[2] \\ \vdots \\ V_{in}[n] \end{bmatrix}$$

Performing this operation on m rows in the matrix A results in a nonideal matrix operation $y = \tilde{A}x$, where

$$\tilde{A} = \begin{bmatrix} \mu[1,1] \prod_{j=1}^{n} k[1,j] & \mu[1,2] \prod_{j=2}^{n} k[1,j] & \dots & \mu[1,n]k[1,n] \\ \mu[2,1] \prod_{j=1}^{n} k[2,j] & \mu[2,2] \prod_{j=2}^{n} k[2,j] & \dots & \mu[2,n]k[2,n] \\ \vdots & \vdots & \ddots & \vdots \\ \mu[m,1] \prod_{j=1}^{n} k[m,j] & \mu[m,2] \prod_{j=2}^{n} k[m,j] & \dots & \mu[m,n]k[m,n] \end{bmatrix} \tag{22}$$

After doing a similar analysis for passive SC-MAC case, we get

$$V_{C_2}[i] = \frac{C_1[i]}{C_1[i] + C_2} V_{in}[i] + \frac{C_2}{C_1[i] + C_2} V_{C_2}[i-1] \tag{23}$$

This can also be written as,

$$V_{C_2}[i] = \mu[i].k[i].V_{in}[i] + k[i].V_{C_2}[i-1]$$

where $k[i] = \frac{C_2}{C_1[i]+C_2}$ and $\mu[i] = \frac{C_1[i]}{C_2}$. Therefore $\tilde{A}$ describes the matrix for passive case too. Note that ideally we would like $k[i] = 1$ so that $\tilde{A} = A$.

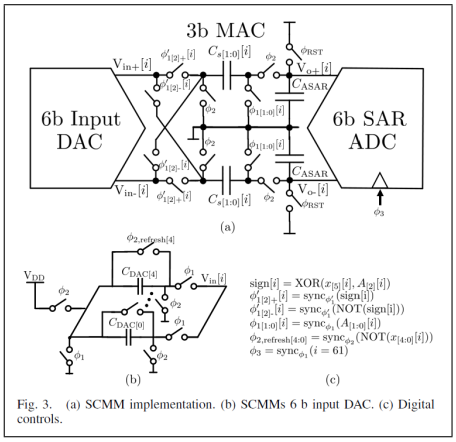


Fig. 3. (a) SCMM implementation. (b) SCMMs 6 b input DAC. (c) Digital controls.

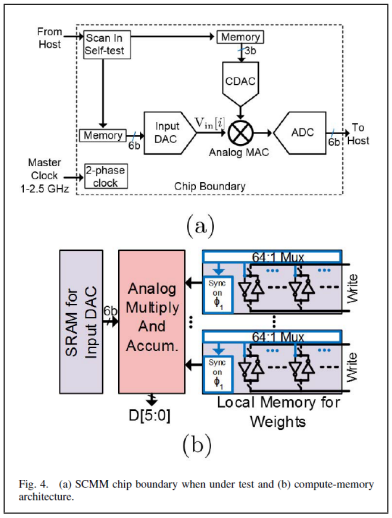


Fig. 4. (a) SCMM chip boundary when under test and (b) compute-memory architecture.

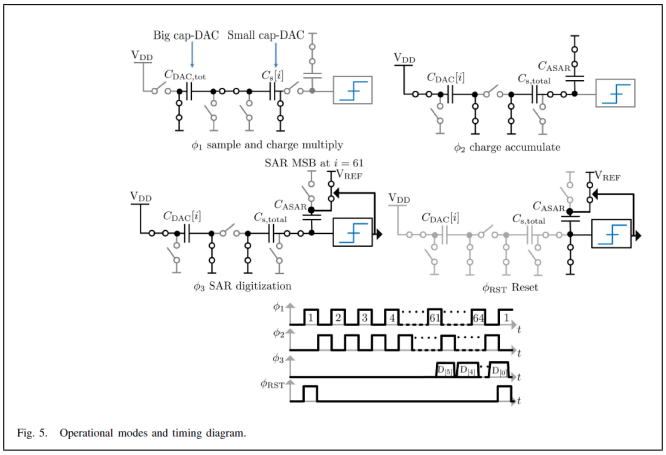


Fig. 5. Operational modes and timing diagram.

To correct for the non-ideality given in equation (22), the authors proposed multiplying the output by a normalisation constant and then applying another matrix B in digital domain. Formally, we solve for B in

$$\min_{B \epsilon \Omega_B} ||A - B\tilde{A}||_F$$

where F is the Frobenius norm, A is the intended ideal matrix, $\tilde{A}$ is the actual matrix due to incomplete charge accumulation, B is the correction matrix, and Ω_B is the set of possible values that B can take on.

2.4.3 Applications and Results

The paper discusses results for two applications where SC-MM was used

1. Energy-efficient feature extraction layer performing both compression and classification in a neural network for an analog front end and
2. Analog acceleration for solving optimization problems that are traditionally performed in the digital domain

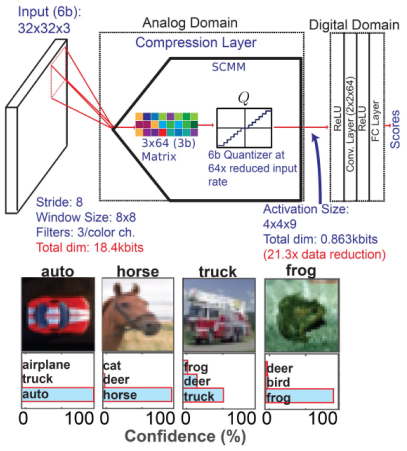


Fig. 14. Designed compression and classifying layer for a small convolutional neural network and measured confidence levels for sampled input images on the CIFAR-10 database.

TABLE I
PERFORMANCE OF THE COMPRESSION LAYER COMPARED WITH THE CONVENTIONAL

	Conventional	This work
Top-3 Accuracy (%)	86	85
Layer's Energy/Op (fJ) at 1GHz	145*	13
# of A/Ds per image at 6b	3072	144
Resolution	6b/4b/6b	Analog/3b /6b
NMSE of feature outputs (avg. over all batches)	0.0033	0.0054

*Energy estimated by synthesis in 40nm
Resolution Notation: Input/Weights/Output

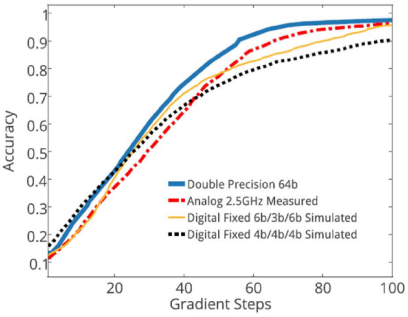


Fig. 15. Classification accuracy as a function of the number of gradient steps in the stochastic gradient descent procedure. NMSE of analog averaged over 100 steps is 0.006.

TABLE II
PERFORMANCE SUMMARY

	ISSCC 2015 [28]	This Work	
Technology	130nm	40nm	
Application	Sensor Classifier	Sensor Classifier	Analog Accelerator
Analog Multiply Rate	20kHz*	1GHz	2.5GHz
Analog Accumulate Rate	N/A	1GHz	2.5GHz
A/D Rate	20kS/s	15MS/s	39MS/s
Resolution	Analog / 4b* / 8b	Analog / 3b / 6b	6b / 3b / 6b
Supply	1.2V	1V	1.1V
Total Power	663nW	228uW**	647uW**
Efficiency (TOPS/W)	0.0603*	8.77**	7.72**

*4b analog multiply and bitshifting after SAR digitization
**includes: SCMM, self-test logic, clock and excludes CML
Resolution notation: Input/Weights/Output

2.5 In-Memory Computation of an ML Classifier in 6T-SRAM array

Reference: J. Zhang, Z. Wang and N. Verma, "**In-Memory Computation of a Machine-Learning Classifier in a Standard 6T SRAM Array**," in IEEE Journal of Solid-State Circuits, vol. 52, no. 4, pp. 915-924, April 2017.

2.5.1 Introduction

An underlying limitation emerges in current architectures for digital accelerators, which separate data storage from computation. Storing the data fundamentally associated with a computation requires area, and thus, its communication to the location of computation incurs energy and throughput cost, which can dominate. This has motivated thinking about architectures that integrate some forms of memory and compute This paper talks about a machine learning classifier where data storage and computation (MAC operations) are combined in a standard 6T SRAM array.

2.5.2 System Overview

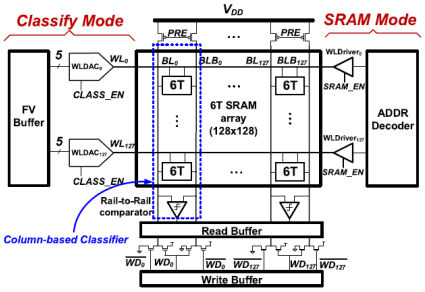


Fig. 2. Architecture of in-memory classifier, showing periphery for two modes of operation.

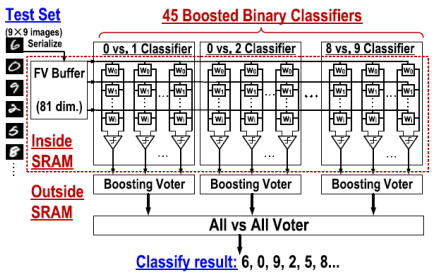


Fig. 17. Implementation of demonstration system for classifying 0–9 handwritten numerical data from MNIST data set.

In the SRAM mode, the operation is typical read/write of digital data. This is how machine-learning models derived from training are stored in bit cells.

In the Classify Mode, all wordlines (WLs) are driven at once to analog voltages. Thus, parallel operation of all bit cells is involved (by comparison, and in the SRAM mode, only one WL is driven at a time). Each analog WL voltage corresponds to a feature in a feature vector we wish to classify. The features are provided as digital data, loaded through the feature-vector buffer, and analog voltages are generated using a WLDAC in each row.

Every column based classifier computes a decision d given by

$$d = sgn\left(\sum_{i=1}^{N} w_i \times x_i\right) \tag{24}$$

where x_i corresponds to elements from a feature vector $\overrightarrow{x}$, and w_i corresponds to weights in a weight vector $\overrightarrow{w}$, derived from training. The circuit implementation makes this a "weak classifier" (cannot be trained to fit arbitrary data distributions) because not only are the bit-cell currents non-ideal, the weights are also restricted to +1 and -1. A "strong classifier" can be constructed by combining many such columns, as shown in figure 17, using a technique called Error-Adaptive Classifier Boosting.

2.5.3 Circuit Implementation and Analysis

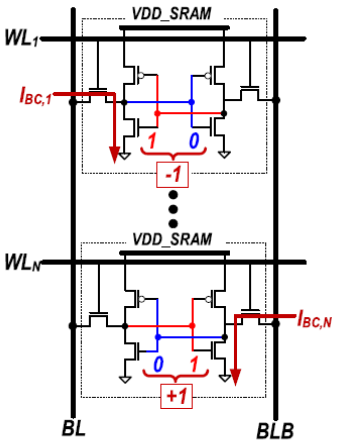


Fig. 3. Column-based weak classifier within the in-memory classifier architecture.

In SRAM mode, the read and write operations work exactly like those in standard 6T-SRAM. All the transistors have to be scaled appropriately to prevent read upset and enable write operation.

In classify mode, the wordline (WL_i) carries input analog voltage. Consider the circuit where the weight is -1 as shown in the figure. Both BL and BLB are precharged, so current flows only through the access transistor connecting BL and 0. Since $V_{WLi} < V_{DD}$, it operates in saturation

$$I_{BC,i} \approx \frac{K}{2}.(V_{WLi} - V_{tn})^2$$

The flow of current causes discharge of capacitor on BL.

$$\Delta V_{BL,i} = \frac{I_{BC,i} \times \Delta t}{C_{BL}}$$

Similarly, if the weight stored is +1, current flows through access transistor connected to BLB and causes discharge of capacitor on BLB. Parallel currents add up and result in some total discharge on both BL and BLB. A sense amplifier senses which line has greater discharge and gives 1 bit output.

$$d = sgn\left(\Delta V_{BLB,tot} - \Delta V_{BL,tot}\right) \propto sgn\left(\sum_{i=1}^{n} I_{BC,i} \times w_i\right) \tag{25}$$

This is what we want to implement as per equation (24), if $I_{BC,i}$ represents input.

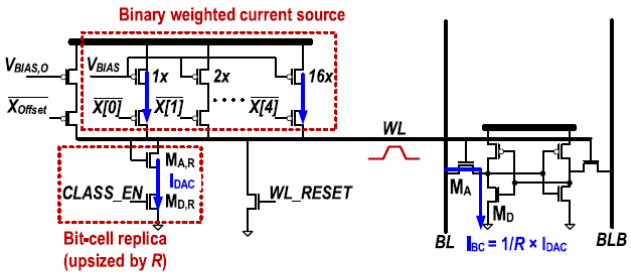


Fig. 7. WLDAC circuit, consisting of binary-weighted current sources and a bit-cell replica, to generate analog voltage on WL corresponding to inputted digital feature values.

The WLDAC is implemented using binary weighted current sources connected in parallel. If class_en is high,

$$I_{DAC} = I_{bias} \times x_{offset} + \sum_{i=0}^{Nbits} I_0 \times 2^i \times x[i] \tag{26}$$

Transistor $M_{A,R}$ sets the WL voltage. Access transistor M_A mirrors $M_{A,R}$ with scaling such that $I_{BC} = \frac{1}{R} I_{DAC}$. Therefore, $I_{BC,i}$ does represent the input and equation (25) matches (24).

3 Simulations

The simulations were a joint effort by me and Anubhav Agarwal. We simulated a column based weak classifier as described in the In-Memory computation paper discussed above.

3.1 Setup

Simulations were carried out with Cadence AMS. UMC65 nm technology was used. The circuit was designed to work for a 100 element sized feature vector input. Every element is represented in 8 bits. The circuit comprises of the following blocks:

1. **FV Buffer** : This block was implemented in Verilog. It allows us to store an array of input values present a text file, into a register as bit values.
2. **WLDAC** : This block was implemented in VerilogAMS. It converts the digital input to analog voltage between 0 and 1.2 V whenever CLASS_EN goes high, and presents high output impedance when CLASS_EN is low.
3. **ADDR Decoder and WLDriver**: This block was implemented in Verilog. It decodes the input address and drives the wordlines whenever SRAM_EN is high, and presents high output impedance when SRAM_EN is low.
4. **Write Driver**: This block was implemented as transistor level schematic. In SRAM mode, it drives the bit data in appropriate bit cell when WRITE_EN is high.
5. **6T SRAM Cell** : This block was implemented as transistor level schematic by appropriate scaling of transistors. It stores the weights as bits.

Same blocks were instantiated multiple times and bus connections were used wherever required.

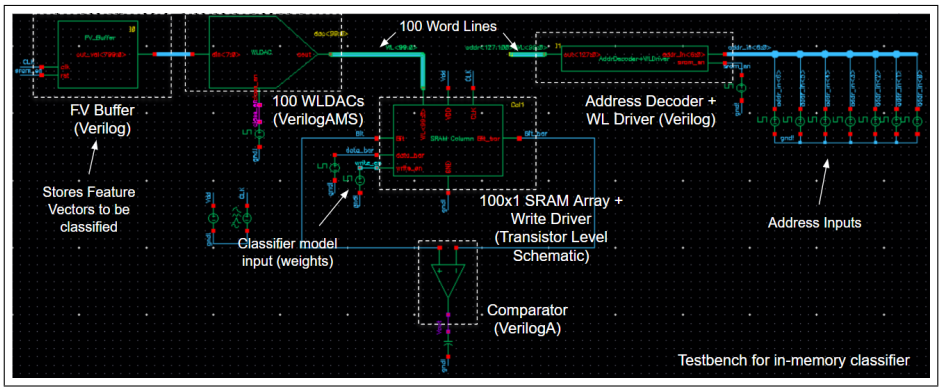


Figure 12: Full Testbench Schematic

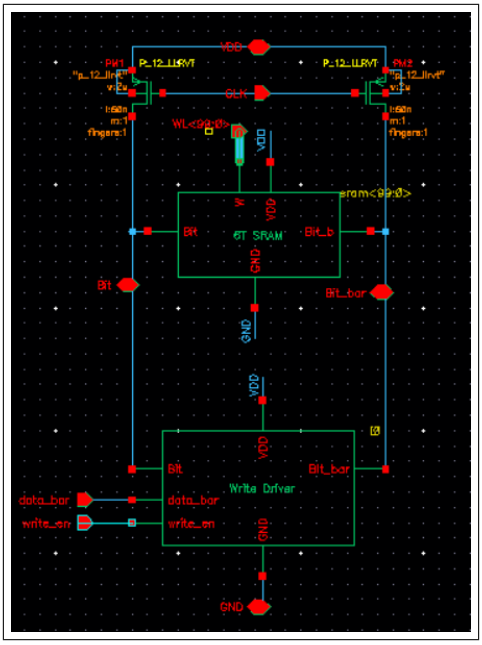


Figure 13: Column of SRAM cells

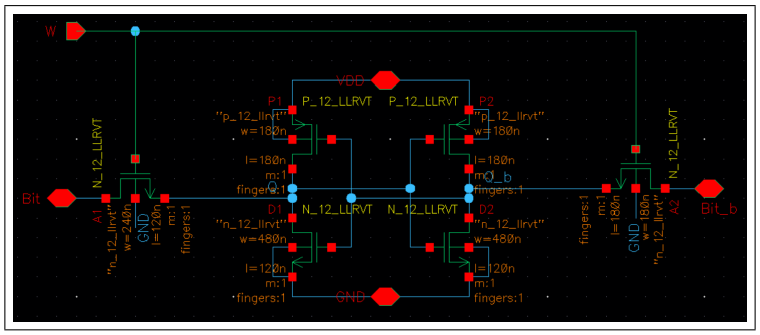


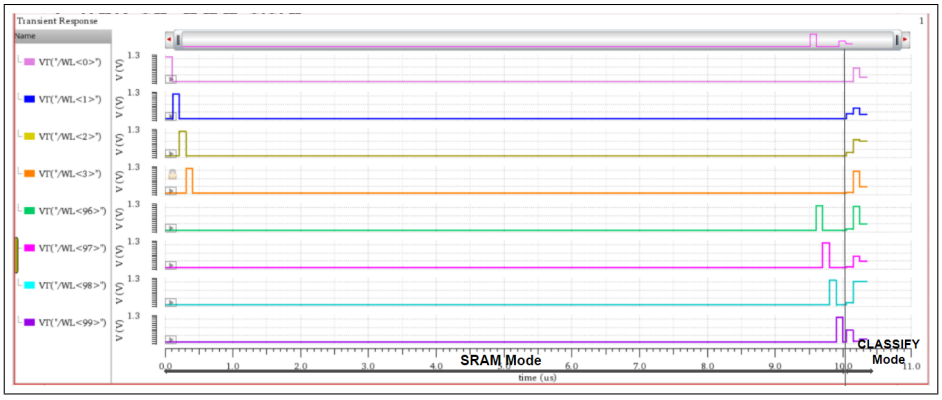
Figure 14: SRAM Cell

3.2 Results

We did a transient analysis. In the first 100 clock cycles, weights were written into bit cells and in the next 3 clock cycles feature vector inputs were classified. For the sake of simplicity, clock frequency for SRAM mode and CLASSIFY mode was kept same (10 MHz).

During sequential writing, only the WL corresponding to cell being written goes high while the rest go to 0. When inputs are being classified, different

analog voltages corresponding to inputs appear on all WLs. This process can be seen in the figure below were WL_{0-3} and WL_{97-99} signals are plotted,



V_{out} in classify mode represents the decision as per equation (25). In the plot below we see that the first feature vector was classified as 1 while the next two were classified as 0. This matched the expected result computed using python.