Sensors **2012**, *12*, 3587-3604; doi:10.3390/s120303587

OPEN ACCESS

sensors

ISSN 1424-8220

www.mdpi.com/journal/sensors

Article

A Voltage Mode Memristor Bridge Synaptic Circuit with Memristor Emulators

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

[1] Division of Electronics and Information Engineering, Chonbuk National University, Jeonju 561-756, Korea; E-Mails: maheshwarsah@hotmail.com (M.P.S.); ychangju@jbnu.ac.kr (C.Y.)

[2] Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, USA; E-Mail: chua@eecs.berkeley.edu

* Author to whom correspondence should be addressed; E-Mail: hskim@jbnu.ac.kr; Tel.: +82-63-270-2477; Fax: +82-63-270-3988.

Received: 21 January 2012; in revised form: 12 February 2012 / Accepted: 7 March 2012 / Published: 14 March 2012

Abstract: A memristor bridge neural circuit which is able to perform signed synaptic weighting was proposed in our previous study, where the synaptic operation was verified via software simulation of the mathematical model of the HP memristor. This study is an extension of the previous work advancing toward the circuit implementation where the architecture of the memristor bridge synapse is built with memristor emulator circuits. In addition, a simple neural network which performs both synaptic weighting and summation is built by combining memristor emulators-based synapses and differential amplifier circuits. The feasibility of the memristor bridge neural circuit is verified *via* SPICE simulations.

Keywords: memristor bridge; non-volatile programming weight; neuron; synapse; synaptic multiplication

1. Introduction

Synaptic multiplications between input signals and weights are key operations in neural networks, programmable analog vector matrix multiplication and cellular neural networks. Most of the previous synaptic multiplications are based on the software models [1–4]. While the flexibility of the software-based model is excellent, its processing speed represents a serious bottleneck. The digital

accelerating board on which the software version of neural network is a practical option representing a compromise between limited flexibility and a high speed processing [5,6]. However, this approach may not be the solution for the problem of bigger size of neural networks.

There have been some research efforts to build artificial synapses (weights) in neural network chip and analog programmable vector matrix multiplication using CMOS technologies [7–11]. To implement the immense amount of neural processing on a chip, extremely high density of integration technology is needed. This is a very challenging goal and not many successful cases of neural implementations have been reported so far. The cellular neural network [12–16] is one of the successful implementations of analog multiplication circuits.

Most of the synaptic weights implemented with the conventional technologies are volatile. Also, synaptic multiplication between input signal and weight is non-linear. Therefore, introducing a new weighting technology which is nonvolatile and linear is very important for the further development of neuromorphic engineering.

In 2008, HP announced a successful fabrication of a very compact and non-volatile nano scale memory called the memristor [17]. It was originally postulated by Chua [18,19] as the fourth basic circuit elements in electrical circuits. It is based on the nonlinear characteristics of charge and flux. By supplying a voltage or current to the memristor, its resistance can be altered. In this way, the memristor remembers information.

Many of recent researches showed the great potential of memristors in the application of memory, and artificial synapses [20–24]. Cantley *et al.* presented an application of memristor synapse for the Hebbian learning in spiking neural network [21]. Snider demonstrated a memristor-based self organized network employing dedicated connections for inhibitory (negative) weighting [22]. For such application in neural network or cellular neural network, every connection has to be weighted either positively or negatively.

In [24], we demonstrated the architecture of the memristor bridge circuit which is able to perform signed synaptic operations. The study was conducted with the mathematical model of the HP memristor, where the operation of the memristor bridge circuit was verified via software simulation. This study is an extension of the previous research advancing toward the circuit implementation where the architecture of the memristor bridge neuron is built with our memristor emulator circuits [25]. Also, a simple neural network which performs both synaptic weighting and summation is built by combining memristor emulators-based synapses and differential amplifier circuits.

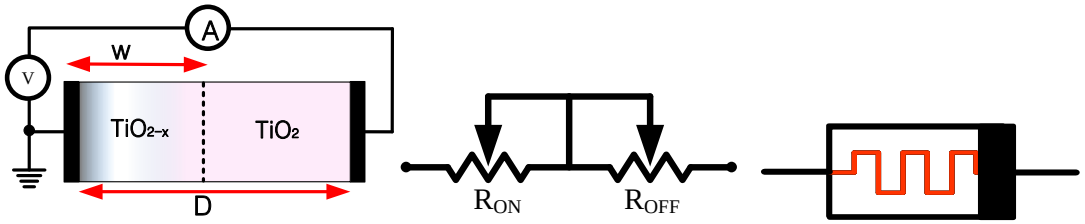
In this paper, the HP TiO_2 memristor model is introduced in Section 2. In Section 3, a memristor emulator circuit is proposed. Memristor bridge synapses built with memristor emulator circuits are described in Section 4. Simulation results are presented in Section 5. In Section 6 we present our conclusions.

2. HP Memristor Models

In HP TiO_2 memristor model [17], an undoped region with highly resistive TiO_2 and doped region with highly conductive oxygen vacancies TiO_{2-x} layer are sandwiched between two platinum electrodes as shown in Figure 1(a). When a voltage or current signal is applied to the device, the border line between the doped and undoped layers shifts as a function of the applied voltage or current.

In consequence, the resistance between the two electrodes is altered. Figure 1(b,c) is the equivalent circuit and the symbol whose polarity is indicated by a black bar at one end. The defined polarity indicates that the memristance is decreased (or increased) when current flows from the left (right) side to the right (left) side of the memristor symbol in Figure 1(c).

Figure 1. (**a**) Structure of TiO_2 memristor, TiO_{2-x} and TiO_2 layers are sandwiched between two platinum electrodes. When a voltage/current is applied, its memristance (resistance of the memristor) is altered; (**b**) equivalent circuit and (**c**) symbol of the memristor.



(**a**) (**b**) (**c**)

Let w be the thickness of the doped area, D be the thickness of the two layers of TiO_2 memristor. Let R_{ON} and R_{OFF} denote the minimum resistance and the maximum resistance values, respectively.

Then, the relation between the voltage and the current is given by:

$$v(t) = \left(R_{ON} \frac{w(t)}{D} + R_{OFF} \left(1 - \frac{w(t)}{D} \right) \right) i(t) \tag{1}$$

where memristance $M(t) = R_{ON} \frac{w(t)}{D} + R_{OFF} \left(1 - \frac{w(t)}{D} \right)$ and $w(t)/D$ is defined as the state variable. In the TiO_2 memristor [17], the rate of change of the state variable is defined as a function of current i; namely:

$$\frac{dw(t)}{dt} = \mu_V \frac{R_{ON}}{D} i(t) \tag{2}$$

where μ_v is the dopant mobility. This model is called a linear drift model, since the velocity of the width is linearly proportional to the current. Integrating Equation (2):

$$w(t) = w_0 + \mu_V \frac{R_{ON}}{D} \int_0^t i(t)dt = w_0 + \mu_V \frac{R_{ON}}{D} q(t). \tag{3}$$

From Equations (1) and (3), the memristance $M(t)$ can be written as:

$$M(t) = R_{OFF} \left\{ \left[1 + \frac{w_0}{D} \left(\frac{R_{ON}}{R_{OFF}} - 1 \right) \right] - \frac{\mu_V R_{ON}}{D^2} \left(1 - \frac{R_{ON}}{R_{OFF}} \right) q(t) \right\} \tag{4}$$

If $w_0/D << 1$ *and* $R_{ON} << R_{OFF}$ the expression of $M(t)$ is simplified as :

$$M(t) \approx R_{OFF} \left\{ 1 - \frac{\mu_V R_{ON}}{D^2} q(t) \right\}. \tag{5}$$

$M(t) = R_{OFF} - Kq(t)$, where $K = \frac{\mu_v R_{ON}}{D^2 R_{OFF}}$.

From Equation (1):

$$v(t) = \left(R_{OFF} - Kq(t)\right) i(t). \tag{6}$$

It follows from Equation (6) that the memristance $M(t)$ decreases when higher voltage is applied to the non-black bar side than that of black bar side in Figure 1(c). Similarly, the memristor is called incrementally biased when a higher voltage is applied at the black bar side than that of non-black bar side in Figure 1(c). With this bias, the current-voltage relationship is given by:

$$v(t) = \left(R_0 + Kq(t)\right) i(t) \tag{7}$$

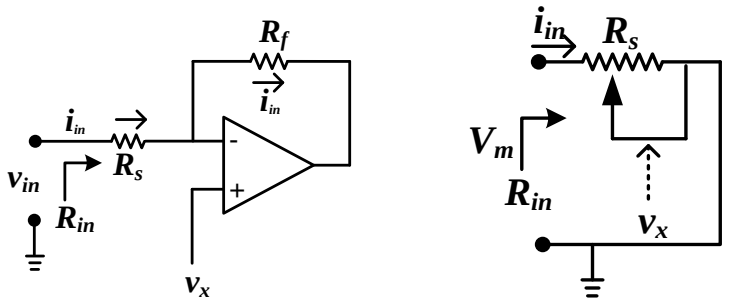
and the memristance $M(t)$ increases as $M(t) = R_o + Kq(t).$

Detailed descriptions of incremental and decremental memristors using our emulators circuits are provided in Section 3.

3. HP Memristor Emulator Circuit

As of today, memristors are not yet available on the market. In order to study memristor-based circuit, building memristor emulators is necessary. Two different approaches to build the memristor emulators are the pure analog circuit-based [25] and the analog-digital mixed-based [26,27]. The memristor emulator circuit adopted for this work is from [25]. The basic idea implemented to design the memristor emulator [25] is shown in Figure 2.

Figure 2. Basic concept for implementing the memristor emulator (**a**) input resistance as a function of voltage v_x; (**b**) equivalent circuit.



(a) **(b)**

In the figure, the voltage at the input terminal is,

$$v_{in} = R_s i_{in} + v_x \tag{8}$$

where i_m is the input current, R_s is a resistance at the inverting input terminal and v_x is the voltage applied to the positive terminal of the op Amp.

Assume that the voltage v_x is proportional to input current i_{in}, then:

$$v_{in} = R_s i_{in} + m i_{in} = \left(R_s + m\right) i_{in} \tag{9}$$

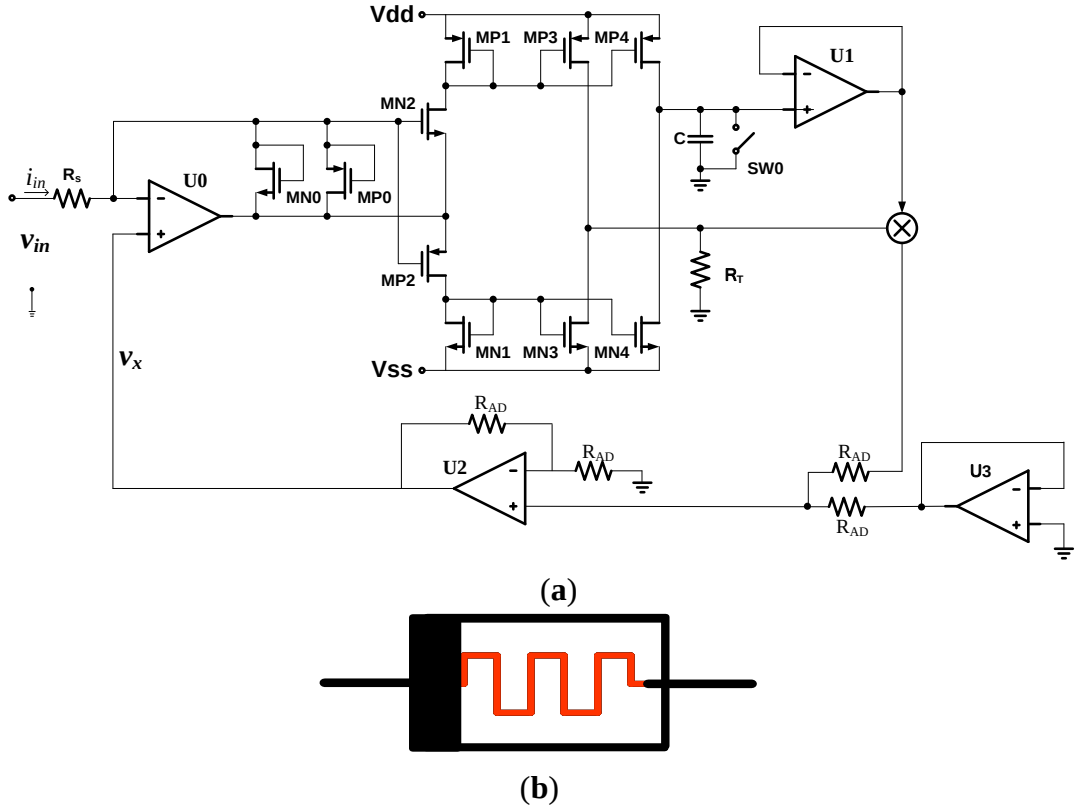
where m is a proportionality coefficient and $v_x = mi_{in}$. Equation (9) implies that the input resistance of the circuit is $R_s + m$. If we can control m so that, it is time integral of the input current i_{in}, then, the circuit in Figure 2 acts as a memristor.

To emulate v_x in Equation (9), three devices (a capacitor, a resistor, and a voltage multiplier) are utilized, in which the voltage from the capacitor and that from the resistor are multiplied using a voltage multiplier.

The memristor emulator needs to be prepared in two different connections such as decremental and incremental emulators, separately.

Figure 3 shows the schematic of the incrementally biased memristor emulator where memristance increases when a positive voltage v_{in} applied at the input terminal. The input voltage applied at a memristor emulator is converted into an input current i_{in} with a resistor R_s and op Amp U0 via the virtual ground constraint. Since the current i_{in} is used at several places, its replicas are generated using current mirrors. Observe that a current mirror copies single directional current only. For bi-directional (positive and negative) currents, i_{in} must be separated into a positive part and a negative part and processed separately at different parts of the circuit. In the circuit of Figure 3, the positive part of the current, duplicated by a current mirror MN0 and MN2 is fed into a resistor R_T and a capacitor C by current mirror MP3 and MP4 with couple of MP1 respectively. On the other hand, MP0 and MP2 acts as the negative part of current mirror that flows out from resistor R_T and capacitor C by current mirror MN3 and MN4 which are coupled with MN1.

Figure 3. incrementally-biased memristor emulator circuit (**a**) memristor emulator circuit; (**b**) a schematic of memristor emulator.



One of the distinguished features of a memristor is the capability of keeping the programmed information for a long time until new programming inputs are presented. The charge stored at capacitor C is for the programmed information in the memristor emulator. To avoid discharging during the period when an input signal does not exist, the path to the output terminal is connected to a Mosfet buffer U1. The switch S_{W0} is initially closed to reset the capacitor voltage to zero. When a voltage pulse is applied through the input terminal of the emulator circuit, the switch is opened. Therefore the capacitor voltage starts to charge from zero voltage to certain level.

In Figure 3, the capacitor produces a voltage v_C by integrating the current i_{in}, and the resistor R_T produces a voltage proportional to the current i_{in}:

$$v_C = \frac{1}{C}\int i_{in}dt = \frac{q_C}{C}, \tag{10}$$

and:

$$v_R = R_T \times i_{in}. \tag{11}$$

These two voltages are multiplied by a voltage multiplier. The output voltage v_x of the voltage multiplier is given by:

$$v_x = \frac{q_C}{C} \times R_T i_{in}. \tag{12}$$

Therefore, the input voltage v_{in} is:

$$v_{in} = \left(R_s + \frac{q_C}{C} \times R_T \right) i_{in}, \tag{13}$$

where the memristance $M(t)$ is:

$$M(t) = \left(R_s + \frac{q_C}{C} \times R_T \right). \tag{14}$$

From Equation (14), when a positive pulse is applied at the input terminal, the resistance increases proportional to the time integral of input current with R_s, we call this configuration the incrementally biased memristor which corresponds to the voltage state where the higher voltage is applied at the black bar side of Figure 1(c).

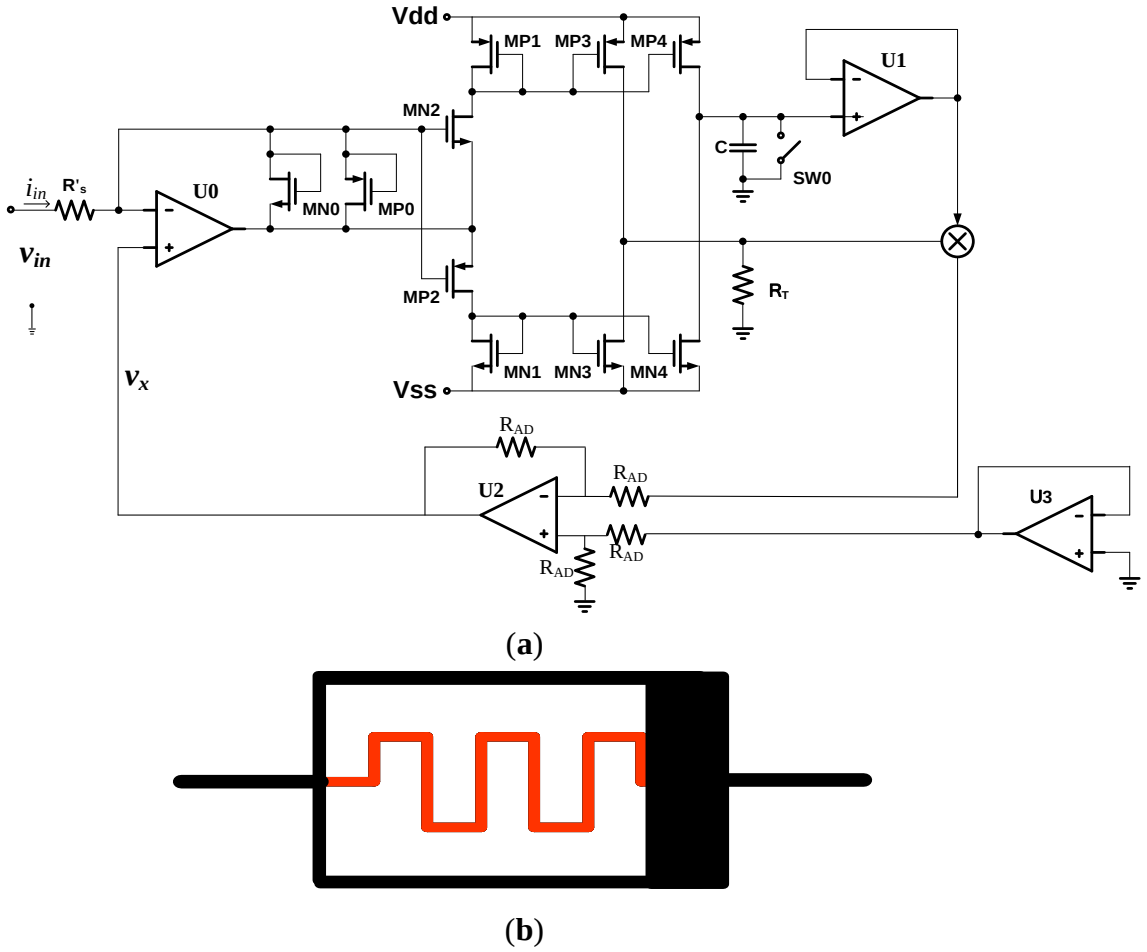
On the contrary, if a higher voltage is applied to the non-black bar side, then, the memristance is decreased. We call this configuration the decrementally biased memristor. By adding a voltage inverter after the voltage multiplier as shown in Figure 4, the decrementally biased memristor can be implemented. The input voltage in the decrementally biased memristor is given by:

$$v_{in} = \left(R_s^{'} - \frac{q_C}{C} \times R_T \right) i_{in}.$$

The resultant memristance $M(t)$ of the decremental memristor is:

$$M(t) = R_s^{'} - \frac{R_T}{C} q(t) = R_s^{'} \left(1 - \frac{R_T}{C R_s^{'}} q(t) \right). \tag{15}$$

Figure 4. Decrementally-biased memristor emulator circuit (**a**) memristor emulator circuit; (**b**) a schematic of memristor emulator.



4. Memristor Neural Circuit Built with Memristor Emulators

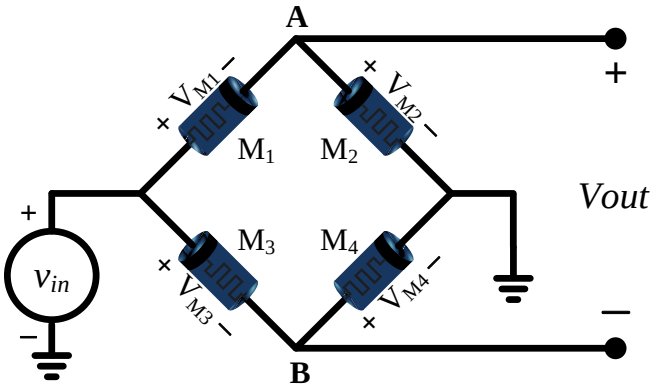
The memristor bridge synapse circuit [24] is composed of four memristors as shown in Figure 5. In this study, the architecture of the memristor bridge synapse is built with memristor emulator circuits.

4.1. The Memristor Bridge Synapse

When a positive or negative strong pulse v_{in} is applied at the input terminal of the memristor bridge synapse in Figure 5, the memristance of each memristor is increased or decreased depending upon its polarity.

When a positive pulse is applied at input terminal of Figure 5, the memristances of M_1 and M_4 (which are decrementally-biased) decrease. On the other hand, the memristances of M_2 and M_3 (which are incrementally-biased) will increase. It follows that the voltage v_A at node A (with respect to ground) increases while the voltage v_B at node B decreases. If the pulse width is wide enough, the output voltage V_{out} varies gradually from negative to positive voltage.

Figure 5. Memristor based synaptic circuit in [24]. It is assumed that M_1 and M_4 are decrementally biased memristor while M_2 and M_3 are incrementally biased memristors.



On the other hand, if a negative pulse is applied, when M_1 and M_4 are minimum and M_2 and M_3 are are their maximum state respectively, then, M_1 and M_4 vary to higher memristance and M_2 and M_3 go to lower value. It follows that the output voltage V_{out} varies gradually from positive to negative voltage. In consequence, the weight is able to be programmed with any weights in the range from −1 to +1 including zero using appropriate duration of pulse.

Let v_{in} be the input voltage pulse. Also, let V_{M1}, V_{M2}, V_{M3}, and V_{M4} be the voltages across memristor M_1, M_2, M_3, and M_4 respectively. Then the voltage at each memristor at time t is:

$$v_{M1} = \frac{M_1}{M_1 + M_2} v_{in}, \tag{16}$$

$$v_{M2} = \frac{M_2}{M_1 + M_2} v_{in} = v_A, \tag{17}$$

$$v_{M3} = \frac{M_3}{M_3 + M_4} v_{in}, \tag{18}$$

$$v_{M4} = \frac{M_4}{M_3 + M_4} v_{in} = v_B, \tag{19}$$

where M_1, M_2, M_3, and M_4 denote the corresponding memristance values of the memristors at time t, as in Figure 5.

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

$$V_{out} = v_A - v_B = \left(\frac{M_2}{M_1 + M_2} - \frac{M_4}{M_3 + M_4} \right) v_{in} \tag{20}$$

where v_A and v_B corresponds to the voltages v_{M2} and v_{M4}, respectively.

Equation (20) can be rewritten as a relationship:

$$V_{out} = \xi \times v_{in}, \tag{21}$$

where $\xi = \frac{M_2}{M_1 + M_2} - \frac{M_4}{M_3 + M_4}$ represents the synaptic weighting factor of the memristor bridge synapse.

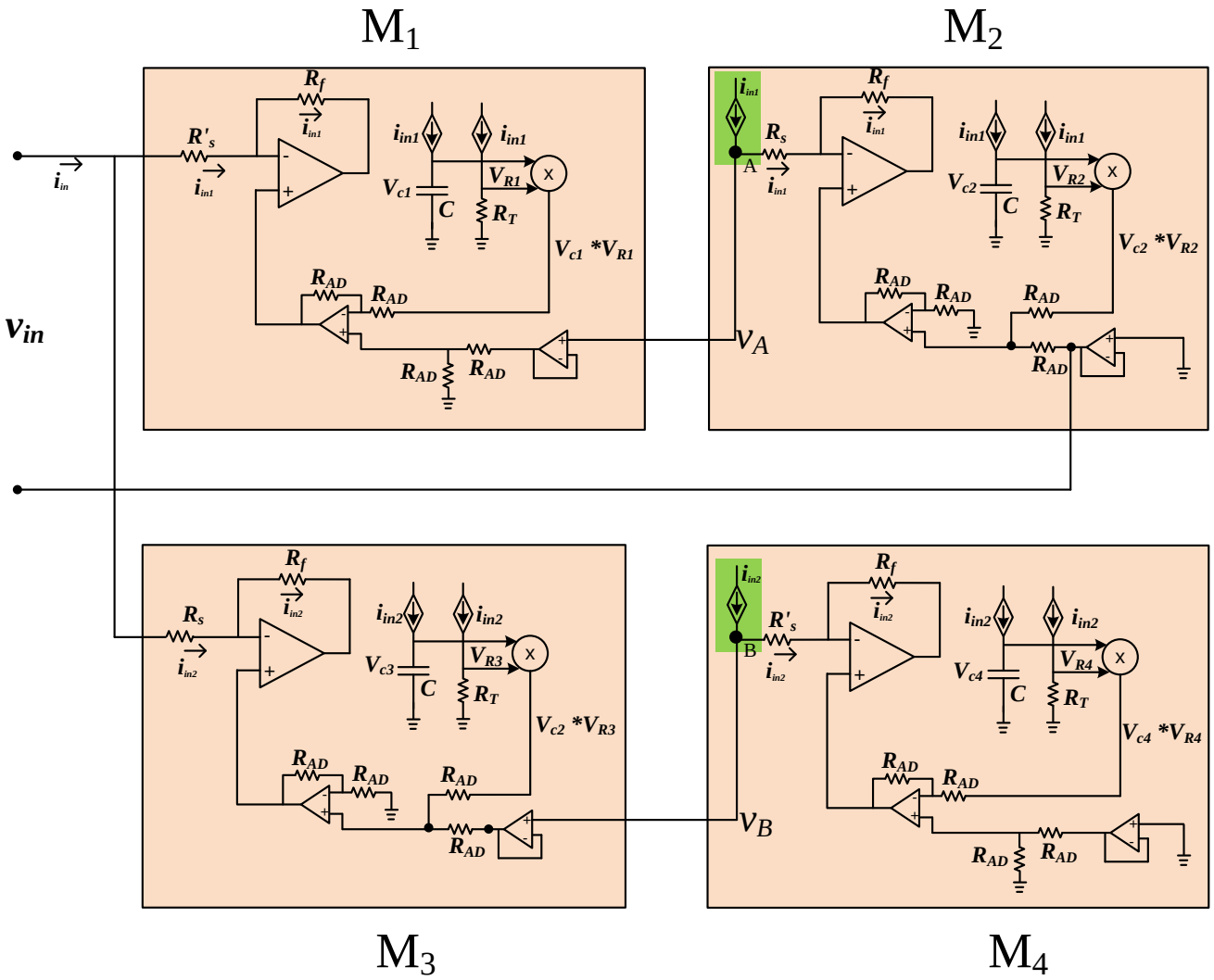
4.2. Memristor Bridge Synaptic Circuit with Memristor Emulators

The memristor bridge circuit in Figure 5 can be built with memristor emulators which are described in Section 3. In the memristor bridge synapse circuit, the serial connection of two memristors M_1 and M_2 are parallel to other serially connected memristors M_3 and M_4.

When a voltage pulse is applied at serially connected memristors, the input voltage is distributed to every memristor according to the voltage law so that the sum of each memristor voltage is equal to the input voltage like in ordinary resistors.

Figure 6 illustrates the memristor bridge synaptic circuit using four memristor emulators. In this architecture, the input current of the first memristor emulator M_1 is replicated by a current mirror and fed to the second memristor emulator M_2 to produce its voltage in the memristor emulator. The voltage produced in the second emulator is added to the first emulator with an analog voltage adder. Therefore, the sum of the individual voltage across each serially connected memristor equals to the input voltage.

Figure 6. Schematics of memristor emulator-based synaptic circuit corresponding to the synaptic structure of Figure 5.



4.3. Synaptic Multiplication

After the weight setting, the synaptic multiplication between input pulse and weight can be performed by applying a pulse with very narrow width. If the weight is set as in Equation (21), the synaptic multiplication (V_{sm}) between input pulse(V_S) and weighting factor (ξ) is:

$$V_{sm} = V_{out} = \xi \times V_s. \tag{22}$$

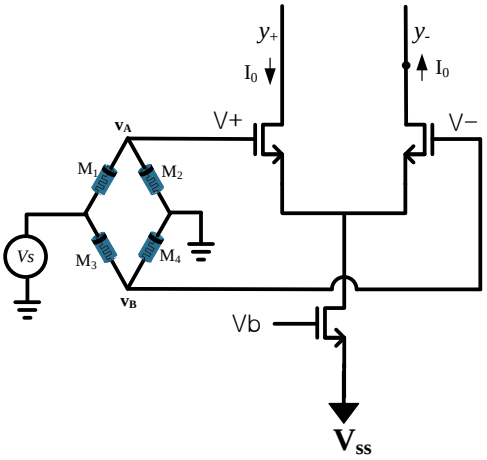
Note that the effect of memristance change is negligible for very narrow pulse signal V_s. Therefore, the weighting factor ξ is constant and output is the linear multiplication between the input pulse and weighting factor ξ. Thus, the memristor bridge circuit acts as a synapse. In case that the memristance change (drift) with weighting operation is really the problem, a doublet circuit can be used to suppress the effect of the memristance change (drift) [28].

The differential amplifier as shown in Figure 7 is used for voltage to current converter. The output current across differential amplifier for input signal V_s is given as:

$$I_0 = \frac{g_m V_{sm}}{2} = \frac{g_m \xi \times V_s}{2} \tag{23}$$

where g_m is the transconductance of Mosfet.

Figure 7. Memristor bridge synaptic circuit. The memristor bridge on the left performs the weighting operation while the differential amplifier on the right performs the voltage to current conversion.



Note that the same input terminal in Figure 7 is shared by the signal v_{in} for synaptic weight programming and the synaptic input signal V_s for weight processing. The two different kinds of signals are discriminated by being assigned at different time slots.

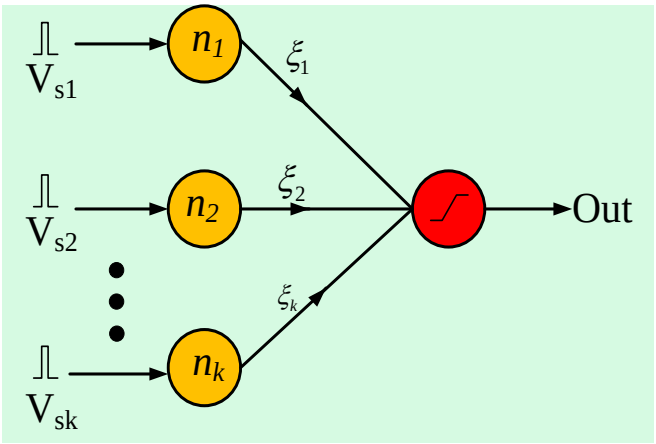
4.4. Memristor Synapse-Based Neural Circuit

The synaptic multiplication in neural network is very important in neuromorphic engineering, programmable analog vector matrix multiplication and CNN circuits [10,11,16].

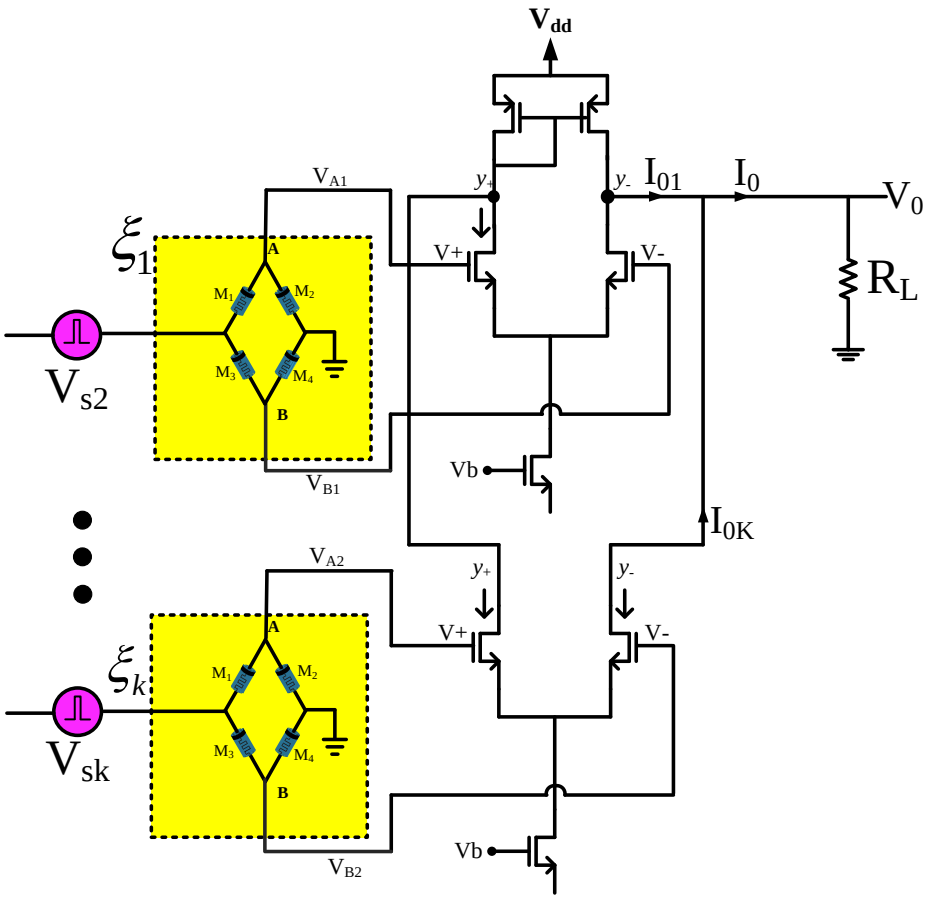
Figure 8(a) is a general single layered neural network. The circuit of the memristor synapse-based neuron using memristor bridge and differential amplifier is shown in Figure 8(b). The synaptic

multiplications among input pulses and memristor-based weights are conducted in the multiple memristor bridge circuits and the results of the multiplications are summed by simply tying the output terminals in a neuron cell. The sum of the currents is then converted back into a voltage using the load circuit R_L.

Figure 8. Neural circuit (**a**) Block diagram of single layer neural network (**b**) Memristor synapse-based neural circuit.



(a)



(b)

The total current (I_0) at the neuron output is:

$$I_0 = I_{01} + I_{02} + I_{03} + ...I_{0k}.$$

where $I_{0k,}$ is the output current across differential amplifier corresponding to input voltage pulse V_{sk} for k th synapse.

The final output voltage across the resistor R_L is given as,

$$V_0 = I_0 \times R_L = \left[I_{01} + I_{02} + ...I_{0k}\right]R_L. \quad (24)$$

From Equations (23) and (24), the output voltage across R_L is,

$$V_0 = \frac{g_m R_L}{2}\left[\xi_1 \times V_{s1} + \xi_2 \times V_{s2} + ...\xi_n \times V_{sk}\right] \quad (25)$$

where ξ_k is the weighting factor of the kth synapse.

Therefore, the output voltage of the neuron is given as:

$$V_0 = \frac{g_m R_L}{2}\sum_{k=1}^{n}\xi_k \times V_{sk}. \quad (26)$$

Equation (26) reveals that, the output voltage at load resistor R_L, is the weighted sum of the product of each input voltage pulse and programming weight.

5. Simulations

In this paper, the memristor bridge architecture [24], is built with memristor emulator circuit. The parameters are chosen as realistic value as possible, so the minimum memristance R_{ON} (R_S) = 100 Ω, and the maximum memristance R_{OFF} (R'_S) = *16 KΩ*, are taken from those of Stanley Williams' real memristor [17]. Also, capacitance C and resistance R_T employed for the memristor emulator are 0.1 μF and R_T = *4 KΩ*, respectively. The architecture has been simulated in PSPICE with input voltage pulse ±1 V and power supply ±5 V.

For the weight programming, strong wide pulses were applied to change the state of memristor and very narrow pulses (3 *ns*) were used for synaptic multiplication. The PSPICE simulations were conducted for the weight programming and synaptic multiplication of the memristor emulator-based bridge synapses.

5.1. Weight Programming

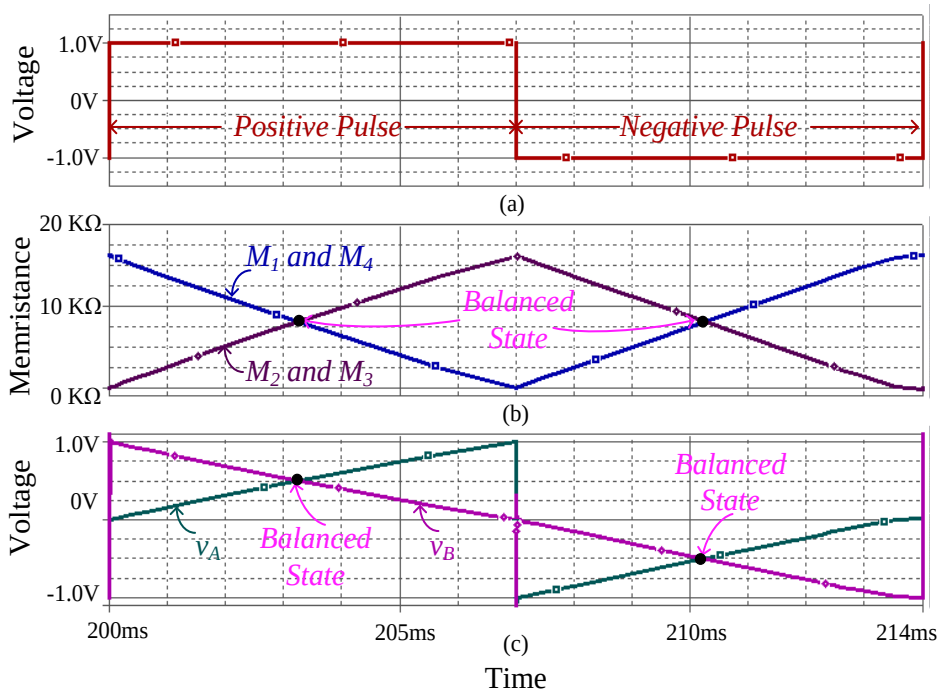
Simulations for the weight programming of the memristor emulator-based synaptic circuit as in Figure 6 have been conducted. The synaptic weights were programmed with ±1 V input pulses. Figure 9(b) and Figure 9(c) show the memristance variation and the voltage across each memristor in the memristor bridge circuit for a positive and negative wide pulse.

We assume that the initial memristance of the memristors M_1 = M_4 and M_2 = M_3 are 16 KΩ(maximum) and 100 Ω(minimum) respectively. Since the polarity of M_1 and M_4 are opposite to that of M_2 and M_3, the memristances M_1 and M_4 decrease, while those of M_2 and M_3 increase for positive pulse input, as shown in Figure 9(b). Thus, the voltage v_A increases while v_B decreases as shown in Figure 9(c). When M_1 = M_2 = M_3 = M_4, v_A equals to v_B and the output voltage becomes zero.

At this state, the synaptic weight is zero. When M_1 or M_4 is less than M_2 or M_3, the voltage v_A is greater than v_B. If the pulse width is sufficiently wide, the voltages at v_A and v_B reach to +1 V and 0 V, respectively. Note that each memristor pair (M_1, M_4) or (M_2, M_3) is with opposite polarity. Therefore, the composite memristance of each memristor pair is constant.

Similarly, when M_1, M_4 and M_2, M_3 are in minimum and maximum state respectively, then a negative wide voltage pulse is applied to the memristor bridge synapse, so that the memristance of memristor M_1, M_4 and M_2, M_3 are moved to the opposite direction compare to the positive case input pulse. In this case, voltage v_A moves toward 0V and that of v_B moves toward −1 V as shown in Figure 9(c).

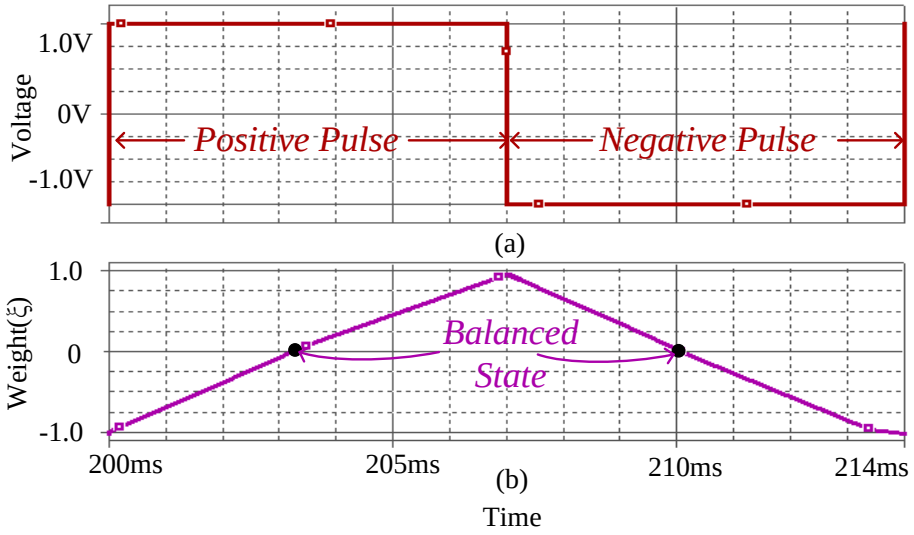
Figure 9. Variation of memristance and voltages (v_A, v_B) when positive and negative pulses are applied to the emulator-based memristor bridge synapse (**a**) positive and negative input voltage pulses; (**b**) memristance variations; (**c**) voltage variations at v_A and v_B.



The linearity of the weight programming of the memristor emulator-based memristor bridge synapse has been tested by applying wide positive and negative pulses. The weight values were computed by measuring the output voltages of the memristor bridge circuit while known input voltages were applied, as described in Section 4.1 and 4.2. The results of circuit simulations for the synaptic weighting are shown in Figure 10.

As seen in this simulation result, synaptic weight (ξ) can be changed toward positive (from −1 to +1) and negative direction (+1 to −1) by a positive pulse and negative pulse, respectively. Observe that the programmed weight (ξ) is almost linearly proportional to the width of the input pulse. The linearity of synaptic weight programming in the memristor bridge comes from the complementary action of the back-to-back memristors at each branch of the memristor bridge circuit.

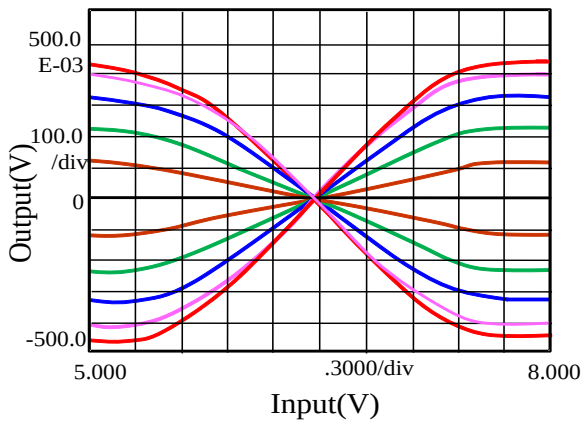
Figure 10. Weight variations of the memristor bridge circuit while positive and negative pulses are applied (**a**) positive and negative input pulses; (**b**) weight variations during each pulse period.



5.2. Synaptic Multiplication

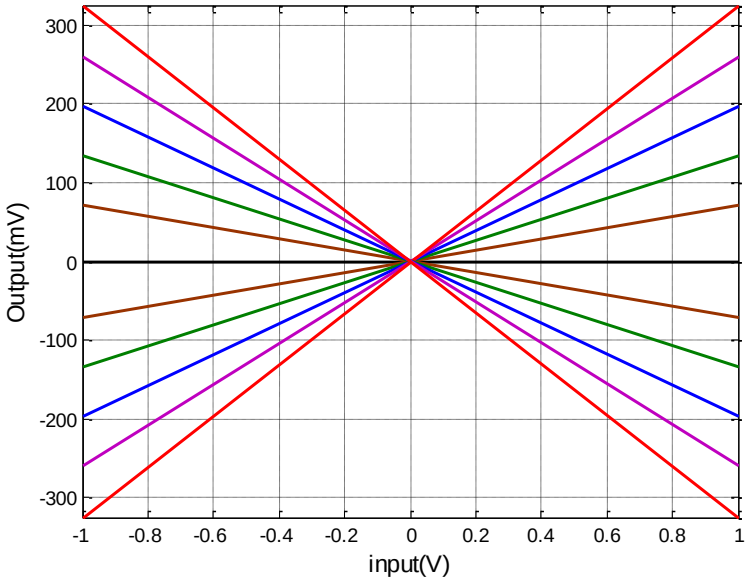
Simulations of the synaptic weight processing were also conducted with our memrisor emulator-based bridge synapse. Figure 11(b) shows the linearity of the relationship between the input voltages, and the output of the memristor emulator-based bridge synapse. The weighting factor ξ is in the range [−0.1,0.1] when synaptic input range is [−1,1] V. The performance of the conventional analog multiplication (synaptic weight) circuit employed in the programmable analog vector matrix multiplication and CNN [10,16] is shown in Figure 11(a). As in the Figure 11(a), the linear region on the function of input-output relation is quite narrow and the intervals between graphs are not quite uniform. However, in the case of memristor bridge synapse, the linear regions are very wide and the intervals between graphs are uniform as in Figure 11(b). The linearity of the memristor bridge synaptic circuit comes from the linear weight assignment at the memristor bridge synapse and the operation at the middle of the memristor dynamic range.

Figure 11. Synaptic multiplication with (**a**) Gilbert multiplier-based circuit [10,16]; (**b**) memristor based circuit.



(**a**)

Figure 11. *Cont.*



(**b**)

5.3. Memristor Synapse-Based Neuron

A single layer neuron with two input terminals as in Figure 8(a) has been built with the proposed memristor emulator-based synapse circuit. Two different kinds of sinusoidal voltage signals were sampled by doublet pulses and applied to the memristor synaptic circuits. Figure 12(a–e) are input voltage signals, weighted voltage signals of Figure 12(a,b) with weighting values of $\xi = -0.25$ and 0.1, and weighted sum appeared across R_L where R_L was 10 K.

Figure 12. Operations of the memristor emulator-based neuron. Input signals sampled with doublet pulses from two different sinusoidal signals were applied to the memristor bridge synapses, (**a**) input voltage signal for $\xi = -0.25$; (**b**) input voltage signal for $\xi = 0.1$; (**c**) weighted voltage signals with $\xi = -0.25$; (**d**) weighted voltage signals with $\xi = 0.1$ and (**e**) weighted sum appeared at the output of the neuron.

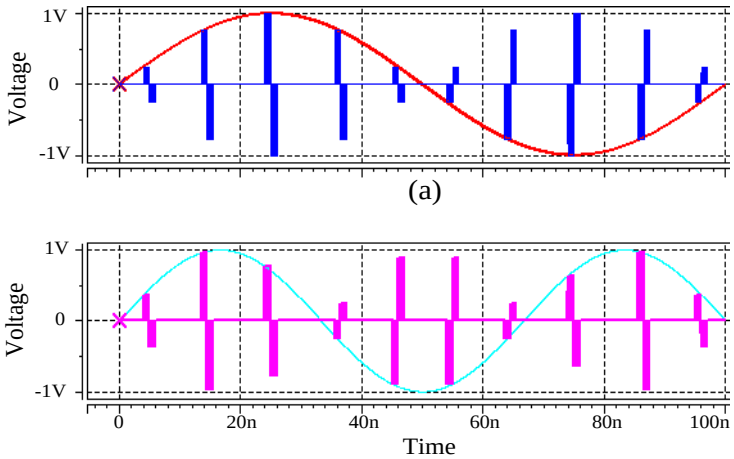
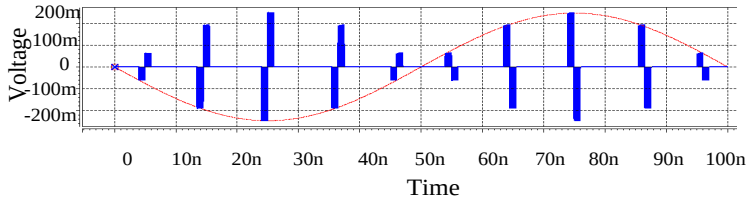
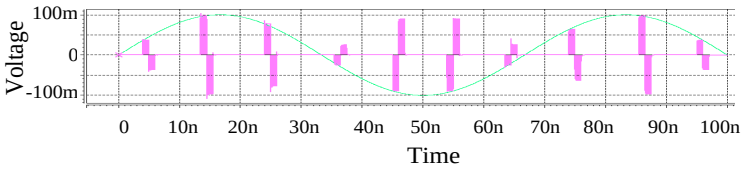


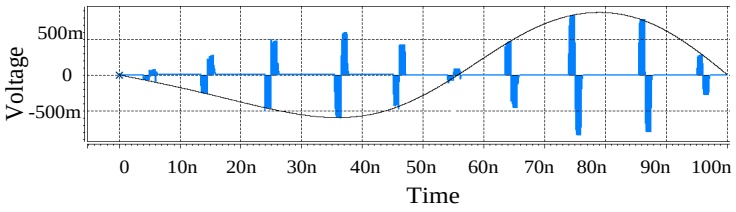
Figure 12. *Cont.*



(c)



(d)



(e)

The use of doublet signals [28] is aimed at preventing the memristances from unwanted drifting. For the subsequent processing with non-memristor circuits, each doublet pulse signal needs to be converted to a singlet pulse. This can be achieved by sampling the output signal at every first pulse period of each doublet. The simulation result shows that the proposed memristor synapse circuit performs synaptic action excellently without significant distortion.

6. Conclusions

This paper is the extension of our previous work on memristor bridge synapses [24]. In this paper the mathematical model-based memristor bridge synapse of the previous work is built with memristor emulator-based synapse circuits.

Simulations for the weight programming were performed with memristor emulator-based bridge synapse circuit. The programmed weights were almost linearly proportional to the width of the input pulses. The linearity of weight programming in the memristor bridge synapse comes from the complementary action of the back-to-back memristor pair of the memristor bridge synapse. The simulations of synaptic multiplication between programmed weight and input signal also was conducted. It showed an excellent linearity compared to that of the conventional Gilbert multiplier-based circuit. In the simulation of a single layer neuron, the proposed memristor-based neural circuit performs both synaptic weighting and summing actions very well without significant distortion.

There are several benefits with the proposed memristor synapse circuit over the conventional circuits. The number of transistors required for the memristor based synaptic circuit is three, while that of Gilbert multiplier-based synaptic circuit is seven. Considering the fact that the total size of four memristors with the proposed circuit is less than that of a single transistor, the size benefit of the

proposed synaptic circuit is obvious. Also, non-volatility as memory and excellent linearity in synaptic operation are additional benefits of the proposed memristor synaptic circuit.

Acknowledgments

This work was supported in part by the National Research Foundation of Korea (NRF) grant (No. 2010-0006871) and the US Air Force grant number FA9550-10-1-0290.

References

1. Haykin, S.S. *Neural Networks: A Comprehensive Foundation*; Prentice Hall: Upper Saddle River, NJ, USA, 1999.
2. Lawrence, J. Introduction to neural networks. *CA Sci. Softw*. **1995**, *346*, 1075–1079.
3. Rumelhart, D.E.; McClelland, J.L. *Parallel Distributed Processing: Exploration in the Microstructure of Cognition*; MIT Press: Cambridge, MA, USA, 1986.
4. Murre, J.M.J. Neurosimulators. In *Handbook of Brain Research and Neural Network*; Arbib, M.A., Ed.; MIT Press: Cambridge, MA, USA, 1995.
5. McCartor, H. A highly parallel digital architecture for neural network emulation. In *VLSI for Artificial Intelligence and Neural Network*; Delagado-Frias, J.G., Moore, W.R., Eds.; Plenum Publishing Company: New York, NY, USA, 1991.
6. Ramacher, U. *et al*. Multiprocessor and memory architecture of the neurocomputers SYNAPSE-1. *Int. J. Neural Syst.* **1993**, *4*, 333–336.
7. Holler, M. Tam, S. Castro, H. Benson, R. An electrically trainable artificial neural network (ETANN) with 10240 "Floating gate" synapse. In *Proceedings of International Joint Conference on Neural Network,* Washington, DC, USA, 18–22 June 1989; Volume 2, pp. 191–196.
8. Withagen, H. Implementing backpropagation with analog hardware. In *Proceedings of IEEE World Congress on Computational Intelligence*, Orlando, FL, USA, 27 June–2 July 1994; Volume 4, pp. 2015–2017.
9. Lindsey, S.; Lindblad, T. Survey of neural network hardware invited paper. *Proc. Appl. Sci. Artif. Neural Networks Con.* **1995**, *2492*, 1194–1205.
10. Kub, F.J.; Moon, K.K.; Mack, I.A.; Long, F.M. Programmable analog vector-matrix multipliers. *IEEE J. Solid-State Circuits* **1990**, *25*, 207–214.
11. Schlottmann, C.R.; Hasler, P.E. A highly dense, low power programmable analog vector-matrix multiplier: The FPAA implementation. *IEEE J. Emer. Sel. Top. Circ. Syst.* **2011**, *1*, 403–411.
12. Chua, L.O.; Yang, L. Cellular neural networks: Applications. *IEEE Trans. Circuits Syst.* **1988**, *35*, 1273–1290.
13. Chua L.O.; Yang, L. Cellular neural networks: Theory. *IEEE Trans. Circuits Syst.* **1988**, *35*, 1257–1272.
14. Kim, H.; Roska, T.; Son, H.; Petras, I. Analog addition/subtraction on the CNN-UM chip with short-time superimposition of input signals. *IEEE Trans. Circuits Syst. I* **2003**, *50*, 429–432.
15. Kim, H.; Son, H.; Roska, T.; Chua, L.O. High-performance viterbi decoder with circularly connected 2-D CNN unilateral cell array. *IEEE Trans. Circuits Syst. I* **2005**, *52*, 2208–2218.

16. Domíınguez-Castro, R.; Espejo, S.; Rodr íguez-V ázquez, A.; Carmona R.A.; F öldesy P.; Zar ándy, Á.; Szolgay P.; Szir ányi, T.; Roska, T. A 0.8-µm CMOS two-dimensional programmable mixed-signal focal-plane array processor with on-chip binary imaging and instructions storage. *IEEE J. Solid State Circuits* **1997**, *32*, 1013–1026.
17. Strukov, D.B.; Snider, G.S.; Stewart, D.R.; Williams, R.S. The missing memristor found. *Nature* **2008**, *453*, 80–83.
18. Chua, L.O. Memristor-the missing circuit element. *IEEE Trans. Circuit Theory* **1971**, *CT-18*, 507–519.
19. Chua, L.O.; Kang, S.M. Memristive devices and systems. *Proc. IEEE* **1976**, *64*, 209–223.
20. Ventra, M.D.; Pershin, Y.V; Chua, L.O. Circuit elements with memory: Memristor, memcapacitors and meminductors. *Proc. IEEE* **2009**, 97, 1717–1724.
21. Cantley, K.D.; Subramaniam, A.; Stiegler, H.J.; Chapman, R.A.; Vogel, E.M. Hebbian learning in spiking neural networks with nanocrystalline silicon TFTs and memristive synapse. *IEEE Trans. Nanotechnol.* **2011**, *10*, 1066–1073.
22. Snider, G. Self-organized computation with unreliable, memristive nanodevices. *Nanotechnology* **2007**, *18*, 1–13.
23. Kim, H.; Sah, M.P; Yang, C.; Roska T.; Chua L.O. Neural synaptic weighting with a pulse-based memristor circuit. *IEEE Trans. Circuit Syst. I* **2011**, *59*, 148–158.
24. Kim, H.; Sah, M.P; Yang, C; Roska, T; Chua, L.O. Memristor bridge synapses. *Proc. IEEE* **2012**, doi:10.1109/jproc.2011.2166749.
25. Kim, H.; Sah, M.P; Yang, C; Cho, S. Chua, L.O. Memristor emulator for memristor circuit applications. *IEEE Trans. Circuit Syst. I* **2012**, in press.
26. Pershin, Y.V.; Ventra, M.D. Practical approach to programmable analog circuits with memristors. *IEEE Trans. Circuits Syst. I* **2010**, *57*, 1857–1864.
27. Pershin, Y.V; Ventra, M.D. *Experimental Demonstration of Associative Memory with Memristive Neural Networks*; Cornell University Library: Ithaca, NY, USA, 2009; ArXiv:0905.2935. Available online: http://arXiv.org/abs/arXiv:0905.2935 (accessed on 18 May 2009).
28. Yang, C.; Sah M.P.; Adhikari S.; Park, D.; Kim, H. Highly accurate doublet generator for memristor-based analog memories. *IJBC* **2012**, in press.