Article

Design and Application of Memristive Balanced Ternary Univariate Logic Circuit

Xiaoyuan Wang 1,2,*, Xinrui Zhang 2, Chuantao Dong 2, Shimul Kanti Nath 3 and Herbert Ho-Ching Iu 4

1 Wenzhou Institute of Hangzhou Dianzi University, Wenzhou 325024, China
2 School of Electronics and Information, Hangzhou Dianzi University, Hangzhou 310018, China; z120431540@163.com (X.Z.); crazypink@163.com (C.D.)
3 School of Photovoltaic and Renewable Energy Engineering, University of New South Wales (UNSW Sydney), Kensington, NSW 2052, Australia; shimul_kanti.nath@unsw.edu.au
4 School of Engineering, The University of Western Australia, Crawley, WA 6009, Australia; herbert.iu@uwa.edu.au
* Correspondence: youyuan-0213@163.com

Abstract: This paper proposes a unique memristor-based design scheme for a balanced ternary digital logic circuit. First, a design method of a single-variable logic function circuit is proposed. Then, by combining with a balanced ternary multiplexer, some common application-type combinational logic circuits are proposed, including a balanced ternary half adder, multiplier and numerical comparator. The above circuits are all simulated and verified in LTSpice, which demonstrate the feasibility of the proposed scheme.

Keywords: memristor; balanced ternary; univariate logic; combinational logic circuit

1. Introduction

In the era of big data, the amount of data is growing explosively, and as a result, digital logic systems are having difficulty in processing such huge amounts of data while striving for ever-increasing efficiency [1]. To meet the demand of data processing speed and power efficiency, ternary logic has received recent attention due to its advantages of higher single-line information carrying capacity and additional logical functions [2–7]. Compared to the binary digital signal, each bit of the ternary digital signal contains more information, resulting in a higher transmission rate at the same frequency. It also helps in reducing circuit interconnections, and digital chips can be made smaller and less expensive [8–10]. Ternary logic can be divided into two categories: balanced ternary {−1, 0, 1} and unbalanced ternary {0, 1, 2} or {0, −1, −2} [11]. Among them, balanced ternary logic has unique advantages, including the ability of having a unified representation for positive and negative numbers without the sign bit, and multiplication operation without generating a carry. Moreover, the symmetry of one-bit addition and multiplication operations can be used for symmetric arithmetic operation circuit design [12,13].

In recent years, ternary digital logic circuits have been implemented in various technologies, including MOSFETs, carbon nanotube field effect transistors (CNTFETs), resonant tunneling diodes (RTD), single-electron transistors, memristors, etc. [14–18]. Among them, memristor-based ternary logic is of considerable interest, as it provides the advantages of non-volatility, nanoscale and compatibility with CMOS technology [19,20].

There are two typical paradigms for designing memristor-based ternary logic circuits; one uses three resistance states of the ternary memristor and the other one uses the voltage value as the logic variable, where the former method makes full use of the resistance change characteristics of the memristor, the operation result can be stored in memristors, the logic state will not be lost after power withdrawal. Several studies [21,22]
reported on the unbalanced ternary basic logic gate circuit using the three resistance states of the ternary memristor which correspond to positive ternary logic ‘0’, ‘1’, and ‘2’. A voltage-controlled tri-valued memristor model was first proposed in Ref. [22], with designs of ternary AND, OR and NOT gate circuits based on it. In this case, three stable resistance states, $R_H$, $R_M$ and $R_L$, correspond to logic ‘0’, ‘1’, and ‘2’, respectively. In Ref. [23], a bipolar three-state ZnO memristor was reported, and then all the 27 possible univariate positive ternary logics were realized with a single memristor cell. Furthermore, Ref. [24] proposed a method of realizing a balanced ternary adder using the resistance state transformation of only one single memristor, in which the circuit area and system power consumption were greatly reduced.

Significant advancement has also been achieved in implementing logic circuits using the second method (i.e., employing the voltage value as a logic variable) [25,26]. For example, Wang et al. [27] reported the construction of positive ternary logic circuits, including, the ternary AND gate, OR gate, inverters, encoder and decoder circuits. Similarly, in Ref. [28], ternary basic logic gates and combinational logic circuits using memristor-CNTFET hybrid circuit were proposed, whose delay and circuit complexity were lower compared to those of the circuits only using CNTFETs. Ref. [29] proposed a systematic method of constructing a two-digit ternary logic function based on the concept of memristive threshold logic (MTL) and applied this method for constructing basic ternary arithmetic operations. Compared to that of the previously reported relevant circuit design schemes, the circuit area of the ternary adder and ternary multiplier was greatly reduced. In Refs. [5,30], the balanced ternary logic circuits based on a memristor and MOSFET were proposed. The design idea was to construct balanced ternary essential logic gates, such as TAND, TOR, TI, TSUM, NCONS, NANY, etc., and then propose design scheme of a balanced ternary full adder.

As a further development in the present study, combinatorial logic circuits are implemented directly by combining univariate logic circuits and multiplexers. The multiplexer uses the circuit proposed in Ref. [31], and its function is to select only one of the data of multiple channels and transmit it to the output terminal according to the state of the selection signal. The proposed design scheme of a memristive balanced ternary digital logic circuit with the voltage value as the logic variable could be beneficial for further improving information storage, processing, and transmission efficiency.

The structure of this paper is as follows: Section 2 presents a design scheme of a balanced ternary single-variable logic function circuit based on a hybrid design of memristor and MOS transistor; in Section 3, based on the proposed univariate logic circuits and the multiplexer designed in our previous study [31], balanced ternary application-type combinational logic circuits are designed, including a half adder, multiplier, and numerical comparator; Section 4 presents the comparison and analysis of the proposed circuit with existing designs; Section 5 contains the conclusion of this paper.

### 2. Balanced Ternary Univariate Logic Circuit

In digital logic circuits, univariate logic functions are used to perform corresponding logic transformations on signals, thus playing an important role in circuit design. For ternary logic, there are three possible values for a single-input variable, with $3^3 = 27$ possible output results in total, as shown in Table 1.
Table 1. Balanced ternary univariate logic function truth table.

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>F₁</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
</tr>
</tbody>
</table>

As evident from Table 1, balanced ternary univariate logic can be divided into three categories, such as three-state to one-state logic, three-state to two-state logic, and three-state to three-state logic. The first category (three-state to one-state logic: F₁, F₁₄, and F₂₇) is also called constant logic; that is, irrespective of the input value, the output is a fixed logic state, and therefore their applications are limited in circuit design.

This paper will mainly involve the circuit design of the other two categories, i.e., the balanced ternary three-state to two-state logic, and the three-state to three-state univariate logic, as well as a detailed analysis and simulation verification of the corresponding circuits. All univariate logic circuits are represented by the circuit symbol shown in Figure 1.

Figure 1. Circuit symbol of univariate logistic function Fₙ.

2.1. Three-State to Two-State Logic

From the truth table of the balanced ternary univariate logic function shown in Table 1, there are 18 kinds of univariate logic functions for three-state to two-state logic. Among them, the logics of F₁₉ and F₂₅ correspond to the NTI gate and the PTI gate, respectively, which have been introduced in detail in Ref. [31] and will not be repeated in this section.

2.1.1. Circuit Design of Logic Function F₄, F₅, F₉, F₁₀, F₁₃, F₁₈, F₂₃ and F₂₆

Table 2 shows the designed circuit diagram and the threshold voltage range of the MOS transistor. While the circuits of logic functions F₄ and F₅ only need one memristor and one NMOS transistor, those of the logic functions F₉, F₁₀, F₁₃, F₁₈, F₂₃ and F₂₆ are all composed of two memristors and one NMOS transistor. Among them, two groups of logic (F₁₀ and F₁₃) and (F₂₃ and F₂₆) adopt the same circuit structure, but the difference is that the threshold voltage ranges of MOS transistors in the corresponding circuits are different. See Table 2 for details.
Table 2. Structure diagram of three-state to two-state logic circuit and threshold voltage of MOS transistor.

<table>
<thead>
<tr>
<th>Logic Function</th>
<th>$F_4$</th>
<th>$F_5$</th>
<th>$F_9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circuit Structure</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOS Transistor</td>
<td>$V_{DD} &lt; V_{th1} \leq 2V_{DD}$</td>
<td>$V_{DD} &lt; V_{th1} \leq 2V_{DD}$</td>
<td>$V_{DD} &lt; V_{th1} \leq 2V_{DD}$</td>
</tr>
<tr>
<td>Threshold Voltage</td>
<td>$V_{DD} &lt; V_{th1} \leq 2V_{DD}$</td>
<td>$V_{DD} &lt; V_{th1} \leq 2V_{DD}$</td>
<td>$V_{DD} &lt; V_{th1} \leq 2V_{DD}$</td>
</tr>
</tbody>
</table>

The working principles of these logic functions can be understood via simply analyzing the circuits of logic functions $F_4$ and $F_5$. For $F_4$, when input $A$ is $-V_{DD}$ (logic ‘−1’) or $0V$ (logic ‘0’), transistor $T_1$ is turned off, and the output terminal will be directly connected to the input terminal through memristor $M_1$, so the output remains consistent with the input. When input $A$ is $V_{DD}$ (logic ‘1’), transistor $T_1$ is turned on, and the output terminal will be directly connected to $-V_{DD}$ through $T_1$, that is, logic ‘−1’ is the output. For $F_5$, when input $A$ is $-V_{DD}$ (logic ‘−1’) or $0V$ (logic ‘0’), transistor $T_1$ is turned off, the output terminal will pass through memristor $M_1$, which is directly connected to the input terminal, and the output is consistent with the input. When input $A$ is $V_{DD}$ (logic ‘1’), transistor $T_1$ is turned on, and there is a current path flowing from the input terminal to $-V_{DD}$ in the circuit. Both memristors $M_1$ and $M_2$ are switched to the $R_{OFF}$ state, and the output terminal is about 0 V after voltage division, that is, the output logic is ‘0’. Similar methods can be used to verify the correctness of other circuits, which will not be repeated here.

2.1.2. The Circuit Design of the Remaining Three-State to Two-State Logic Function

The remaining three-state to two-state logic function circuits, including $F_2$, $F_3$, $F_7$, $F_11$, $F_{15}$, $F_{17}$, $F_{21}$, and $F_{24}$ logic, can be obtained via cascading the circuits as mentioned above. For example, for the $F_2$ logic circuit, it is only necessary to cascade an $F_4$ logic circuit after the $F_{26}$ logic circuit to complete the logic conversion corresponding to $F_2$. As shown in Table 3, it is a design scheme of a single-variable three-state to two-state logic circuit designed via the cascade method. Among them, ‘$F_m + F_n$’ indicates that the $F_n$ logic circuit is cascaded after the $F_m$ logic circuit.
Table 3. The scheme of univariate three-state to two-state logic circuit designed via the cascade method.

<table>
<thead>
<tr>
<th>Logic Function</th>
<th>$F_2$</th>
<th>$F_3$</th>
<th>$F_7$</th>
<th>$F_{11}$</th>
<th>$F_{15}$</th>
<th>$F_{17}$</th>
<th>$F_{21}$</th>
<th>$F_{24}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Composition</td>
<td>$F_{26} + F_4$</td>
<td>$F_{25} + F_{19}$</td>
<td>$F_4 + F_9$</td>
<td>$F_4 + F_{10}$</td>
<td>$F_{25} + F_{26}$</td>
<td>$F_4 + F_{18}$</td>
<td>$F_4 + F_{19}$</td>
<td>$F_4 + F_{23}$</td>
</tr>
</tbody>
</table>

2.1.3. Simulation Verification of Three-State to Two-State Logic Circuit

To validate the above approach, the proposed circuit is simulated and verified using LTSpice. Figures 2–4 show the simulation waveforms of three kinds of three-state to two-state logic, including the transition from the three-state logic circuits to logic $(-1,1)$, $(-1,0)$ and $(0,1)$.

![Figure 2](image-url)  
**Figure 2.** Simulation results of transition from three-state logic circuits to logic $(-1,1)$.

![Figure 3](image-url)  
**Figure 3.** Simulation results of transition from three-state logic circuits to logic $(-1,0)$.

![Figure 4](image-url)  
**Figure 4.** Simulation results of transition from three-state logic circuits to logic $(0,1)$.

2.2. Three-State to Three-State Logic

There are 6 types of single-variable logic functions in this category, including $F_6$, $F_8$, $F_{12}$, $F_{16}$, $F_{20}$ and $F_{22}$. Among them, the output of $F_6$ is equal to the input, which is called ‘follower logic’. Only five types of three-state to three-state logic are effective and used in
circuit design. However, the $F_{22}$ logic (STI gate) circuit has been discussed in detail previously [31], and the remaining four logic circuits will be introduced here.

2.2.1. Circuit Design of Up-Spin Logic Function $F_{16}$ and Down-Spin Logic Function $F_{20}$

The circuit structure diagram of the up-spin logic function, $F_{16}$, the down-spin logic function, $F_{20}$, and the threshold voltage range of the MOS transistor used is shown in Table 4. The $F_{16}$ circuit uses two memristors and two NMOS transistors, while the $F_{20}$ circuit uses three memristors and three NMOS transistors. In the case of $F_{20}$, when input $A$ is $-V_{DD}$ (logic ‘$-1$’), MOS transistors $T_1$, $T_2$, and $T_3$ are all turned off, and the output terminal is pulled up to $V_{DD}$ through memristor $M_1$, that is, the output logic is ‘1’. When input $A$ is $0$ V (logic ‘0’), both $T_1$ and $T_2$ are turned off, $T_3$ is turned on, and the output terminal is directly connected to $-V_{DD}$ through $T_2$, that is, the output logic is ‘$-1$’. When input $A$ is $V_{DD}$ (logic ‘1’), both $T_1$ and $T_2$ are turned on, $T_3$ is turned off, and there is a current path from $V_{DD}$ to $-V_{DD}$ in the circuit. Both memristors $M_1$ and $M_2$ are switched to the $R_{OFF}$ state, and the output terminal outputs a voltage nearly $0$V, that is, the output logic is ‘0’. The correctness of spin-up logic function $F_{16}$ can be verified via a similar method, which will not be repeated here.

Table 4. Circuit structure diagram of up-spin logic function, down-spin logic function and threshold voltage of MOS transistor.

<table>
<thead>
<tr>
<th>Logic Function</th>
<th>Up-Spin Logic Function, $F_{16}$</th>
<th>Down-Spin Logic Function, $F_{20}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circuit Structure</td>
<td><img src="image" alt="Circuit Structure" /></td>
<td><img src="image" alt="Circuit Structure" /></td>
</tr>
<tr>
<td>MOS Transistor Threshold Voltage</td>
<td>$T_1: V_{th1} &gt; V_{DD}$</td>
<td>$T_1: V_{th1} &gt; V_{DD}$</td>
</tr>
<tr>
<td></td>
<td>$T_2: V_{th2} \leq 2V_{DD}$</td>
<td>$T_2: V_{th2} \leq 2V_{DD}$</td>
</tr>
<tr>
<td></td>
<td>$T_3: 0V &lt; V_{th3} \leq V_{DD}$</td>
<td>$T_3: 0V &lt; V_{th3} \leq V_{DD}$</td>
</tr>
</tbody>
</table>

2.2.2. The Circuit Design of the Remaining Three-State to Three-State Logic Function

The remaining three-state to three-state logic function circuits, including $F_8$ and $F_{12}$ logic, can also be obtained via cascading the circuits mentioned above. For example, for the $F_8$ logic circuit, it is only necessary to cascade an $F_{22}$ logic circuit after the $F_{20}$ logic circuit to complete logic conversion corresponding to $F_8$. Similarly, the $F_{12}$ logic circuit can be obtained via cascading $F_{16}$ logic and $F_{22}$ logic. Table 5 shows the design scheme of the univariate three-state to three-state logic circuit using the cascade method. The term ‘$F_m + F_r$’ indicates that the $F_r$ logic circuit is cascaded after the $F_m$ logic circuit.

Table 5. Design scheme of univariate three-state to three-state logic circuit designed via cascade method.

<table>
<thead>
<tr>
<th>Logic Function</th>
<th>$F_8$</th>
<th>$F_{12}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Composition</td>
<td>$F_{20} + F_{22}$</td>
<td>$F_{16} + F_{22}$</td>
</tr>
</tbody>
</table>
2.2.3. Verification of Three-State to Three-State Logic Circuit Using LTSpice Simulation

The above circuit was simulated in LTSpice, which provides a verification of the design for a given input signal. The simulation waveform diagram of the three-state to three-state logic circuit is shown in Figure 5.

![Figure 5](image)

**Figure 5.** Simulation waveform diagram of three-state to three-state univariate logic circuit.

3. Design of Balanced Three-Valued Combinational Logic Circuit Based on Univariate Logic and Multiplexer

A multiplexer can select one of several input signals to the output. This paper uses the balanced ternary multiplexer circuit proposed in Ref. [31], which can realize the output of one signal from the three inputs. The corresponding input–output relationship is expressed as follows:

\[
Y = \begin{cases} 
I_1 & S = -1 \\
I_0 & S = 0 \\
I_1 & S = 1 
\end{cases}
\]  

(1)

Here, \( S \) is a selection signal, and \( I_1, I_0, \) and \( I_1 \) are three input signals. The multiplexer is composed of a balanced ternary one-line–one-line decoder, three balanced ternary minimum gates and one balanced ternary maximum gate. The circuit structure diagram is shown in Figure 6.

![Figure 6](image)

**Figure 6.** (a) Circuit diagram of balanced ternary multiplexer (b) symbol.

When the selection signal is \( S = -1 \), the output terminals \( S_1, S_0 \) and \( S_1 \) of the one-line–three-line decoder output logic \( 1, -1 \) and \( -1 \), respectively. According to the working principle of the minimum value gate and the maximum value gate, the output signal of the circuit is equal to input signal \( I_1 \), that is, \( Y = I_1 \), and the circuit realizes the function of output signal \( I_1 \). When the selection signal \( S = 0 \), the output terminals \( S_1, S_0 \) and \( S_1 \) of the decoder output the logic \( -1, 1, -1 \), respectively. In this case, \( Y = I_0 \), that is, the circuit realizes the function of outputting signal \( I_0 \). Finally, when the selection signal \( S = 1 \) occurs, the output of decoder terminals \( S_1, S_0 \) and \( S_1 \) output the logic \( -1, -1, \) and \( 1 \), respectively, resulting in \( Y = I_1 \).
In this paper, a balanced ternary half adder, a balanced ternary multiplier and a balanced ternary numerical comparator are also designed using the multiplexer and the univariate logic circuit described in Section 2. The truth tables and circuit structures of these applications are summarized in Table 6 and Table 7, respectively. The design process and working principle of each circuit are explained in the following three subsections, along with the corresponding simulation results.

Table 6. Truth table of balanced ternary half adder, balanced ternary multiplier, and balanced ternary numerical comparator.

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Half Adder</td>
<td>Multiplier</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 7. Circuit structure of each application.

3.1. Balanced Ternary Half Adder

It can be seen from the truth table that when input signals $A = -1$ and $B$ selects $f$ the values of $\{-1, 0, 1\}$, the ‘SUM’ outputs the sum of the half adder outputs, corresponding to $[1, -1, 0]$. According to the working principle of the multiplexer, if $A$ is used as the selection signal, we can obtain the following results. When $A = -1$, the multiplexer selects the $I_1$ input terminal for the output, that is, $SUM = I_1$. And as shown in Table 1 a univariate logic $F_{20}$ just can fulfill the conversion demanded in the red square in Table 6, so $F_{20}$ is selected to connect the input B and $I_1$ in the circuit. Similarly, when input signal $A = 0$, the sum output of the half adder is $SUM = I_0 = B$, so we directly connect B to $I_0$. When input signal $A = 1$, $SUM = I_1$, the logic $F_{16}$ is consistent with the conversion, so $F_{16}$ is selected to connect the input B and $I_1$ in this case. The ‘CARRY’ output circuit part is designed in the same way. Figure 7 shows the corresponding logic conversion diagram of the balanced ternary half adder.
According to the univariate logic function relationship in Table 1, for the ‘sum’ output part, the three logic conversion relationships correspond to the down-spin logic function, $F_{20}$, the follow-up logic function, $F_{0}$, and the up-spin logic function, $F_{16}$. For the ‘carry’ output part, the three logical conversion relationships correspond to the logical functions $F_{5}$, $F_{14}$, and $F_{15}$. Therefore, it is only necessary to introduce the corresponding univariate logic circuit into circuit design. The LTSpice simulation waveform diagram is given in Figure 8.

**Figure 8.** Simulation waveform diagram of half adder.

### 3.2. Balanced Ternary Multiplier

Balanced ternary does not generate carry during multiplication, so it has certain advantages over the unbalanced ternary logic. The multiplier circuits design is as follows: When $A = -1$, the multiplexer selects the $I_{-1}$ input terminal for the output, According to Tables 1 and 6, $F_{22}$ can be selected to connect the input $B$ and $I_{-1}$ in the circuit. When $A = 0$, the $I_{0}$ terminal of the multiplexer is gated, and now the output terminal outputs a logic ‘0’, so we can directly connect $I_{0}$ to the ground. When $A = 1$, the logic value of the output terminal is consistent with the input signal $B$, so we connect input signal $B$ to the $I_{1}$ terminal of the multiplexer in this case. Figure 9 shows the LTSpice simulation waveform diagram of the circuit.
3.3. Balanced Ternary Numerical Comparator

As we known, the output of multiplexer equals to $I_{-1}$ when the input signal $A$ is selected as $-1$, that is, $MLE = I_{-1}$. And according to the truth Tables 1 and 6, logic $F_{10}$ performs the same function when input $A = -1$. So $F_{10}$ is selected to connect the input $B$ and $I_{-1}$ in the circuit. Similarly, logics $F_{22}$ and $F_{26}$ are chosen to perform the corresponding functions when input $A = 0$ and $A = 1$. Figure 10 shows the simulation results for a balanced ternary numerical comparator.

4. Comparison and Analysis

The number of components using the proposed method are given in Table 8 and are compared with that reported earlier [31]. It is evident that there are significant advantages of the proposed method in terms of the balanced ternary half adder, multiplier, and numerical comparator circuit as the number of circuit components is reduced by 37.8%, 39.5%, and 48.2%, respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>Components</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>THA</td>
</tr>
<tr>
<td>Method in This Paper</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>(13T33M)</td>
</tr>
<tr>
<td>Multiplexer-Based Method in [31]</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>(10T64M)</td>
</tr>
</tbody>
</table>

A comparison of power consumptions of the circuits in Ref. [31] is also given in Table 9, including static power consumption, average power consumption and dynamic power.
consumption. The static power consumption shown in this table is the maximum static power consumption value of all the nine input combinations, and the average power consumption is the average static power consumption for every input combination. The dynamic power dissipation was estimated and calculated in accordance with the following formula:

\[ P(\text{dynamic}) = |P(\text{max}) - P(\text{avg})| \]  

where \( P(\text{max}) \) means the instantaneous maximum power consumption, and the \( P(\text{avg}) \) is the average power dissipation, which can be obtained through SPICE simulations.

It is evident that, there are no significant advantages over the method reported in Ref. [31] for the balanced ternary half adder, multiplier and numerical comparator circuits. Particularly, THA's static power consumption exceeds about three times that in Ref. [31]. This is because we use three more transistors with relatively higher power consumption than that in Ref. [31]. However, our dynamic power consumption is relatively lower for our present study, showing the significance of the current approach.

Table 9. Comparison of power consumption statistics of designed circuits.

<table>
<thead>
<tr>
<th>Method in this paper</th>
<th>Avg. Power (uW)</th>
<th>Static Power (uW)</th>
<th>Dynamic Power (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>THA</td>
<td>MUL</td>
<td>MLE</td>
</tr>
<tr>
<td>Method in this paper</td>
<td>246.99</td>
<td>72.84</td>
<td>0.31</td>
</tr>
<tr>
<td></td>
<td>[-1&amp;1]</td>
<td>[-1&amp;1]</td>
<td>[0&amp;−1]</td>
</tr>
<tr>
<td>Method in [27]</td>
<td>72.65</td>
<td>72.84</td>
<td>0.56</td>
</tr>
<tr>
<td></td>
<td>[-1&amp;1]</td>
<td>[0&amp;1]</td>
<td>[0&amp;−1]</td>
</tr>
</tbody>
</table>

5. Conclusions

In summary, a design scheme for a balanced ternary logic circuit based on a memristor and MOS transistor was proposed. At first, the design of a balanced ternary single-variable logic circuit was introduced, including the commonly used three-state to two-state logic and three-state to three-state logic. Then, combined with the balanced ternary multiplexer, several design schemes of application-type combinational logic circuits were proposed, including a balanced ternary half adder, multiplier and numerical comparator. The designed circuits were simulated and further verified using LTSpice. Finally, the proposed circuit was compared with other design methods. Our results show that the number of components can be significantly reduced using the proposed design method, which could further reduce the complexity of the circuit.

Author Contributions: Conceptualization, X.W.; methodology, X.W., C.D. and X.Z.; software, C.D. and X.Z.; validation, X.W., X.Z. and C.D.; formal analysis, X.Z. and C.D.; investigation, X.Z. and C.D.; resources, X.W.; data curation, C.D.; writing—original draft preparation, C.D.; writing—review and editing, X.W., S.K.N. and H.H.-C.I.; supervision, X.W.; project administration, X.W.; funding acquisition, X.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the [National Natural Science Foundation of China] grant number [61871429], and the [Natural Science Foundation of Zhejiang Province] grant number [LY18F010012].

Data Availability Statement: The datasets generated and analyzed during the current study are available from the corresponding author on reasonable request.

Acknowledgments: This work was supported in part by the National Natural Science Foundation of China under Grant 61871429, in part by the Natural Science Foundation of Zhejiang Province under Grant LY18F010012.

Conflicts of Interest: The authors declare no conflict of interest.
References


**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.