Chapter 5

Memristor Threshold Logic FFT Circuits

Alex Pappachen James

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/66583

Abstract

One of the possible approaches to achieve more than Moore's law with signal processing circuits is to inspire from functioning of human brain to mimic neural functions by exploring emerging technologies such as memristor circuits. While fast Fourier transform (FFT) implementations are largely based on CMOS gates, they are limited by the computation speed and availability limits on the number of Boolean variables it can handle at a given time. Biological neurons and networks on the other hand are generalized in nature and can handle both analogue and digital signals. Through this chapter, memristor-based resistive threshold logic family of gates that inspire from brain-like large variable logic functions is introduced. This logic consists of a memristors acting as weights to the inputs followed by threshold operations emulating neuronal synapse. Using this Boolean logic, a processing unit that can compute Fourier transform of a given set of inputs was developed. Various comparisons of the circuit are found to be advantageous in implementing neuromorphic circuits. The existing logic families were carried out and the proposed logic family was found too advantageous in many ways.

Keywords: memristors, threshold logic, circuits, Fourier, FFT

1. Processing in brain

Human brain is the processing centre of human nervous system. Average adult human brain weighs about 1.5 kg and [1, 2] is composed of about 10 billion nerve cells or neurons. On average, each neuron is connected to other neurons through about 10,000 synapses. The brain's network of neurons forms a massively parallel information processing system.

On the other hand, approximates arithmetic, such as the ability to calculate whether '26 + 32' is closer to 60 or 75, is not processed in the same manner or in the same part of your brain. Memorization of the results of operations such as multiplication tables reduces the need to use complex cognitive processing, such as going through the learning processes every time

the task is given. The idea of storing for long-term retrieval is supported by the long-term memories in the human brain, where the retrieval of results from the memory reduces the effort on cognitive processes.

The ability of the brain to process computational operations in real time is reflective of an active working memory. In many of the mental calculations, one makes on a day-to-day basis can be analysed by looking into the activities in the prefrontal cortex (see **Figure 1**). The studies using neuroimages indicate 10 separate regions in the brain that contribute to even simple task of subtraction of two numbers. The main areas of activation for this simple task include fusiform gyrus, parietal cortices, lateral and medial parts of the temporal lobe and inferior parts of the frontal lobes.

The interconnections between the modules and the way they interact with each other for different set of arithmetic operations are different. It is also found that there is a separate network for estimation (bilateral inferior parietal cortex) as opposed to computation (left parietal and frontal cortices). These features point out the fact that there is one specific unit for performing computation; instead it is a collaborative effort between various regions in the brain.

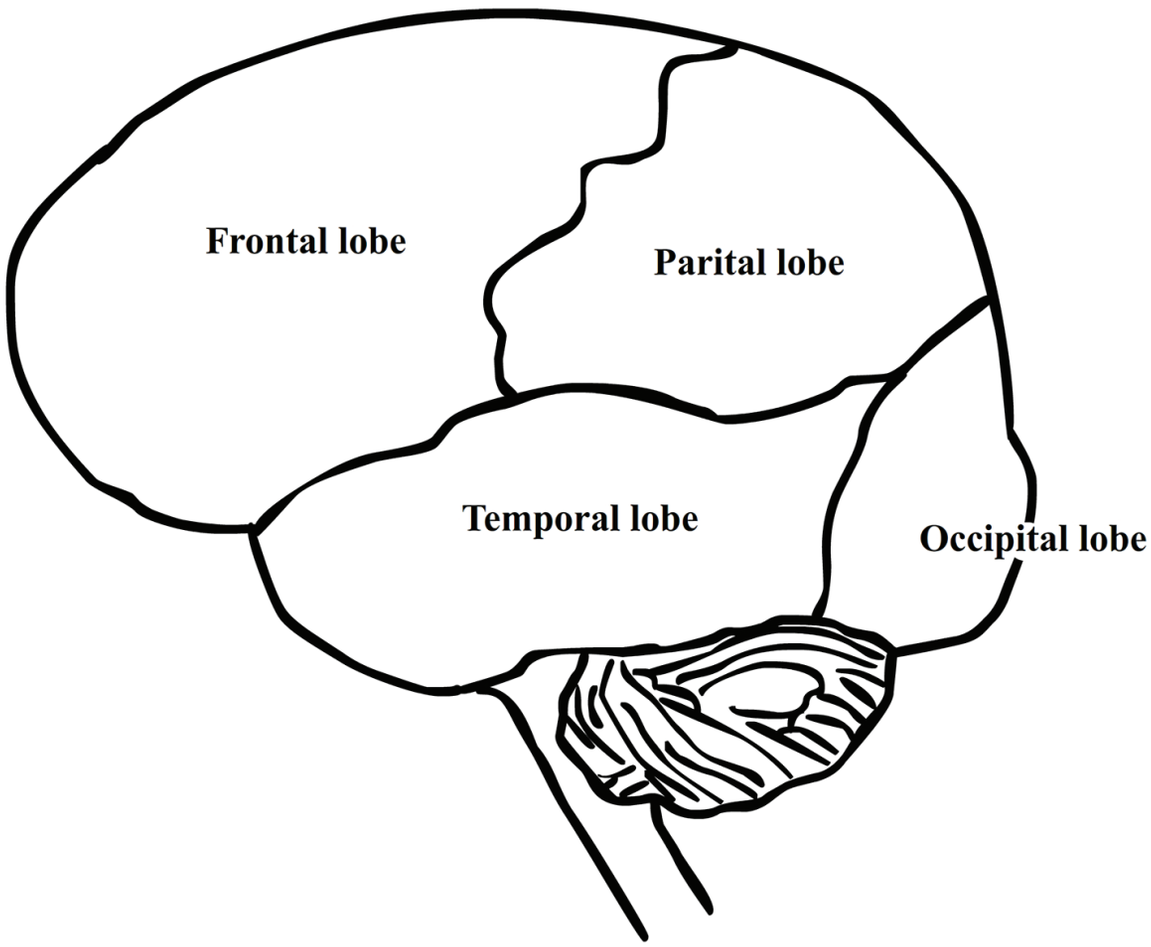

Figure 1. Functional units in brain.

These features of brain are similar to the normal CPU architecture—a separate unit for arithmetic and logic functions, with a difference in the mode of operation. Instead of doing algorithmic approach, the brain works based on what it learned or we can say that the patterns it has learned. If we can implement this key feature of brain in hardware, then we can have a brain-like fast processing unit.

In the brain the basic functional unit is a cell, called neuron. For each function, there will be separate set of neuron cells, which are learned to do that particular task. If we can develop a neuromorphic circuit for the neuron cell and can make it learn to do particular tasks, we can use it to develop a brain-like processing unit, that is what is achieved through this project work.

2. Neurons

Neurons are the basic building blocks of the nervous system, which includes brain, spinal cord and peripheral ganglia. Neurons are electrically excitable cells and they process and transmit information through electrochemical signals. Neurons connect together to form what is known as neural networks.

The basic structure of a biological neuron is shown in **Figure 2**. It consists of a cell body, dendrites and axons. Cell body or the soma is bulbous in shape and contains the nucleus. The cell body or the soma contains many cell organelles, including Nissl granules that are the site of protein synthesis. Nissl granules contain endoplasmic reticulum and free polyribosomes.

Dendrites arise from the cell body, branches into what is known as the 'dendritic tree'. Dendrites are the branched projections of the neuron arising from the cell body and its function is to receive the electrochemical simulations from other neurons and to conduct it to

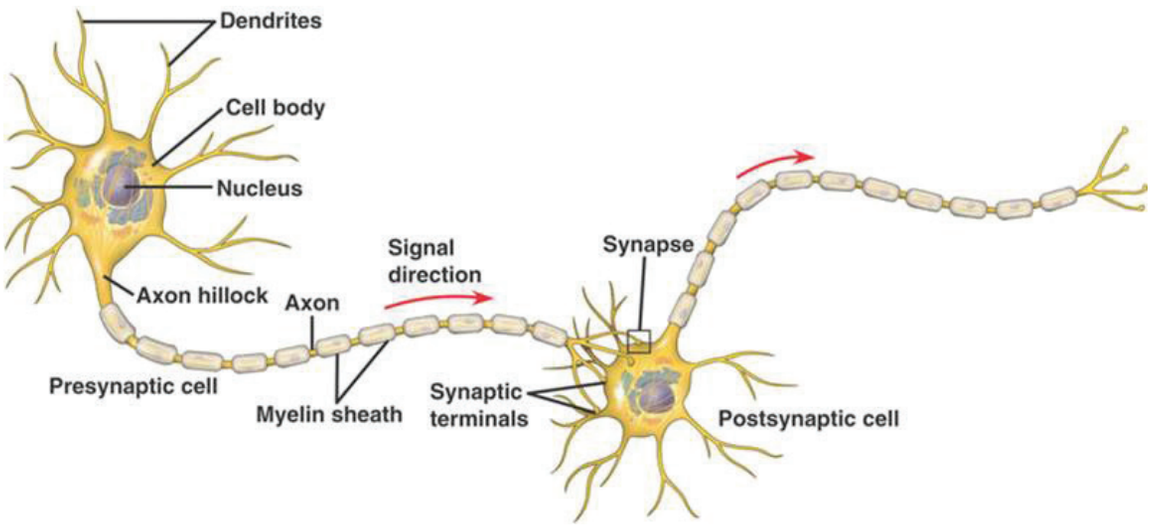

Figure 2. Biological neurons.

the cell body. The electrical simulations are transmitted from one neuron to the dendrite of another neuron at the synaptic terminals.

Another important part of the neuron is the axon. Axon arises from the cell body at a site called axon hillock and extends to over 1 m in length. A neuron can have multiple dendrites, but only one axon. Axon is covered by a layer of dielectric material myelin, known as myelin sheath. Before termination, the axon gets divided into a large number of branches.

The axon terminals of one neuron connect to the dendrites of another neuron through synapses. Electrochemical signals are transmitted from one neuron to another through synapses. Chemicals known as neurotransmitters are released from the presynaptic neuron, which binds to the receptors located at the dendrites of the postsynaptic neurons. These neurotransmitters are initially present in small bag-like structures known as synaptic vesicles that are found at the axonic terminals of the neurons. These synaptic vesicles, when excited, migrate towards the synapse and get attached to the synapse and release the chemical ions through the semipermeable membrane of the synapses.

The major ions that are involved in the process are sodium, potassium, chlorine and calcium. Once released, these ions diffuse through the semipermeable membrane and binds to the receptors which are present on the dendrites of the post-synaptic neurons. The basic structure of a synapse is shown in **Figure 3**.

Due to the ion exchange between neurons, a gradient in the ion concentration arises on either side of the semipermeable membrane. Due to this ion concentration difference, a potential will be generated, known as Nernst potential. Changes in the cross-membrane voltage between the intra-cellular and extra-cellular potential will alter the function of the voltage-dependention

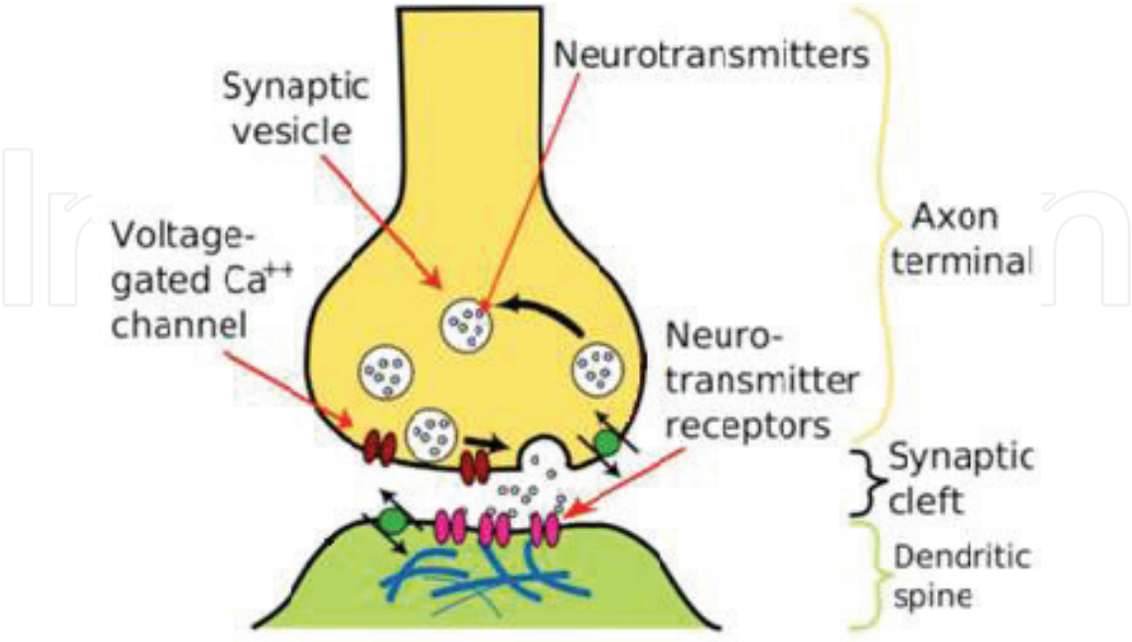

Figure 3. Structure of synapse.

channels. As the difference in ion concentration increases, the resultant Nernst potential also increases and when this potential reaches a particular threshold value, the post-synaptic neuron fires and an action potential is generated which moves from cell body to the next neuron through the axon. This is how a biological neuron transmits signals.

There are several differences between the processing in human brain and processing in a computer. One of the most important differences is that brain is analogue whereas the computers are digital. The computers work with 0's and 1's whereas neuron signals are not bi-state. But we can find a superficial similarity between neurons and digital circuits in the aspect that neurons fire an action potential when they reach a threshold value. In computers, information in memory is accessed by polling its precise memory address. This is known as byte-addressable memory whereas brain uses content-addressable memory.

Human brain can be considered as a massively parallel machine, where different functions are carried out simultaneously in different parts of the brain. Brain has got several dedicated modules for carrying out different functions. But if we consider the case of computers, the processing is in modular and serial in nature.

The brain has got a body at its disposal. This may seem to be trivial, but this is a major difference which gives the humans a clear advantage over the computers. Once the brain takes a decision based on the input signals, the brain directs the body to respond to the signals. But in computers, although it can take decisions based on the input signals, there is no body so that it can respond to the stimulus.

Although there are several other differences, one of the most important differences between brain and computing processors is that there exists no distinction between memory and processing architecture in brain. These two important activities are not separable. As the neurons process information, they also modify their synapses that are the substrate of memory. But in computers there exist a clear distinction between processor and memory.

3. Memristor

Memristor is considered to be the fourth fundamental electronic component. The basic state variables in any circuit are voltage (V), current (I), charge (q) and flux (ϕ). The state variables and relations between them are shown in **Figure 4**. Prior to the 1970s, only resistor, capacitor and inductor were known. No component showing the property of memristance was known to the scientific community. It was in 1971 that Leon Chua gave the scientific and logical basis for the existence of a two terminal circuit element called memristor (memristor is the shortened form of ‘memory resistor') [3]. He reasoned the existence of the fourth element through symmetry arguments. Although he showed that the memristor has many interesting and valuable circuit properties, he was unable to implement the memristor in the form of a physical device without an internal power supply.

The six different mathematical relations connect the four fundamental circuit variables: voltage (V), current (I), charge (q) and magnetic flux (ϕ). These relations are indicated in **Figure 4**.

Since there were no devices that reflected a relation for long between the charge and flux, the memristor was referred to as the missing element, with memristance (M), with $d\phi = Mdq$.

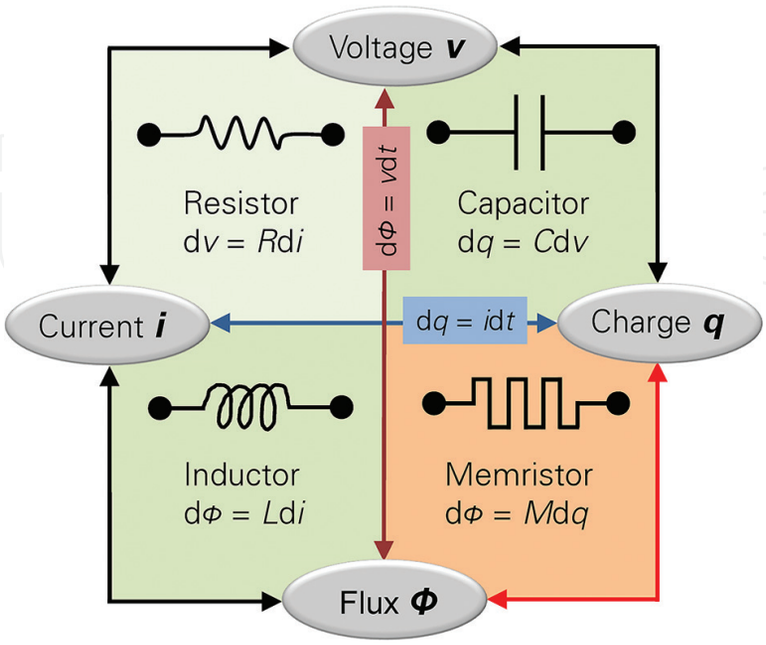

Figure 4. Relation between various state variables in an electronic circuit.

One of the most advertised and commercially inclined versions of the memristor was develop by HP Labs that was based on a thin film of titanium dioxide [4, 5]. The main reason that gained attention for this device was the possibility to scale the device beyond the traditional CMOS limits. While, there is debate on the charge transport mechanisms and resistance switching behaviours, the hypothesis is that the hysteresis requires some sort of atomic rearrangement that modulates the electronic current. The HP memristor device consists of a thin film of titanium dioxide (TiO_2) sandwiched between two platinum electrodes, with one side of the titanium dioxide doped with oxygen vacancies, TiO_{2-x} (see **Figure 5**).

The undoped region is insulated and has higher resistance than the doped region. The effective resistance within the memristor is determined by the boundary between the doped and

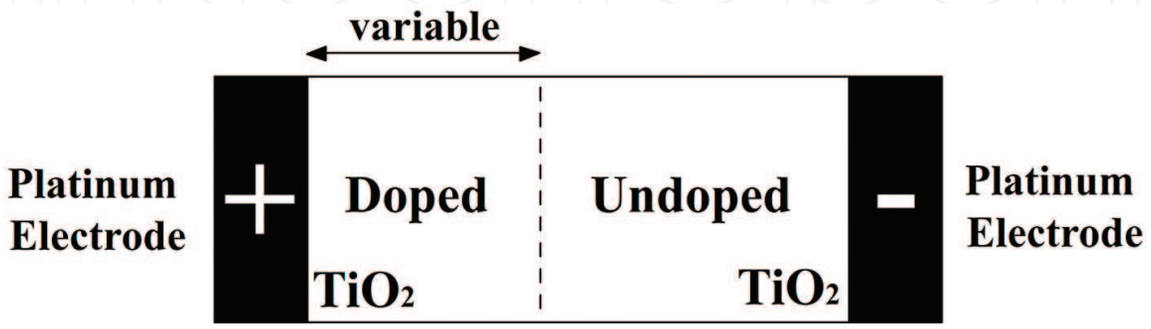

Figure 5. Memristor modelled by HP.

http://dx.doi.org/10.5772/66583

undoped regions. Let D be the total width of the TiO_2 layer and W be the width of the doped region. Then the effective resistance of the device is given by $M_{eff} = (W/D)R_{ON} + (1-(W/D))R_{OFF}$, where R_{ON} is the resistance of the device if it is completely doped and R_{OFF} is the resistance of the device if it is completely undoped, see **Figure 6**.

Under the situation, when a positive voltage at the side of the doped region and negative voltage at the side of the undoped region, the oxygen vacancies move from the doped side to the undoped side, thus, increasing the width (W) of the doped region. This results in the overall resistance of the memristor.

If the polarity of applied voltage is reversed, that is, positive potential is applied to the undoped side and negative potential is applied to the doped side, then the width of the undoped region increases, thereby increasing the effective resistance of the device.

When input voltage is withdrawn or when there is no potential difference between the terminals, the memristor maintains the boundary between the doped and undoped region, since the oxygen ions remain immobile after removal of the input voltage. Thus, the resistance will be maintained at the same value before withdrawing the input voltage.

The resistance of the memristor increases when current flows through it in one direction and the resistance value decreases when the current flows through it in the opposite direction. It can retain the resistance value it had at that point of time, if the current is stopped.

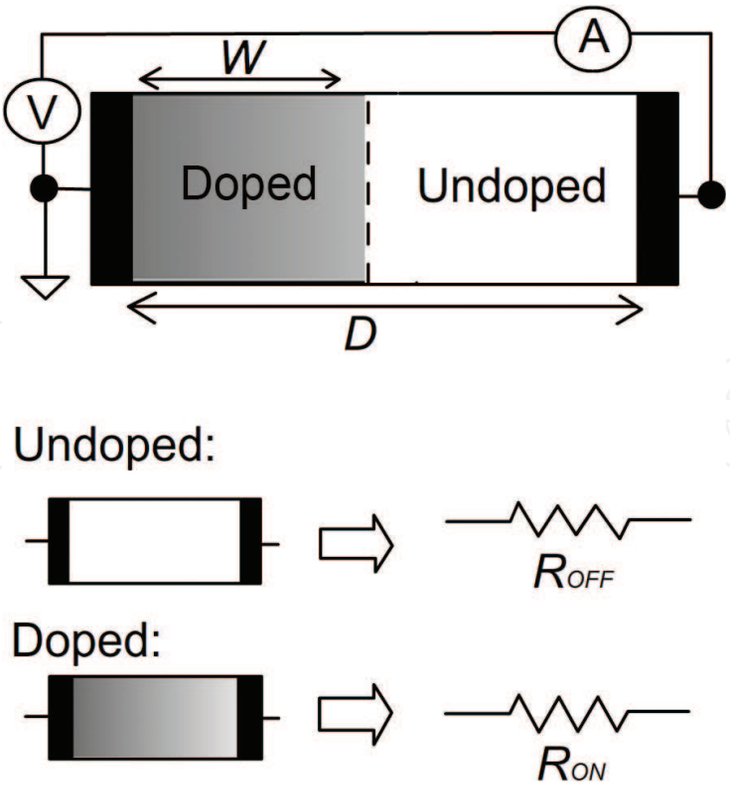

Figure 6. Working of a memristor.

We can see from $i = v/M(q)$ that when there is no voltage difference across the memristor, there is no current through the memristor. When the potential applied is reversed, the width of the undoped region increases resulting in an increase in effective resistance. The high resistance blocks any reverse leakage current and adding more inputs, the collective current does not increase significantly as the effective resistance remains constant.

The *V-I* characteristics of the memristor are shown in **Figure 7**. Generally, indicative of a pinched hysteresis effect [3, 4, 6], the changes in the slope indicate the switching behaviour, with each of the switch having at least two resistance states. With change in operating frequencies, the resistance values of the state become equal at high frequencies. The frequency dependence of memristor is shown in **Figure 7**.

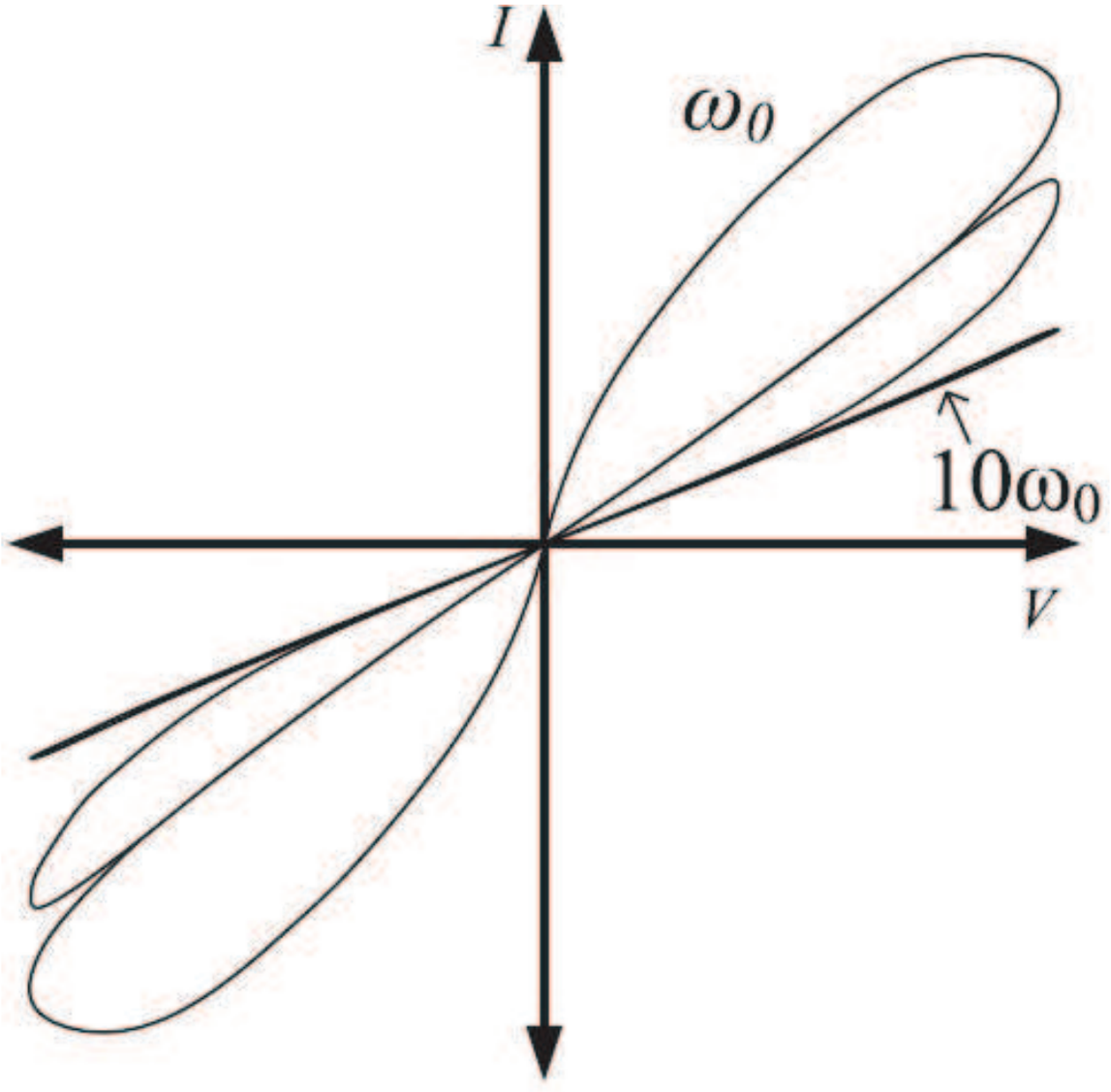

Figure 7. The V-I characteristics of the memristor device.

Over the years, there have been several efforts to manufacture memristors. The various attempts include polymeric or ionic memristors, resonant tunnelling diode memristors, manganite memristors and spintronic memristors. In addition to the memristor devices, there are circuits that emulate the memristor behaviour, generally referred to as memristive systems.

The crossbar architecture with memristor is used to build ultra-dense memory cells (RRAM—resistive random-access memory). Another application is the use of memristor for emulating neural circuits that can help develop a range of hardware-based machine learning methods. The memristors can be also used to implement multilevel memories [7, 8] and analogue memories [9]. They also find application in configurable logic arrays [10].

4. Resistive threshold logic

Resistive threshold logic [11, 12] is a new logic family based on a resistive voltage divider and threshold logic, which is the hardware implementation of the neuron cell by configuring the cognitive memory network [13]. This circuit is capable of doing all Boolean logic [14, 15].

Two-input basic resistive threshold logic cells are shown in **Figure 8**. For a straight forward approach, we started by using semiconductor resistors for the resistive divider. The input to the resistive divider is voltage levels that can be equated to the logic inputs [10] of a digital logic gate. Based on the logic functionality required, predefined threshold levels will be used in the thresholding part.

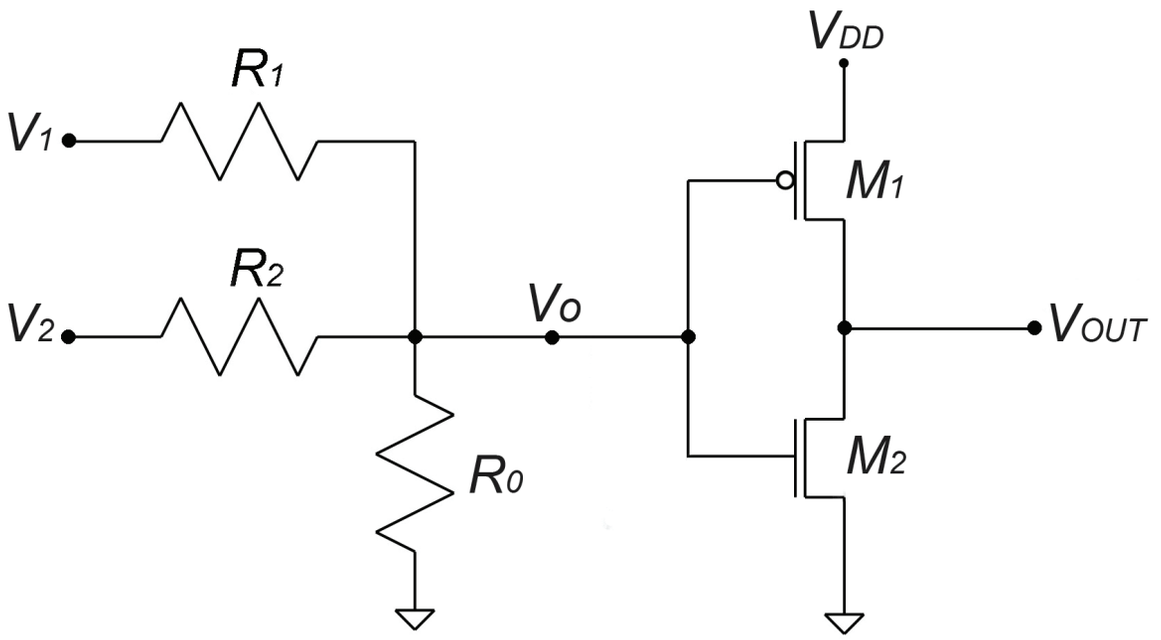

Figure 8. Two-input basic resistive threshold logic cell.

An N-input resistance divider circuit consist of N input resistors R_i and one reference resistor R_0. The output voltage V_0 for N-input voltages V_i is shown in Eq. (1),

$$V_0 = \frac{\sum_{i=1}^{N} \frac{V_i}{R_i}}{\left(\frac{1}{R_0} + \sum_{i=1}^{N} \frac{1}{R_i}\right)} \tag{1}$$

We keep equal values to R_i's and $R_0 = m R_i$, which results in:

$$V_0 = \frac{\sum_{i=1}^{N} V_i}{\frac{1}{m} + N} \tag{2}$$

The inverter with a threshold V_{th} and a two-input resistive divider is used to implements the NAND and NOR gates shown in **Table 1**. Given that, $V_{dd} = 1V$, $V_H = 1V$, $V_L = 0V$, (see **Table 1**), when the threshold voltage of the inverter is set between 0 and 1/3 V, the cell functions as NOR logic, while between 1/3 and 2/3 V the cell functions as NAND logic. This implies that varying the threshold voltage of the inverter a single cell structure can be used to implement NAND and NOR logic.

In general, the range of threshold voltage, V_{th} of NOR gate is $\frac{Nm\,V_L}{1+Nm} \le V_{th} \le \frac{(V_H + (N-1)V_L)\,m}{Nm+1}$ and NAND gate is, $\frac{m(V_L + (N-1)V_H)}{Nm+1} \le V_{th} \le \frac{mN\,V_H}{Nm+1}$.

To find the m value in Eq. (2), the lower limit of NAND gate threshold range $\frac{m(V_L + (N-1)V_H)}{Nm+1}$ is equated to $\frac{V_H + V_L}{2}$. And to solve the equation V_L is taken as 0 V. So we get the m value as $\frac{1}{N-2}$. Now we can say that the threshold voltage of NAND gate must be between $\frac{V_H + V_L}{2}$ and $\frac{mN\,V_H}{Nm+1}$.

But there are certain drawbacks in using semiconductor resistors for building the voltage divider. One of the most important factors is the large leakage current of the semiconductor resistors. When the number of inputs increases the problem of leakage current becomes prohibitively high. Another drawback of using semiconductor resistor is that a change in the resistance value of the resistor due to second-order implementation effects, such as improper junctions and defects. This change in the resistance value is generally termed as the tolerance

Input voltage (V_i)		**Output voltage**	**NAND**	**NOR**
V_1	V_2	V_0	$\frac{V_L + V_H}{3} < V_{th} < \frac{2V_H}{3}$	$\frac{2V_L}{3} < V_{th} < \frac{V_L + V_H}{3}$
V_L	V_L	$\frac{2V_L}{3}$	V_H	V_H
V_L	V_H	$\frac{V_L + V_H}{3}$	V_H	V_L
V_H	V_L	$\frac{V_L + V_H}{3}$	V_H	V_L
V_H	V_H	$\frac{2V_H}{3}$	V_L	V_L

Table 1. Truth table of two-input resistive divider for NAND and NOR gates.

value of a resistors, which is usually $\mp 10\%$. This change in the resistance value may not create any problem when we are considering two-input or three-input circuits. But in practical implementations having a large number of inputs, these changes in the resistance value will have an adverse effect on the output of the circuit.

To overcome these drawbacks, semiconductor resistors were replaced with memristors developed by HP, which had negligible leakage current. Thus, the logic gate was modified as shown in **Figure 9**. The advantage of using op-amp in the circuit is that it can act as a buffer and isolates the inputs from output of the circuit thus enabling realistic implementations of a large number of inputs per gate. As the number of inputs shown in **Figure 8** increases, the threshold voltage will change. This change can affect the functionality of the gate. This problem can also be avoided by using the circuit shown in **Figure 9**. Here, the op-amp will boost the signal before applying it to the inverter. Thus, it offers the advantage of scalability over the number of inputs.

The op-amp reference voltage for NOR logic, V_{REF} is fixed as $V_{L+\Delta}$ and for NAND logic, V_{REF} is fixed as $V_{H-\Delta}$, where Δ is a small voltage defined to ensure the bounds of V_{th}. The op-amp shifts the voltage to a high value or low value depending on the input voltage, V_0.

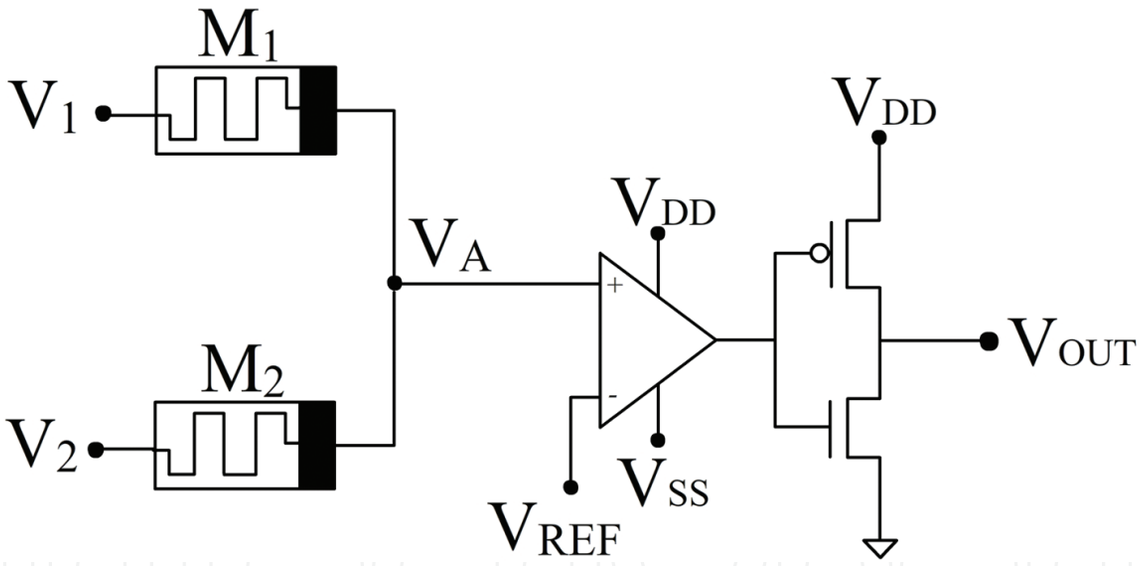

Figure 9. Resistive threshold logic.

The universal gate circuit using resistive threshold logic is shown in **Figure 10**. For the cell to work as a NAND logic, the switches S1 and S4 are closed and the output is taken from V_{out}. To implement AND logic, the switches S1 and S3 are closed and the output is taken from V_{out}. If the switches S2 and S4 are closed, we get a NOR logic from V_{out}. If both S2 and S3 are closed, OR logic can be implemented.

Figure 11 shows the circuit diagram of an *N*-input resistive threshold logic gate. Here $\{V_1, V_2, V_3, \ldots, V_N\}$ represents the inputs to the cell. $\{M_1, M_2, M_3, \ldots M_N\}$ represents the input memristors. Depending on the values of the inputs, a potential V_A is generated which is given

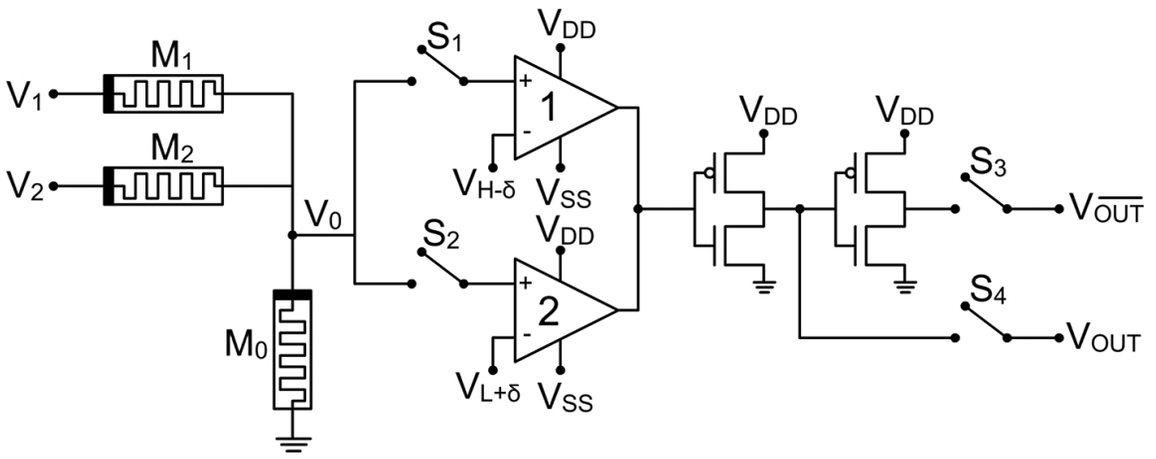

Figure 10. The universal gate structure that implements NAND, NOR, AND, OR and NOT logic functions.

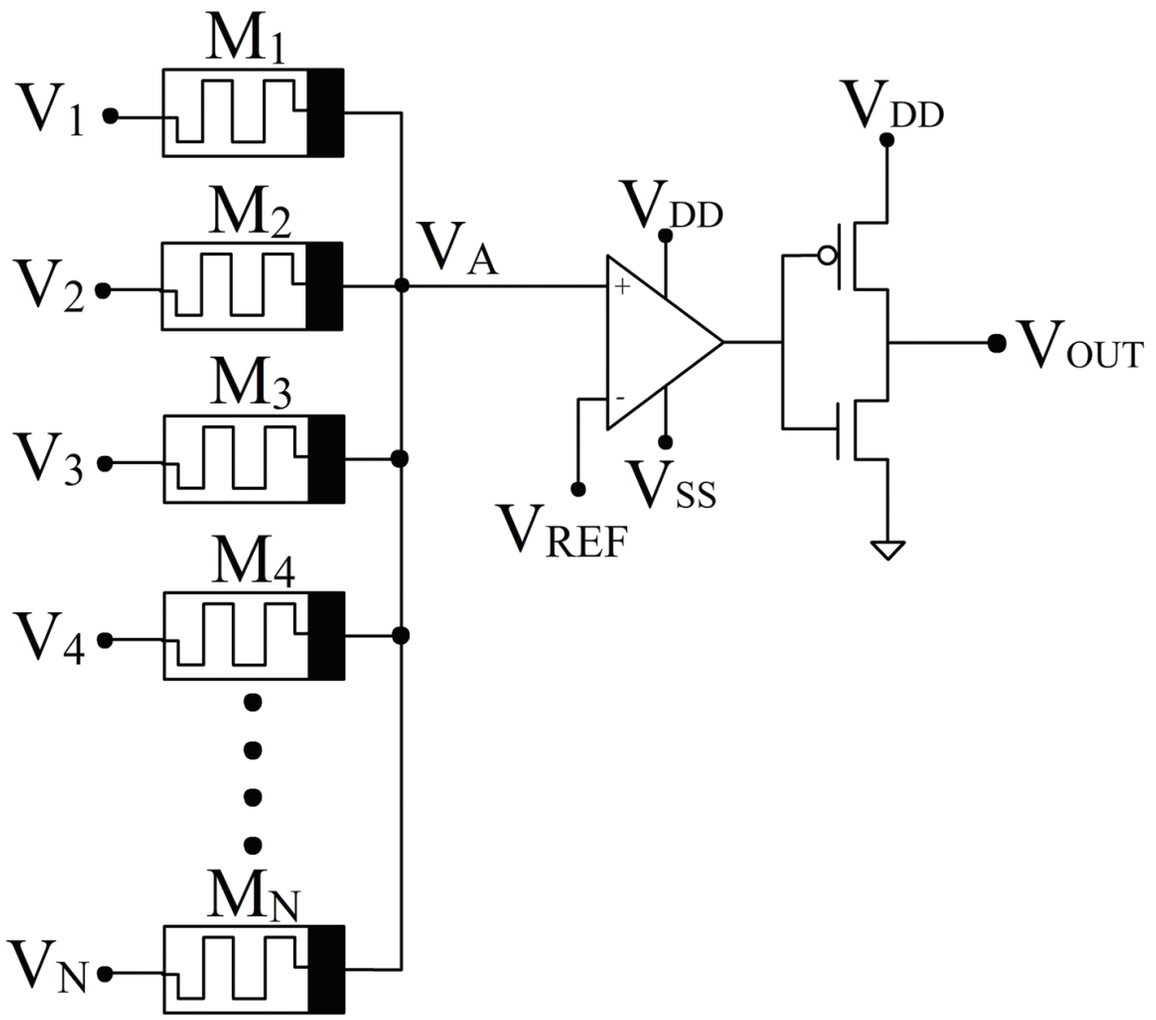

Figure 11. N-input resistive threshold logic gate.

http://dx.doi.org/10.5772/66583

to the non-inverting terminal of the op-amp. To the inverting terminal of the op-amp, a V_{REF} is given depending upon which the cell acts as either *N*-input NOR gate or *N*-input NAND gate. If the V_{REF} is fixed at $V_L + \Delta$ then the circuits acts as NOR gate and for NAND logic, V_{REF} is fixed as $V_H - \Delta$, where Δ is a small voltage defined to ensure the bounds of V_{th}. Thus, an *N*-input logic gate can be implemented using memristor-based resistive threshold logic. Here, as the number of inputs increase, only the number of memristors increases. The area consumed by a memristor is very small. Hence, even a very large input logic gate can be implemented in a very small area using the proposed resistive threshold logic.

5. Fast Fourier transform circuits

The chapter summaries our previous work on the use of threshold logic in developing a hardware implementation of Fast Fourier transform [16]. Fast Fourier transform/inverse fast Fourier transform (FFT/IFFT) is widely used algorithm to compute the discrete Fourier transform and its inverse of a given set of inputs. FFT/IFFT is mainly used in digital signal processing applications such as communication systems including orthogonal frequency division multiplexing (OFDM), spectrum analysis, DSL modems, speech coding, HDTV etc. Due to the large number of applications, it is important to design a FFT circuit which can handle large number of inputs and at the same time it should be small in area and should not become too complex.

The basic equation of four-point DFT is

$$X(k) = \sum_{i=0}^{3} x(n)\, e^{\frac{-j2\pi k}{4}},\ k = 0, 1, 2, 3 \tag{3}$$

Eq. (3) can be rewritten as an N-by-N multiplication as

$$X = Wx \tag{4}$$

Eq. (4) can be expanded in matrix form as given below.

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & 1 \\ 1 & j & -1 & j \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} \tag{5}$$

In the above matrix multiplication, $X(k),\ k = 0, 1, 2, 3$ are the DFT outputs whereas $x(k), k = 0, 1, 2, 3$ are the inputs. Using this matrix equation, we can represent the signal flow graph of four-point DFT, as shown in **Figure 12**. The exponential term $e^{\frac{-j2\pi nk}{4}} = \pm 1 \text{ or } \pm j$. Since multiplications with ±1 and ±*j* are trivial, no multipliers are needed to implement them, i.e. they can be simply realized with bypass, inversion, and/or swap for 2's complement numbers. Hence, it does not

require any multiplier to construct a butterfly element for a four-point DFT (radix-4 butterfly). From **Figure 12**, it can be seen that we can implement the processor using adders and inverters.

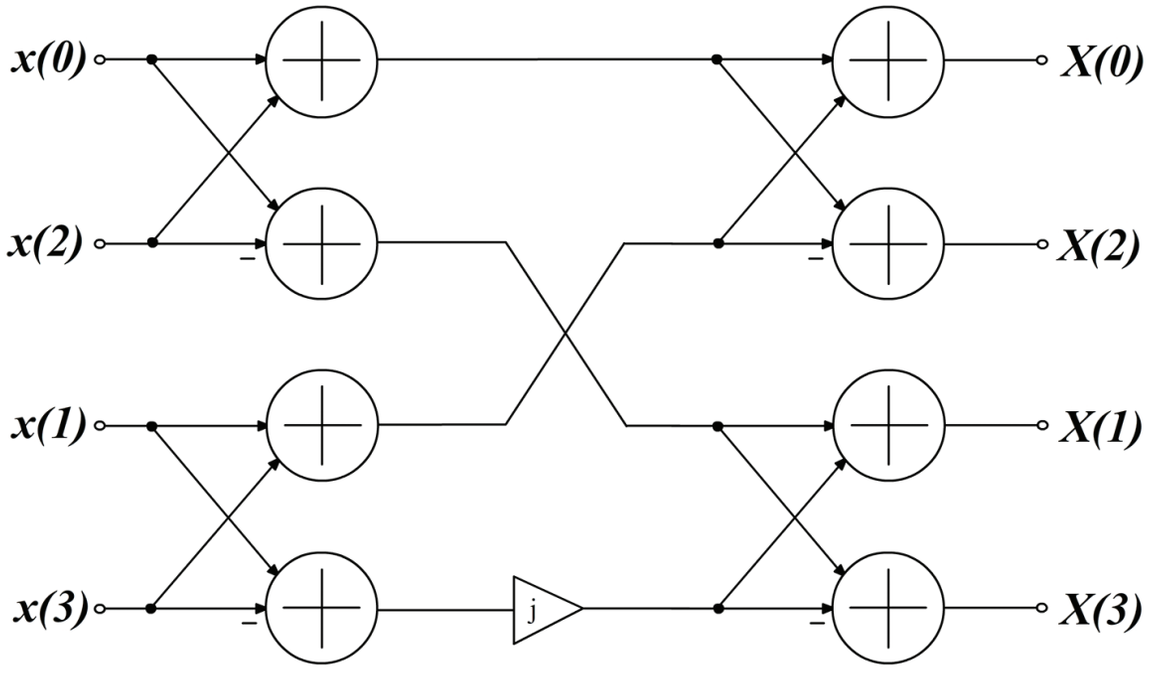

Figure 12. Signal flow graph of a four-point FFT.

Implementation of the FFT processor can be done, as shown in **Figure 13** where $x(0)$, $x(1)$, $x(2)$ and $x(3)$ are the inputs and $X(0)$, $X(1)$, $X(2)$ and $X(3)$ are the outputs of the FFT circuits. The circuits are capable of handling complex numbers and the complex numbers are separated into real and imaginary part and are given into the circuit. All the inputs to the circuit are 8 bits long.

All the FFT units take in four inputs and give the corresponding FFT output. Inputs are given to the FFT units according to **Figure 12**. From the signal flow graph of the four-point FFT, we can see that FFT outputs are obtained by equations of the form

$$X(k) = a + b + c + d \tag{6}$$

or

$$X(k) = a + b - c - d \tag{7}$$

So it is understood that certain inputs need to be subtracted in order to get the Fourier transform output. In digital circuits, subtraction is carried out by taking the 2's complement of the subtrahend and then adding it to the minuend to get the difference. For obtaining 2's complement of a number, first of all, 1's compliment of the number has to be found out and then add one to it.

From **Figure 12**, it can be seen that the real and the imaginary part of the first output, i.e. $X(0)$, in four-point FFT is obtained by computations of the form Eq. (6). So it requires addition operations only. Hence, the inputs to the first two FFT units are not complimented, as shown in **Figure 13**.

http://dx.doi.org/10.5772/66583

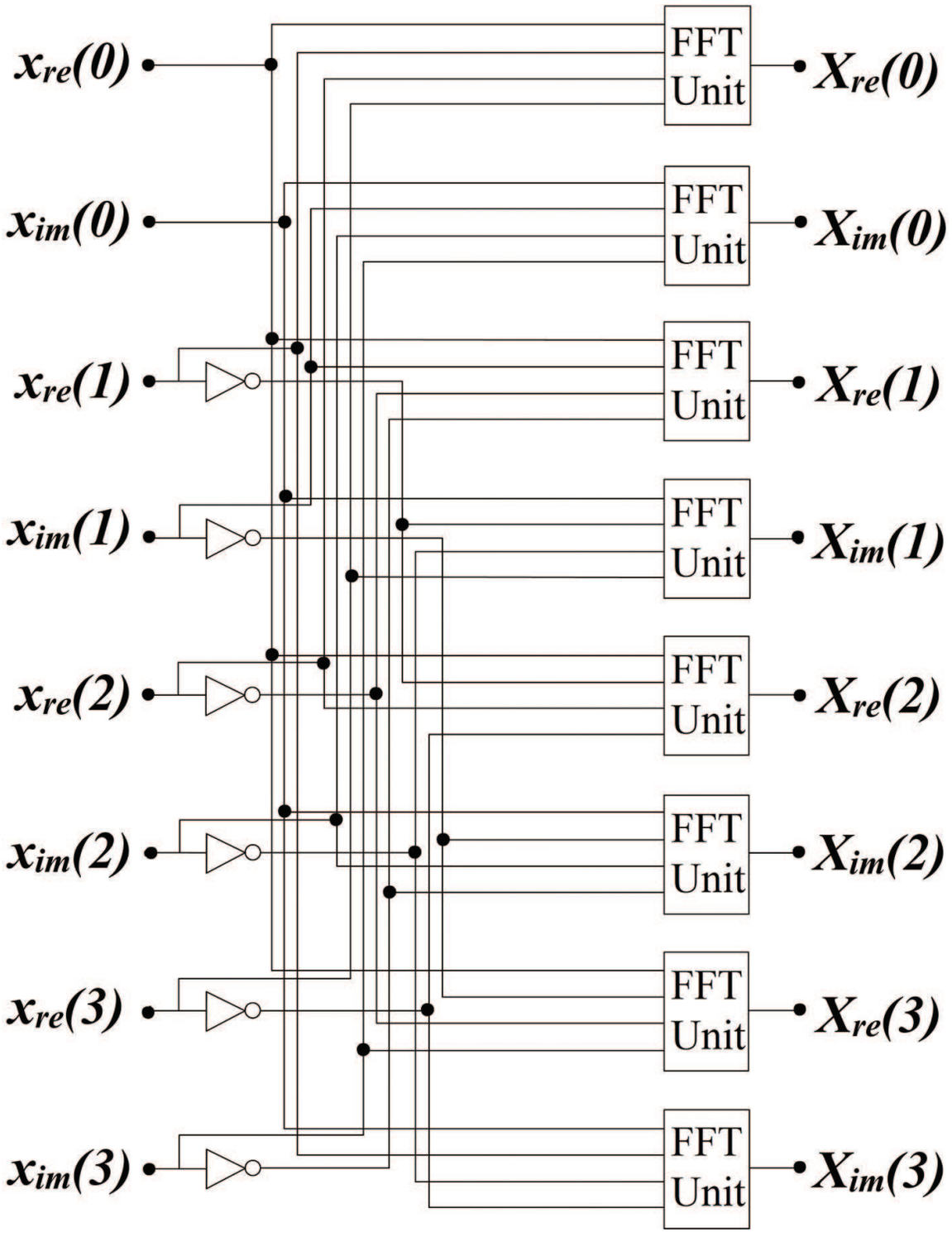

Figure 13. General block diagram of a four-point FFT circuit.

However, all other outputs of the four-point FFT are obtained by computations of the form Eq. (7), see **Figure 12**. So it consists of addition as well as subtractions. In order to implement subtraction, two's complement of the inputs is taken and is added. For taking 2's complement, first of all, the signals need to be inverted. This is obtained by the inverters shown in **Figure 13**.

Since all the inputs are of 8-bit length, 8-bit inverters are used in the circuit. The output of these inverters will be the 1's compliment of the input signals. To get 2's complement one must be added to the 1's compliment. Since two signals are to be complimented according to Eq. (7), two must be added to their sum. It is equivalent to adding one be the (LSB + 1)th place of their sum. This is done inside their corresponding FFT units, see **Figure 14**.

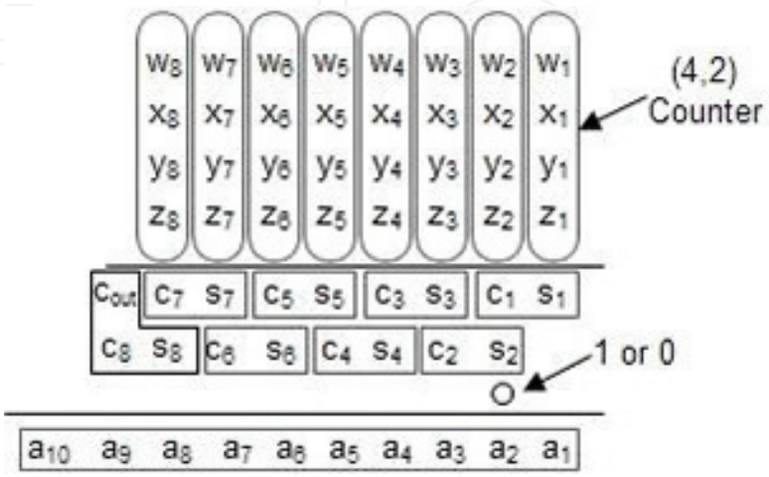

Figure 14. General block diagram of a FFT unit.

Let the four inputs to an FFT unit be denoted as w, x, y and z. All these inputs are 8-bit long. So the input w can be represented as $w_8 w_7 w_6 w_5 w_4 w_3 w_2 w_1$. Similarly, x, y and z can be represented as $x_8 x_7 x_6 x_5 x_4 x_3 x_2 x_1$, $y_8 y_7 y_6 y_5 y_4 y_3 y_2 y_1$ and $z_8 z_7 z_6 z_5 z_4 z_3 z_2 z_1$, respectively. For the FFT units that are computing real and imaginary parts of $X(0)$, namely $X_{re}(0)$ and $X_{im}(0)$, only additions are involved. Hence w, x, y and z need to be added to get $X(0)$. Addition of four binary numbers can be done using (4, 2)-counters or (4, 2)-compressors. So in order to implement a FFT unit, eight (4, 2)-counters are required.

The basic block diagram of a (4, 2)-counter is as shown in **Figure 15**. This circuit takes in four inputs and gives 3 outputs - sum, carry and carry out. A parallel arrangement of eight such (4, 2)-counters are needed to implement a single FFT unit. As seen from **Figure 15**, these (4, 2)-counters are implemented using the basic logic gates. All these logic gates are implemented using the proposed memristor-based resistive threshold logic.

The OR gate presented in **Figure 15** is implemented as shown in **Figure 16**, where A and B represent the inputs, $V_L + \Delta V$ represent the reference voltage and V_{OUT} represents the output of the gate.

AND gate can be obtained either from inverting the output of the NAND gate or by giving inverted inputs to the NOR gate. **Figure 17** shows the AND gate, used in **Figure 15** which is obtained by giving inverted inputs to the resistive threshold NOR gate.

XOR logic gate is implemented as shown in the circuit diagram of **Figure 18**. It is implemented using NOR logic. Thus, by using the circuits of **Figures 16–18**, a (4, 2)-counter can be implemented. By using a parallel implementation of eight such (4, 2)-counters, a single FFT unit can be implemented. For implementing a full FFT circuit, we require eight FFT units, four for the real parts and four for the imaginary parts of the outputs.

http://dx.doi.org/10.5772/66583

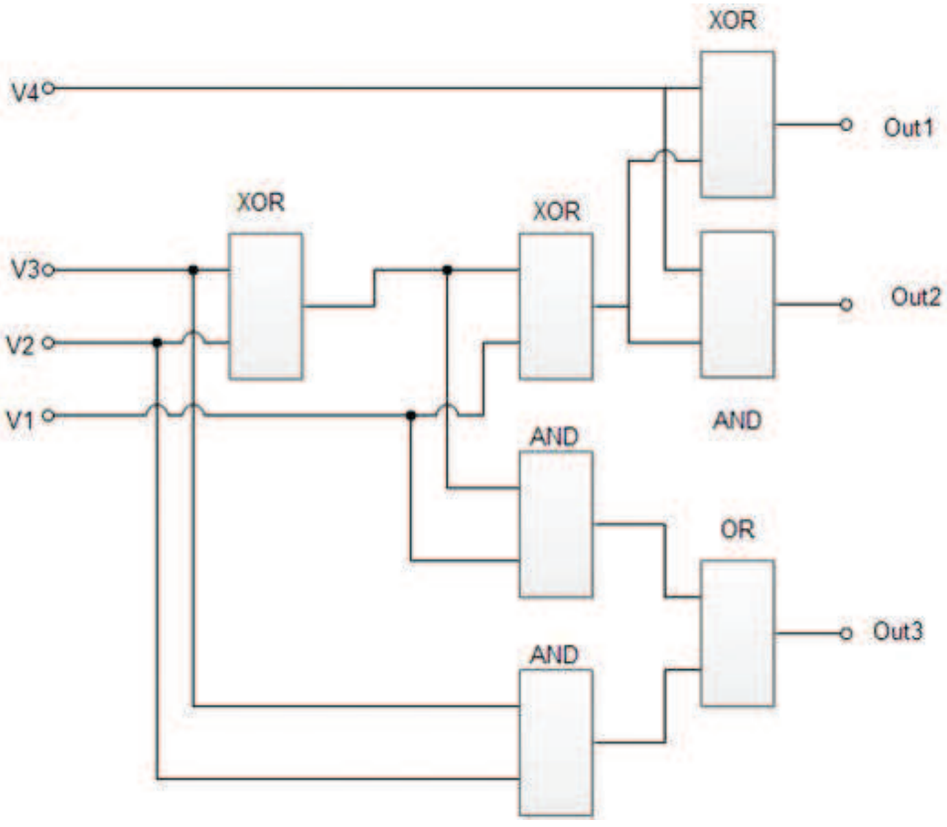

Figure 15. Block diagram of a single (4, 2)-counter.

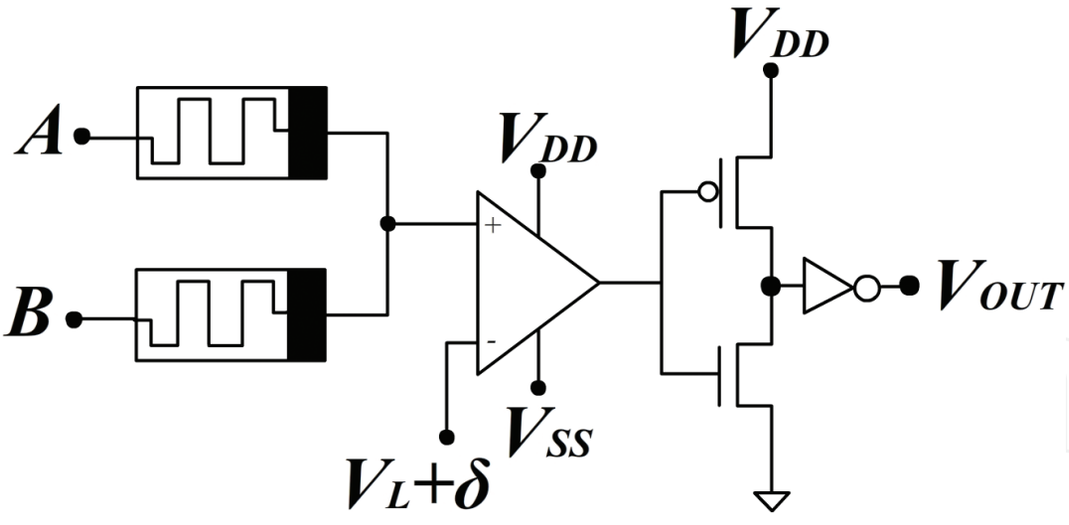

Figure 16. Circuit diagram of OR gate using memristor-based resistive threshold logic.

But while implementing the FFT units for real and imaginary parts of outputs $X(1)$, $X(2)$ and $X(3)$, the implementations involve subtraction. For subtractions, 2's complement of the subtrahend is taken and added. For taking 2's complement, first of all, the signal is inverted and then one is added. Inversion of the signal is carried out using an 8-bit inverter, as shown in **Figure 13**. The addition of one is done inside the FFT unit. Since two signals are to be subtracted per FFT unit, one has to be added twice which is equivalent to adding two or in binary terms, adding one to the (LSB+1)th position. This addition is shown in **Figure 14**.

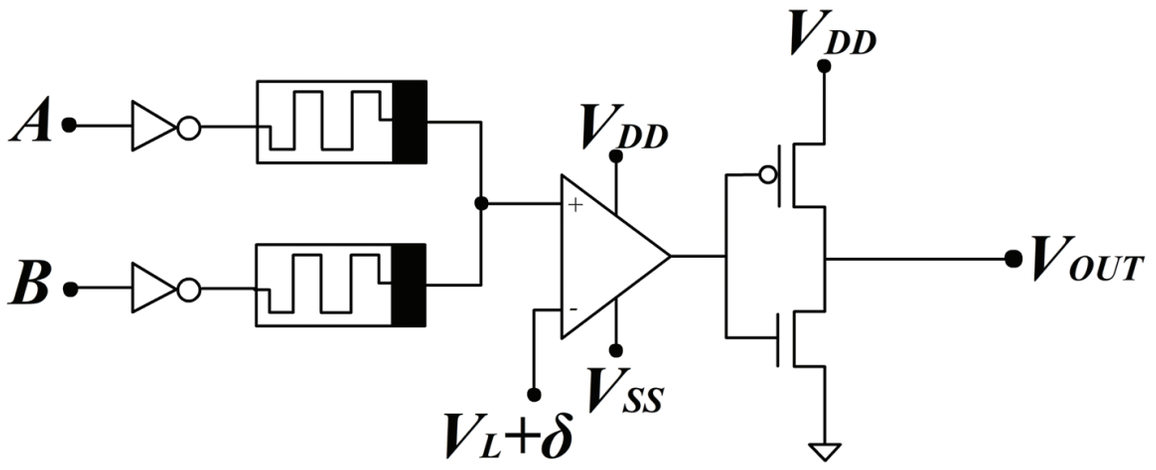

Figure 17. Circuit diagram of AND gate using memristor-based resistive threshold logic.

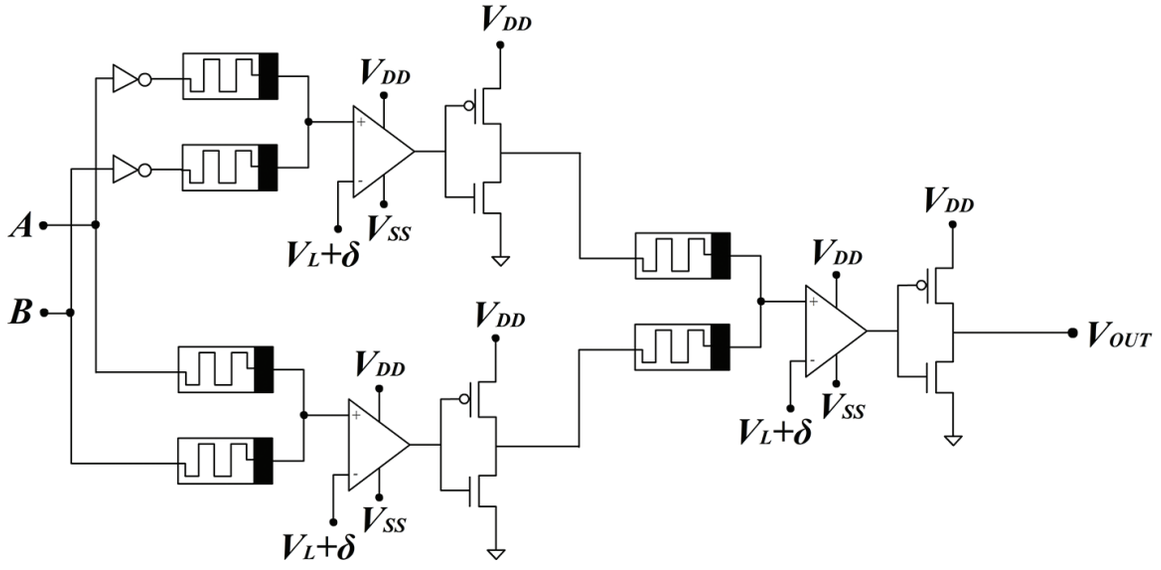

Figure 18. Circuit diagram of XOR gate using memristor-based resistive threshold logic.

6. Results and discussion

The proposed resistive threshold logic and FFT computing architecture have been tested and compared with corresponding CMOS [14], dynamic MOS and pseudo-NMOS circuits. Propagation delay, total harmonic distortion, area, power dissipation and leakage power were compared. For the fairness in comparison the technology, sizes of all components in the circuit are kept same.

Comparisons of the memristor-based resistive threshold logic with other logic families were carried out. **Tables 2** and **3** give the comparison results of propagation delay of NAND and NOR gates, respectively, implemented using various logic families. Comparison of resistive threshold logic with CMOS, pseudo-NMOS and dynamic-MOS logic families was carried out. The experiment was carried out for various numbers of inputs. The general trend in the

existing logic family is that the propagation delay of a gate increases as the number of inputs increases. This is due to the fact that the increase in the number of inputs means the increase in the number of input transistors. More transistors in the path lead to increase in the propagation delay of the gate.

But in the case of resistive threshold logic, this problem can be avoided. This is due to the fact that when the number of inputs increases, the number of memristors increases, but in a parallel fashion. This does not increase the path between input and output. Thus, the increase in the number of inputs does not affect the propagation delay of the overall circuit. This leads to lower propagation delay in the resistive threshold logic compared to other logic families, as the number of inputs increases. Comparison of propagation delay is shown for both NAND and NOR gates (see **Tables 2** and **3**).

Logic family	3i/p	10i/p	100i/p
CMOS logic	0.47 μs	0.54 μs	0.65 μs
Pseudo NMOS	0.48 μs	0.60 μs	0.85 μs
Dynamic MOS	0.48 μs	0.51 μs	0.75 μs
Resistive threshold logic	0.45 μs	0.45 μs	0.45 μs

Table 2. Comparison of propagation delay of NAND gate various logic families.

Logic family	3i/p	10i/p	100i/p
CMOS logic	0.50 μs	0.52 μs	0.66 μs
Pseudo NMOS	0.51 μs	0.58 μs	0.72 μs
Dynamic MOS	0.51 μs	0.58 μs	0.75 μs
Resistive threshold logic	0.60 μs	0.60 μs	0.60 μs

Table 3. Comparison of propagation delay of NOR gate various logic families.

Another comparison that shows the advantage of the proposed logic family is that of total harmonic distortion (THD). The total harmonic distortion gives the measure of harmonic distortion present in a circuit. Consider a signal of frequency x, the harmonics of the signals are signals containing frequencies $2x$, $3x$, $4x$ and so on. These harmonics cause distortion in the output of the circuit. The total amount of distortion caused by the harmonics is measured as total harmonic distortion. It can be defined as the ratio of powers of all harmonics to the power of the fundamental frequency.

Let the power of the fundamental frequency signal be P_1. Let the powers of the harmonics be $P_2, P_3, P_4, \ldots P_\infty$. Then the total harmonic distortion can be expressed as Eq. (8).

$$THD = \frac{\sum_{i=2}^{\infty} P_i}{P_1} \tag{8}$$

So higher the THD value means higher the distortion caused by the harmonics. When comparing the values in **Table 4**, it can be seen that the total harmonic distortion of the proposed

memristor-based resistive threshold logic is less when compared to other existing logic families. Thus, it can be seen that the proposed logic family has several advantages when compared to the existing logic families. Various comparisons were carried out for the fast Fourier transform circuits of **Figure 13**. **Table 5** gives the comparison of area of the FFT circuit implemented using different logic families.

Logic family	THD of NAND (%)	THD of NOR(%)
CMOS logic	71.2091	107.4098
Pseudo NMOS	76.2288	97.9864
Dynamic MOS	51.0608	146.7818
Resistive threshold logic	61.7043	85.2743

Table 4. Comparison of total harmonic distortion of NOR and NAND gate various logic families.

Logic family	Area (μm^2)
CMOS logic	83.5200
Pseudo NMOS	53.0496
Dynamic MOS	58.4852
Resistive threshold logic	58.1292

Table 5. Comparison of the total area of FFT circuits using various logic families.

The area comparison clearly shows that the circuit implements using the proposed logic family require less area compared to most of the other existing logic families. Another comparison was carried out regarding the power dissipation of the circuit, the results of which are shown in **Table 6**.

Logic family	Power
CMOS logic	19.3842 μW
Pseudo NMOS	14.6467 μW
Dynamic MOS	03.9515 nW
Resistive threshold logic	13.7932 μW

Table 6. Comparison of thetotal power dissipation of FFT circuits using various logic families.

The proposed logic family consumes less power when compared to other logic families. In **Table 6**, the power dissipation of the circuit using dynamic MOS is much less compared to other logic families owing to the fact that the dynamic MOS is a clocked logic family.

Now, **Table 7** shows the comparison of the FFT circuit implemented using various logic gates, in respect of their leakage power. The slight increase in the leakage power of the circuit using proposed logic family is due to the presence of op-amp in the circuit.

http://dx.doi.org/10.5772/66583

Logic family	Leakage power
CMOS logic	8.5377 nW
Pseudo NMOS	1.0629 mW
Dynamic MOS	4.8410 nW
Resistive threshold logic	11.1378 nW

Table 7. Comparison of leakage power of FFT circuits using various logic families.

As mentioned earlier, each of the FFT unit takes in four inputs, each of which is 8-bit long. **Figure 19** shows the 8th bit of the four inputs to a single FFT unit. The outputs of each FFT unit will be 10-bit long. **Figure 20** shows the LSB of the outputs of FFT unit for inputs shown in **Figure 19**.

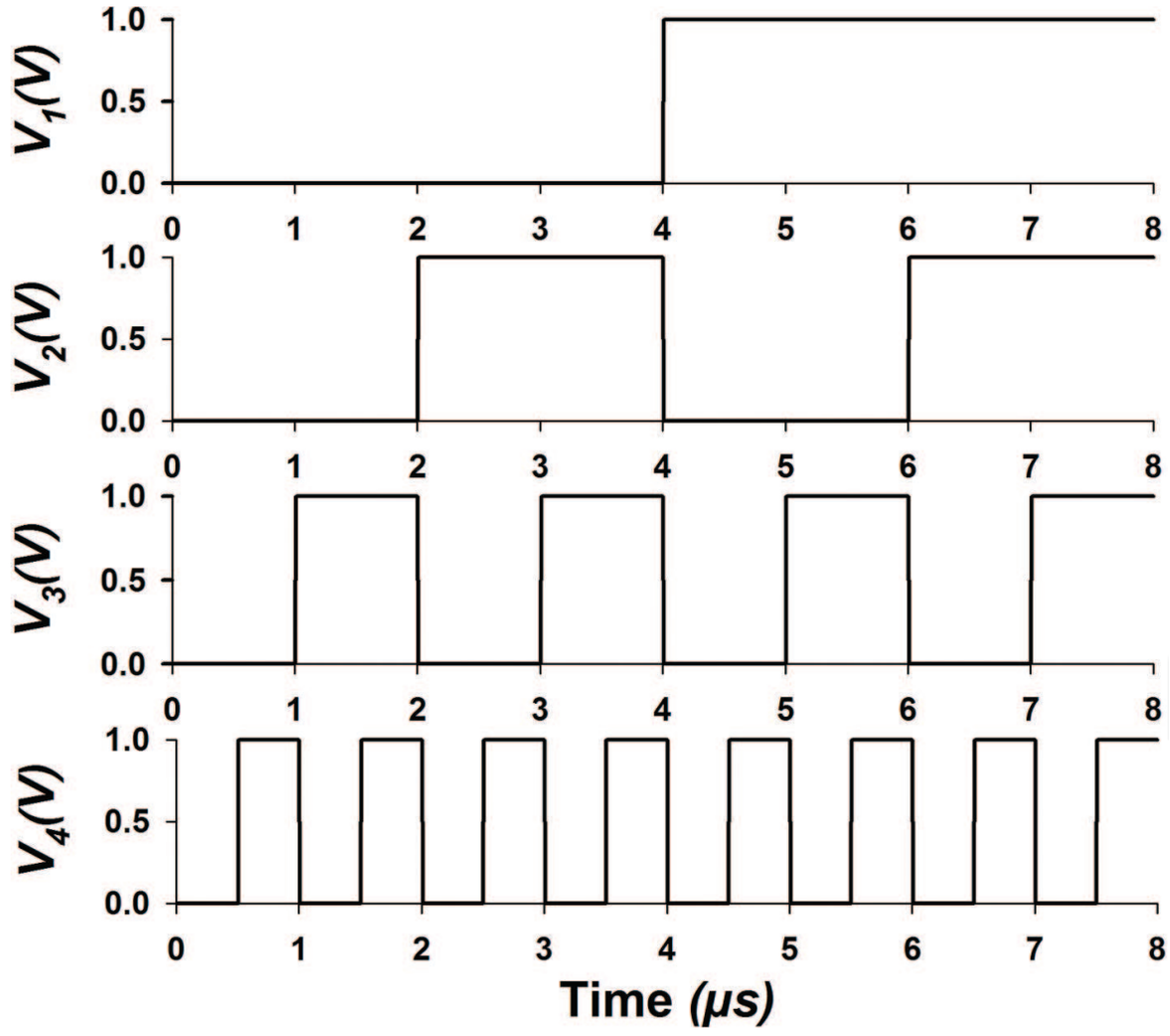

Figure 19. Eighth bit of the four inputs to a FFT unit.

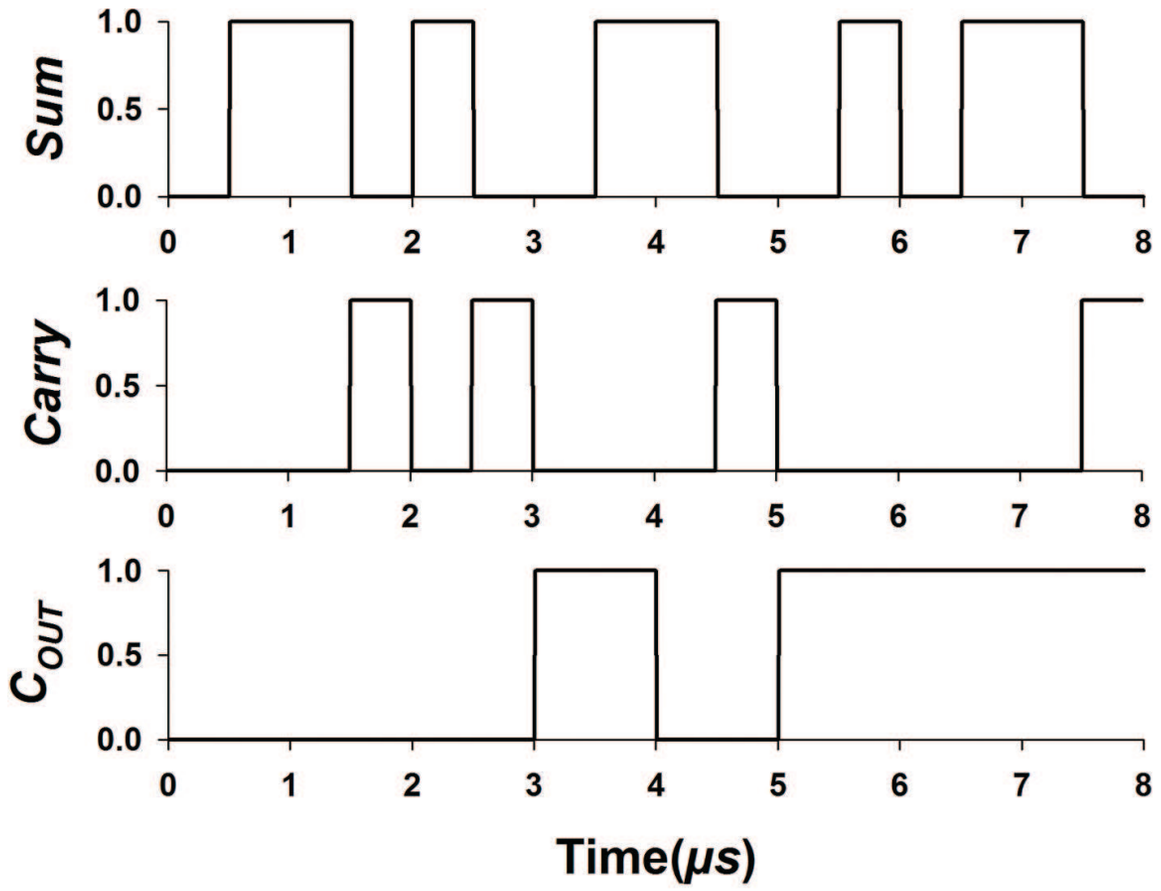

Figure 20. LSB of the output of the FFT unit is shown as sum. Carry and Cout of the LSB are also shown.

7. Discussion

In order to maintain practical relevance of the approach, all the results mentioned in the previous section are based on device parameters from 90 nm TSMC process. HP memristor models with R_{ON} = 100 Ω and R_{OFF} = 100 k Ω were used and all the simulations were carried out using LTSpice VI.

From the results shown in the previous section, it can be seen that the proposed logic family is clearly of advantage when we are trying to implement large input circuits. The comparison of propagation delay of the logic gates showed that the memristor-based resistive threshold logic family can be used to implement brain-like large input logic functions. Moreover, it gives a clear advantage in terms of the total harmonic distortion also. After the successful implementation of the new logic family, it was used to implement a circuit that could compute the fast Fourier transform of the given set of inputs. A four-point FFT circuit was implemented. It can take in four inputs at a time and give four FFT outputs. The circuit can handle complex numbers and the complex inputs were separated into real and imaginary part for the easy computation. All the inputs considered in this project were 8-bit long.

All the circuits and gates involved in the FFT circuit were implemented using the proposed memristor-based resistive threshold logic family. The speed of operation of the circuit can be further increased and the area can be further reduced if low power and high speed op-amp designs are developed, or an alternative thresholding circuit needs to be developed. If a better op-amp can be developed, then the leakage power can also be reduced to a great extent. Moreover, in the proposed logic, we are not concentrating on the memory power of memristor. If we can use that property of memristor in our architecture to save the data, then we can implement a memory unit also.

As a future work, the memory property of the memristor can be made use of using this property, a very efficient memory unit can be implemented. Since the area required for the memristor is very small, a very compact and area-efficient memory cell can be developed which can be used as an alternative for the existing memory cells. Also, only two-state logic gates are considered in this project. But in the future, by making use of the memory property of the memristor, multi-state logic functions can also be implemented. The op-amps used in this project can be replaced by more efficient and smaller op-amps, thereby, further reducing the area and leakage power and also increase the speed of operation of the circuit.

8. Summary

The concept of brain-like processing unit was presented using resistive threshold logic. The proposed memristor-based resistive threshold logic gate was used to implement circuits that could compute the fast Fourier transform of a given set of inputs. The proposed logic family was found too advantageous in many ways. However, there are certain disadvantages too, like the leakage power. Further development in the thresholding circuit will help to overcome this issue. In addition, the ability of the proposed architecture to perform brain-like processing can be seen as an early step in achieving the goal of mimicking brain-like processing unit in VLSI.

Author details

Alex Pappachen James

Address all correspondence to: apj@ieee.org

Bioinspired Microelectronic Systems Lab, School of Engineering, Nazarbayev University, Astana, Republic of Kazakhstan

References

[1] Zaidi ZF. Gender differences in human brain: a review. The Open Anatomy Journal. 2010;**2:37–55.**

[2] Borzenko A. Language processing in human brain. In: Proceedings of the First AGI Conference; March 1–3; Memphis. IOS Press; 2008.

[3] Chua LO. The missing circuit element. IEEE Tans. Circuit Theory. 1971;**18**:507–519.

[4] Williams RS. How we found the missing memristor. IEEE Spectrum. 2008;**45**:28–35.

[5] Williams RS, Strukov DB, Snider GS, Stewart DR. The missing memristor found. Nature. 2008;**453**:80–83.

[6] Joglekar YN, Wolf SJ. The elusive memristor: properties of basic electrical circuits. European Journal of Physics. 2009;**30**:661–675.

[7] Kim H, Sah MP, Yang C, Chua LO. Memristor-based multilevel memory. In: 12th International Workshop on Cellular Nanoscale Networks and their Applications (CNNA 2010); February 3–5; Berkeley, CA. pp. 1–6; 2010.

[8] Manem H, Rose GS, He X, Wang W. Design considerations for variation tolerant multilevel CMOS/Nano memristor memory. In: Proceedings of the 20th symposium on Great lakes symposium on VLSI; May 16; ACM; 2010. pp. 287–292.

[9] Laiho M, Lehtonen E. Arithmetic operations within memristor-based analog memory. In: 12th International Workshop on Cellular Nanoscale Networks and their Applications (CNNA 2010); Feb 3; IEEE; 2010. pp. 1–4.

[10] Xia Q, Robinett W, Cumbie MW, Banerjee N, Cardinali TJ, Yang JJ, Wu W, Li X, Tong WM, Strukov DB, Snider GS. Memristor—CMOS hybrid integrated circuits for reconfigurable logic. Nano Letters. 2009;**9**(10):3640–3645.

[11] James AP, Francis LR, Kumar DS. Resistive threshold logic. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2014 ;**22**(1):190–195.

[12] Maan AK, Jayadevi DA, James AP. A survey of memristive threshold logic circuits. IEEE Transactions on Neural Networks and Learning Systems. 2016;**99:**1–13**.**DOI: 10.1109/TNNLS.2016.2547842

[13] James AP, Dimitrijev S. Cognitive memory network. Electronics Letters. 2010;**46** (10):677–678.

[14] Petrick, S.R. A direct determination of the irredundant forms of a Boolean function from the set of prime implicants. Air Force Cambridge research Center, Cambridge. Techn. Rep. AFCRC-TR-56–110, 10 (1956).

[15] McGeer PC, Sanghavi JV, Brayton RK, Sangiovanni-Vicentelli AL. ESPRESSO-SIGNATURE: a new exact minimizer for logic functions. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 1993;**1**(4):432–440.

[16] James AP, Kumar DS, Ajayan A. Threshold logic computing: memristive-CMOS circuits for fast Fourier transform and vedic multiplication. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2015;**23**(11):2690–2694.