Memristor Bridge Circuit for Neural Synaptic Weighting

Maheshwar Pd. Sah[1], Changju Yang[1], Hyongsuk Kim[1], Tamás Roska[2] and Leon Chua[3]

[1]Division of Electronics and Information Engineering, Chonbuk National University, Republic of Korea

[2]Computer and Automation Research Institute of the Hungarian Academy of Sciences, Budapest, H-1518, Hungary

[3]Electrical Engineering and Computer Sciences University of California Berkerley, USA

***Abstract*—A simple and compact memristor-based bridge circuit which is able to perform signed synaptic weighting in neuron cells is proposed. The proposed memristor-based synapse is composed of four memristors which makes a bridge type configuration. By programming different values on each memristor of the memristor bridge circuit, weighting values can be set on the memristor bridge synapses. Various simulation results are included.**

I. INTRODUCTION

Synaptic circuits are core components of artificial neural networks and Cellular Neural Networks (CNN). There have been lots of research efforts to build artificial synapses (weights) in neural network chip and CNN [1]-[3]. For these applications, the implementation of immense amount of circuits into a chip is needed. It is a very challenging goal and not many successful cases of neural implementations have been reported so far. For such synapse and analog multiplication, CMOS weighting circuit is generally used in conventional synaptic circuits. However, non-volatile and programmable weighting functions are not possessed in the conventional synaptic circuits. Also, nonlinearity in synaptic multiplications between input and weight is also a problem of the conventional circuit. Therefore, introducing a new weighting circuit is inevitable for further development in neuromorphic engineering.

In 2008, HP announced a successful development of a very compact and non-volatile nano-scale memory switches called memristors in the cross bar memory [4]. It was originally postulated by Chua [5], [6] as the fourth basic circuit elements in electrical circuits. It is based on the nonlinear characteristics of the charge and flux. By supplying a voltage or current to a memristor, its resistance can be altered depending on the past history of input current or voltages, which enables this devices to be a non-volatile memory.

Recent researches demonstrated that memristors can be utilized as artificial synapses of neural networks [7]-[9]. Cantley et al. [7], presented a suitable application of memristor synapse in Hebbian learning in spiking neural network and Snider showed a memristor-based self organized network employing dedicated connections with inhibitory (negative) weighting [8]. To be able to build general neural networks, signed weights are needed. Kim et al. [9], presented a bridge circuit employing five memristors toward this goal. However, it is difficult to program very small weights which are near to zero with this circuit. Also, the programmed weight is a nonlinear function of input signal.

In this paper, the memristor-based synapse employing four memristors in a bridge type configuration is proposed, where zero as well as signed weights can be programmed easily. Its weight programming is a linear function of the input signal.

II. HP MEMRISTOR MODEL

The HP TiO_2 memristor [4], is sandwiched between an undoped region with highly resistive TiO_2 and doped region with highly conductive oxygen vacancies TiO_{2-x} between two platinum electrodes. When a voltage or current is applied to the device, the dividing line between the doped and undoped layers shifts with a function of the applied voltage or current. In consequence, the resistance between the two electrodes is altered.

Let the thickness of the doped area be *w*, D be the thickness of the two layers of TiO_2 memristor and let R_{ON} and R_{OFF} denote the low resistance and the high resistance values, respectively. Then the memristance *M(t)* of the TiO_2 memristor model is given by,

$$M(t) = R_{ON}x(t) + R_{OFF}\left(1 - x(t)\right), \tag{1}$$

where $x(t) = \dfrac{w(t)}{D}$ is called the state variable of memristor.

The relation between the voltage *v(t),* and current *i(t)* of memristor is given by,

$$v(t) = M(t)\,i(t). \tag{2}$$

The state variable x(t) is defined as,

$$\frac{dx(t)}{dt} = \mu_V \frac{R_{ON}}{D^2} i(t), \tag{3}$$

where μ_v is the dopant mobility. Since the velocity of the width is linearly proportional to the current, the model is called a linear drift model.

This work was supported by the second stage of Brain Korea 21 Project in 2012

978-1-4673-0289-0/12/$31.00 ©2012 IEEE

Integrating (3), the state variable of memristor with respect to charge *q(t)* is achieved as,

$$x(t) = \mu_V \frac{R_{ON}}{D^2} q(t) + x_0, \tag{4}$$

where $x_0 = \frac{w_o}{D}$ is the initial state of memristor.

Since, $M = \frac{d\varphi}{dq}$, from (1) and (4), the relation between flux $\varphi(t)$ and charge $q(t)$ is given by,

$$\varphi(t) = R_{OFF} q(t) - \left(\frac{q^2(t)}{2Q_0} + x_0 q(t) \right) \Delta R, \tag{5}$$

where $Q_0 = \frac{D^2}{\mu_V R_{ON}}$.

Solving the quadratic and finding the value of *q(t),*

$$q(t) = \frac{Q_0 M_0}{\Delta R} \left[1 - \sqrt{1 - \frac{2\Delta R \varphi(t)}{Q_0 M_0^2}} \right], \tag{6}$$

where, $\Delta R = R_{OFF} - R_{ON}$, and $M_0 = R_{OFF} - x_0 \Delta R$ (the initial memristance of the memristor).

Plugging (6) into (4), yields,

$$x(t) = \frac{M_0}{\Delta R} \left[1 - \sqrt{1 - \frac{2\Delta R \varphi(t)}{Q_0 M_0^2}} \right] + x_0. \tag{7}$$

From (1), and (7), the memristance of the memristor is given as,

$$M(t) = M_0 \sqrt{1 - \frac{2\Delta R \varphi(t)}{Q_0 M_0^2}}. \tag{8}$$

III. PROPOSED MEMRISTOR BRIDGE SYNAPTIC CIRCUIT

A. Memristor-based Weighting Circuit

The proposed memristor-based synaptic weighting circuit is a bridge-like circuit consisting of four identical memristors with the polarities indicated in Fig. 1. Input signal is applied from the left end of the circuit and the output is taken from two middle nodes as a differential form. When a positive or a negative, strong pulse v_{in} is applied at the input terminal, the memristance of each memristor is increased or decreased depending upon its polarity. Applying a positive pulse, the memristances of M_1 and M_4 (whose polarities are forward-biased) will decrease and the memristances of M_2 and M_3 (whose polarities are reverse-biased) will increase. It follows that the voltage v_A at node A (with respect to ground) becomes smaller than the voltage v_B at node B. Since, the node voltage v_A is less than v_B, the output voltage (v_{out}) across the bridge is negative.

If v_{in} be input signal applied to the memristor bridge circuit in Fig. 1 at time *t*. The input voltage will be divided via the well-known "*voltage-divider formula*" as follows,

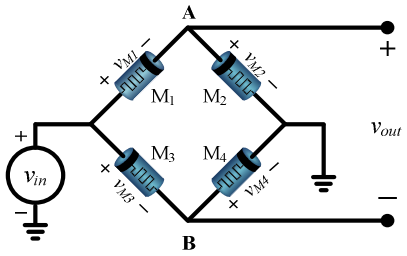


Figure 1. Memristor bridge circuit. The synaptic weight is programmable by varying the input voltage v_{in}. The weighting (multiplication) of the input signal is also performed in this circuit.

$$v_{M2} = \frac{M_2}{M_1 + M_2} v_{in} = v_A, \tag{9}$$

$$v_{M4} = \frac{M_4}{M_3 + M_4} v_{in} = v_B, \tag{10}$$

where M_1, M_2, M_3, and M_4 denote the corresponding memristances of the memristors at time *t*, in Fig. 1. Note that the voltage divider formula for memristors is the same as that for resistors.

The output voltage v_{out} of the memristor bridge circuit is equal to the voltage difference between terminal A and terminal B; namely,

$$v_{out} = v_A - v_B = \left(\frac{M_2}{M_1 + M_2} - \frac{M_4}{M_3 + M_4} \right) v_{in}. \tag{11}$$

From (11), the voltage v_{out} is given as,

$$v_{out} = \psi \times v_{in}, \tag{12}$$

where $\psi = \frac{M_2}{M_1 + M_2} - \frac{M_4}{M_3 + M_4}$ represents the synaptic weight of the memristor bridge circuit.

It follows that if the synaptic weight ψ is larger than 0, namely,

$$\frac{M_2}{M_1 + M_2} - \frac{M_4}{M_3 + M_4} > 0. \tag{13}$$

Our memristor bridge circuit represents as a *positive* synaptic weight. Rearranging (13), the condition for positive, negative and zero synaptic weight is as follow,

$$\psi = \begin{cases} Positive & if \quad \frac{M_2}{M_1} > \frac{M_4}{M_3} \\ Negative & if \quad \frac{M_2}{M_1} < \frac{M_4}{M_3} \\ 0 & if \quad \frac{M_2}{M_1} = \frac{M_4}{M_3} \end{cases}. \tag{14}$$

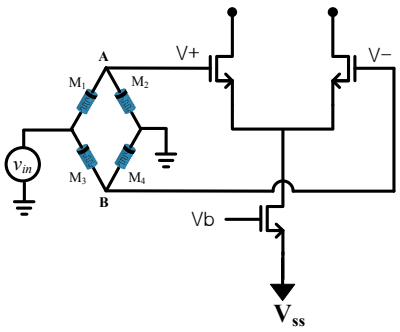


Figure 2. Memristor bridge synaptic circuit.

The state when the output is zero is henceforth called the *balanced state*. At the balanced state, the synaptic weight is zero. Note that the synaptic weight programming signal and the synaptic input signal v_{in} for processing share the same input terminal in Fig. 1. The two different kinds of signals are discriminated by being assigned at different time slots. Very narrow pulses with negligible effect on memristance changes are used for synaptic multiplication, while very strong pulses are used for programming synaptic weights.

B. *Memristor-based Neuron Circuit*

Each neuron has to add a set of weighted input signals from diverse signal sources. This is implemented in our memristor bridge neuron via summing input signals with current mode circuits. The differential amplifier with three transistors in Fig. 2 is the voltage-to-current converter which provides interface to the neuron for summing input signals easily.

Fig. 3 shows the complete circuit of our memristor-bridge neuron with multiple input synapses. All the positive terminals of the input synapses are connected together, as are the negative terminals. The circuit at the bottom right of Fig. 3 is the cell biasing circuit that provides the DC bias voltage for the output.

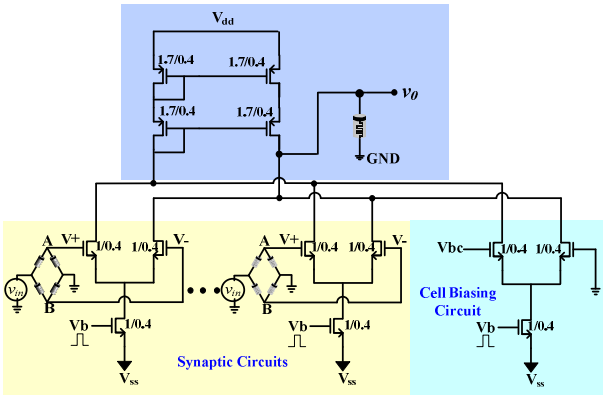


Figure 3. Complete schematic of our memristor-bridge neuron. The synaptic circuits are located at the left bottom and the cell biasing circuit is at the right bottom. The current mirror at the top acts as an active load that is shared by all synapses.

The differential amplifier shown in Fig. 3 is used for voltage to current converters in which each neuron cells add a set of weighted input signals from diverse sources with current mode circuit. The active load at the top of Fig. 3 is shared among all input synaptic circuits. The sum of all synaptic input signal currents appears at the output and, is converted back to a *voltage* signal via a memristor load circuit. The transistor circuit as well as the memristor circuit operates based on voltage pulses, which enables the circuit to save power greatly.

Our memristor-bridge neuron can be used for any kind of neural networks, including Cellular Neural Networks (CNN).

IV. SIMULATIONS

Our memristor bridge architecture is simulated using HP TiO_2 memristor model. The basic parameters for the simulations are based on the data given by HP [4], where R_{ON}=100 Ω, R_{OFF} =16 *KΩ*, D=*10 nm*, and $\mu_v = 10^{-14} m^2 V^{-1} S^{-1}$.

In the weight-sign setting, a strong wide pulse is applied to change the state of memristor and a very narrow pulse for synaptic multiplication is applied to avoid the drifting the memristor state. The weight setting of the architecture is verified through computer simulations and synaptic multiplication, memristor synapse-based neural circuit is performed in HSPICE with replacement of identical resistance of the memristor programming weight.

A. *Synaptic Weight Programming in Memristor Bridge Circuit*

In the weight-sign setting, a strong wide pulse 1V amplitude is applied to change the state of memristor. Fig. 4(a) shows the changes in the memristances $M_1(t)$, $M_2(t)$, $M_3(t)$ and $M_4(t)$ as a function of time, obtained via computer

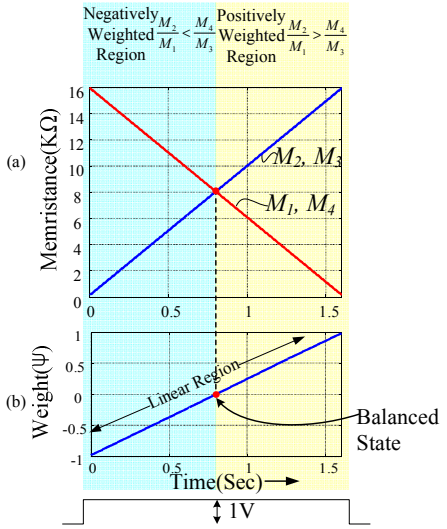


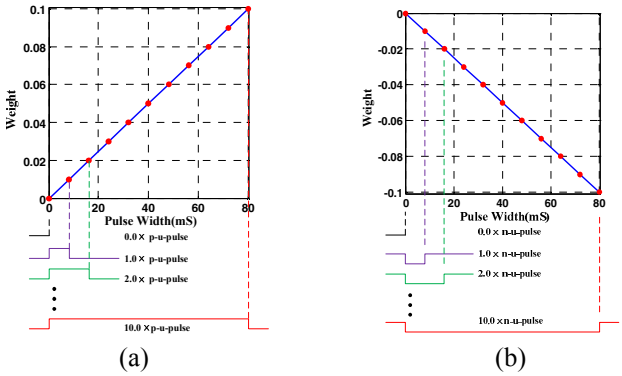
Figure 4. Time variations of $M_1(t)$, $M_2(t)$, $M_3(t)$, $M_4(t)$, and $\psi(t)$ with our memristor bridge circuit when a wide pulse is applied. The linear memristor model is assumed. The initial memristances are $M_1(0) = M_4(0) = 16$ kΩ, and $M_2(0) = M_3(0) = 100$ Ω. (a) $M_1(t)$, $M_2(t)$, $M_3(t)$, and $M_4(t)$. (b) Weight $\psi(t)$.

simulations of the memristor bridge circuit in Fig. 1 with initial memristances, $M_1(0) = M_4(0) = 16\ K\Omega$, $M_2(0) = M_3(0) = 100\ K\Omega$. Fig. 4(b) is the corresponding weight computed with the memristance values in Fig. 4(a) using (12). As shown in the Figs., these numerically computed memristances in Fig. 4(a) and the corresponding weight in Fig. 4(b) are very linear. The linearity of synaptic weight programming in the memristor bridge comes from the complementary action of the back-to-back memristors in our memristor bridge circuit.

In order to investigate the linearity of memristor weight, any desired synaptic weight can be set on each memristor by applying either a single wide pulse, or several successive narrow pulses. The synaptic weight was computed by measuring the output voltage of our memristor bridge circuit for a known input voltage, as described in section III. Each simulation was assumed to start from the balanced state where the synaptic weight is zero. Fig. 5(a) and Fig. 5(b) show the weight programming results for positive and negative unit pulse [1V, 8mS] and [-1V, 8mS], respectively. The Figs. show that weight increases or decreases almost *linearly,* depending upon the sign and the widths of the applied pulses.

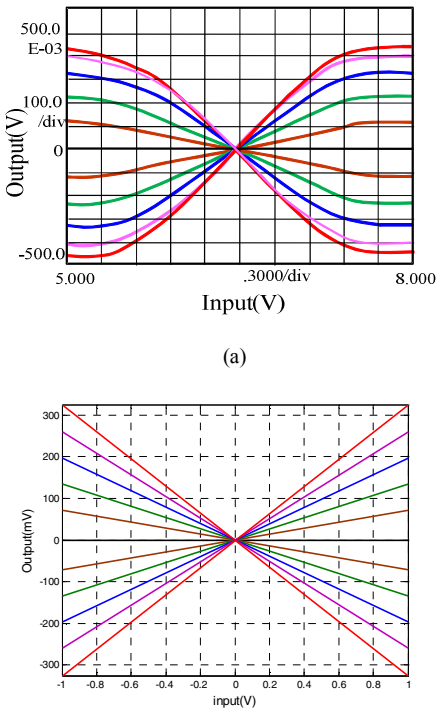
B. Synaptic Multiplication

Simulations of the synaptic weight processing were performed in our memristor-bridge with input voltage 5mS pulse width at [-1, 1]V amplitude and synaptic weighting Ψ=[-0.1, 0.1]. Fig. 6(b) shows the linearity of the relationship between the input voltages, and the memristor bridge neuron output voltage, for various synaptic weights of the proposed circuit. Fig. 6(a) is the performance of a conventional synaptic circuit employed in the programmable analog vector matrix multiplication and CNN [1]-[3]. Note that the linearity of our memristor-based synaptic weighting is superior to that of the conventional multiplication circuit. The linearity of the memristor bridge synaptic circuit comes from the linear assignment of voltage and weight at the memristor bridge circuit.



(a) (b)

Figure 5. Programmed weights with different pulse widths when (a) positive and (b) negative pulses were applied. Simulations start from the balanced state in which the weight is zero.



(a)

(b)

Figure 6. Synaptic multiplication with (a) Gilbert multiplier–based circuit [1]-[3], and (b) memristor based bridge circuit.

C. CNN Application

The memristor bridge neuron was used to implement a two-dimensional (2-D) image processing task. The basic architecture [2] of a feed forward CNN with zero **A** template was designed using our memristor bridge synapses as template elements. The circuit has nine synaptic inputs representing the **B** template and one bias voltage representing the threshold. The *average* template, and the *Laplacian* template, as shown in Fig. 7, were tested with our memristor bridge neurons. Note that Z is a bias template of the CNN and zero for the both cases. For the *average* template, the circuit of the CNN cells was simulated via SPICE, and the simulation with the input image (size:16x16) in Fig. 8(a) was performed. The images shown in Figs. 8(b) and (c) are the results obtained by using the CNN software, and by using memristor bridge synapses, respectively. Also, our simulation of the *Laplacian* template was performed using the input image (size: 59x59) in Fig. 9 (a). The images shown in Figs. 9(b) and (c) are results obtained by using the CNN software, and by using memristor bridge synapses, respectively. Although image processing using circuits in general has poorer performances than that obtained from CNN software, the results obtained from memristor bridge synapse is very close to the result obtained from CNN software: the average pixel difference between the results with the CNN simulator and that of the proposed circuit 0.89 % and 0.86 % for Fig. 8 and Fig. 9, respectively.

	0	1	0			-1	-1	-1	
B=	1	2	1	**A**=0	**B=**	-1	8	-1	**A**=0
	0	1	0	Z=0		-1	-1	-1	Z=0
	(a)					(b)			

Figure 7. CNN templates tested for image processing with memristor bridge synapses. (a) Average template. (b) Laplacian template.

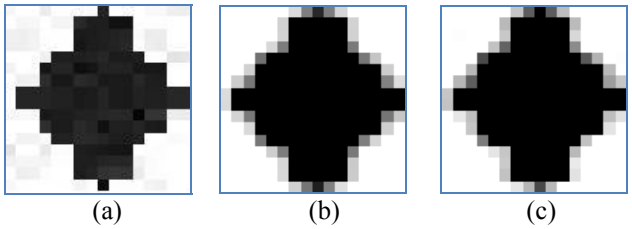


Figure 8. Comparison of image processing performance using the average template. (a) Input image. (b) Processing implemented with CNN software. (c) Processing implemented with memristor bridge synapses representing B template.

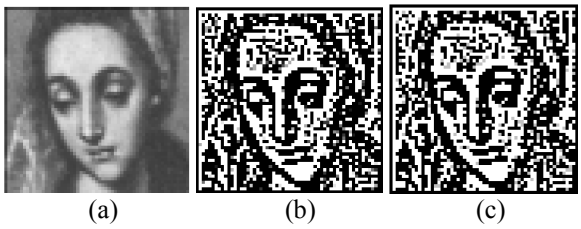


Figure 9. Comparison of image processing performance using the Laplacian template. (a) Input image. (b) Processing implemented with CNN software. (c) Processing with memristor bridge synapses representing the B-template.

D. Features of the Proposed Circuit

In this subsection, the features of the proposed circuit are discussed. Since, memristor is in several nano-scale [4], the size of memristor bridge synaptic circuit is much smaller than that of conventional weighting circuit [1]-[3]. The number of transistors required for the memristor based synaptic circuit is 3 while that of Gilbert multiplier-based synaptic circuit is 7. Considering the fact that the size sum of 4 memristors with the proposed circuit is less than that of a single transistor, the size benefit of the proposed synaptic circuit is obvious. Additionally benefit is the non-volatility, while that of the conventional circuit is volatile, which requires for the conventional circuit to be refreshed from time to time, but it is not necessary in the proposed circuit. Also, linearity in synaptic operation of the proposed circuit is superior to that of conventional multiplier.

V. CONCLUSION

We presented a pulse based memristor synapse circuit which is able to perform signed synaptic weighting. Weights are programmed with strong pulses and synaptic multiplications are performed with weak pulses through a single input line.

Simulations on synaptic multiplications have been performed with HSPICE in the weighting range of ξ= [-0.1, 0.1]. The synaptic multiplication between programmed weights and input signal showed an excellent linearity compared to the conventional Gilbert multiplier-based circuit.

The performance of memristor bridge synapses was investigated via two CNN templates (*average template* and *Laplacian template*). Circuit simulations using SPICE showed processing results that were very close to the ideal result obtained via software processing. Thus, the proposed bridge synaptic circuit is well applicable in programmable analog vector matrix multiplication, CNN and neural network weighting circuit.

REFERENCES

[1] F. J. Kub, K. K. Moon, I. A. Mack, and F. M. Long, "Programmable analog vector-matrix multipliers," IEEE journal of Solid-State Circuits, vol. 25, no. 1, pp. 207-214, 1990.

[2] R. Domíinguez-Castro, et. al., "A 0.8-μm CMOS two-dimensional programmable mixed-signal focal-plane array processor with on-chip binary imaging and instructions storage," IEEE J. of Solid-State Circuits, vol. 32, no. 7, pp. 1013-1026, 1997.

[3] J. M. Cruz and L. O. Chua, "A 16x16 cellular neural network universal chip: the first complete single-chip dynamic computer array with distributed memory and with gray-scale input-output," Analog Integrated Circuits and Signal Processing, vol. 15, no. 3, pp. 227-237, 1998.

[4] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," Nature, vol. 453, pp. 80-83, 2008.

[5] L. O. Chua, "Memristor-the missing circuit element," IEEE Trans. Circuit Theory, vol. 18, no. 5, pp. 507-519, 1971.

[6] L. O. Chua and S. M. Kang, "Memristive devices and systems," Proc. of IEEE , vol. 64, no. 2, pp. 209-223, 1976.

[7] K. D. Cantley, A. Subramaniam, H. J. Stiegler , R. A., Chapman, and E. M. Vogel, "Hebbian learning in spiking neural networks with nanocrystalline silicon TFTs and memristive synapse," IEEE Trans. on Nanotechnology, vol. 10, no. 5, pp. 1066-1073, 2011.

[8] G. S. Snider, "Self-organized computation with unreliable, memristive nanodevices," Nanotechnology**,** vol. 18, no. 36, pp. 1-13, 2007.

[9] H. Kim, M. P. Sah, C. Yang, T. Roska, and L. O. Chua, "Neural synaptic weighting with a pulse-based memristor circuit., IEEE Trans. on Circuit and Systems-I , vol. 59, no.1, pp. 148-158, 2012.