International Journal on Advanced Science Engineering Information Technology

Vol.11 (2021) No. 4
ISSN: 2088-5334

A New Approach for Designing of Computer Architectures Using Multi-Value Logic

Alessandro Simonetta[a,*], Maria Cristina Paoletti[a], Maurizio Muratore[a]

[a] *Department of Enterprise Engineering, University of Rome "Tor Vergata", Via del Politecnico, 1, Rome, 00133, Italy*
*Corresponding author: *alessandro.simonetta@gmail.com*

Abstract— **In the last decade, we have seen how Moore's law has lost its validity because it has reached the physical limit of miniaturization of components and the problem of thermal dissipation increasing with the chip's clock frequency. For this reason, multi-core architectures were born, and quantum computing is being looked at as a possible solution for more computing power. The aim of this article is to demonstrate the possibility to realize computer architectures using multi-value logic. The objective is reached when we are able to translate any MVL function into the corresponding MVL circuit. The method proposed is completely independent of the basis adopted in the MVL domain and the physical quantity used to represent or transfer the domain values. Thus, this approach provides a new theoretical and practical context for applying new technologies or polymorphic materials, which can represent multiple values. In order to give concreteness to the analyzed theoretical aspects, we will represent a case study based on a classical combinational circuit, the summing circuit using LTspice® XVII (© Analog Device Corporation) as simulation environment has been carried out. The results show that it is possible to build MVL combinatorial circuits, although there are limitations because the proposed solution is just a proof of concept. The method can also be successfully applied to sequential MVL circuits, which are not the subject of the present article.**

Keywords— **Multi-valued logic; computer architecture; linear algebra; digital circuits; combinatorial circuit.**

Manuscript received 27 Feb. 2021; revised 15 Jul. 2021; accepted 10 Aug. 2021. Date of publication 31 Aug. 2021.

I. INTRODUCTION

Digital circuits are used in many areas, including telecommunications, signal processing, biomedical, and machine learning [1]–[4]. Although multi-value logic, or simply MVL, has been the subject of much research [5]–[7] as a possible evolution of two-based systems, there has been no realization that it can supplant a traditional digital processing system. Some think that there are technical and economic reasons that do not permit its success [8]. Others, however, believe it is desirable to use multi-valued logic for quantum computing [9] to promote the interpretability of machine-learning systems' decisions without compromising its computational efficiency [10]. It must be said that technology has made great strides, and now the construction of electronic devices that are capable of supporting multiple states is a reality [11], [12]. Initially, neglecting the problem of physical implementation and assuming that we maintain a discrete coding system, is it fair to ask what the best basis for a computing system is?

If we target processing systems comparable to human processing, the answer could be obvious, i.e. base 10 is preferable to binary. However, it is legitimate to ask whether, from the theoretical point of view, if an optimal base exists [13]. In general, in a positional numbering system, the smaller the base, the greater the number of digits needed to express the numbers. Assuming we are working in base R, the number of digits needed to express X is governed by the equation $X = R^N$ where R is the base and N is the number of digits, rounded to the next highest integer value. If we assume that the complexity of the system (C) is proportional to the ability to represent d digits:

$$C = k \cdot R \cdot N = k \cdot R \cdot \frac{\log X}{\log R} \qquad (1)$$

where k is some constant. The minimum value for C is obtained when R is worth the Euler number and ≈ 2.718. We also arrive at the same result when considering the information registration density:

$$Y(a) = \frac{\ln y(a)}{a} = \frac{\ln a}{a} \qquad (2)$$

From these two different perspectives, the base-three numbering system seems to be the best. However, if we assume that the cost of the circuit and its complexity is independent of the representation base, then the total cost of the system C is proportional to the number of digits (N). In this case:

$$C = k \cdot N = k \cdot \frac{\log X}{\log R} \tag{3}$$

which is a cost gradually decreasing as the base R increases. This means that digital systems using numbering systems with a base greater than binary one is more convenient, even if the benefits tend to decrease as R becomes larger. Furthermore, these considerations ignore any additional benefits of non-binary encoding or native using MVL circuits.

The gap analysis also shows a clear departure from the circuits we propose since modern computers' level of maturity and industrialization is extraordinary. However, our study has allowed us to consider that this gap can be bridged through simplification. Indeed, one can build a machine that performs its task effectively and efficiently from the point of view of consumption if it uses a base greater than two [14], [15].

Moreover, the analysis shows an interesting perspective from the point of view of the research scenario that matches the current technological innovation of new materials and components [16]. In fact, the proposed method does not specify the physical quantity used to encode the values nor the number of symbols used by the encoding alphabet. As a result, to realize this kind of circuit, it is possible to overcome the traditional model that the signals must be electronic and based on the value of a voltage as representative of the information. Furthermore, since the proposed solution is independent of the adopted basis, this could offer scalability and scope for continuous technological improvement [17].

This work aims to demonstrate how it is possible to build efficient computers that work with numbering systems greater than binary and that use multi-value logic. Thus, it is necessary to realize all the circuits that are performed in a CPU. The main objective is divided into two sub-objectives: first, to create a universal multi-value logic gate (in analogy to the NAND and NOR logical gates of Boolean algebra) and second, to create a method that allows to construct any function inside the MVL's domain.

In order to make this discussion less abstract, we have focused the experimentation on the realization both of the universal circuit and of a generic two-valued function in base four using the canonical approach. Finally, we will show how a schematic of a full-adder MVL circuit can be realized from half-adder circuits, highlighting the efficiency in connections and execution time with respect to the binary case.

II. Materials and Method

We first discuss the possibility of constructing a universal function that is minimal, i.e., it can be used as an atomic element to build more complex functions. This method has the shortcoming due to the exponential number of components used concerning operands. As in the binary case, the multiplexer is a circuit capable of realizing any logical function having, in the simplest case, many selection inputs equal to the number of variables of the function to be implemented. If two-way muxes were used, a binary logic function of N variables would require $2^N - 1$ multiplexers. Then, from the theoretical point of view, the availability of a mux circuit operating in MVL logic allows to implement any function in the same domain. So, we define the function:

$$f_{mux}(x, k_0, k_1, \dots, k_{R-1}) = k_i \text{ when } x = i \\ \text{with } x, i, k_0, \dots, k_{R-1} \in [0,1,\dots,R-1] \tag{5}$$

It is easy to show that all R^R unary functions on T can be written by the function $f_{mux}(\cdot)$.

TABLE I
Unary Functions on a MVL Domain

x	f_0	f_1	...	f_{R-1}	...	f_{R^R-1}
0	0	1	...	0	...	R-1
1	0	0	...	0	...	R-1
...	...	...	...		...	...
R-1	0	0	....	R-1	...	R-1

In fact, we can write such functions $f_i(x)$ as follows:

$$f_i(x) = f_{mux}(x, k_0, k_1, \dots, k_{R-1}) \\ \text{with } i \in [0, \dots, R^R - 1] \tag{6}$$

where the coefficients k_i are the values of the i-th column in Table I.

Now consider the case of the function $f_{mux}(\cdot)$ expressing a generic function of N variables: $g_N(x_0, \dots, x_{N-2}, x_{N-1})$. The behavior of the function $g_N(\cdot)$ is described through the MVL truth table (Table II). In the Table, the combinations assumed by the input variables x_i are obtained through the R-base representation of the numbers zero to $R^N - 1$. In Table II, we can see that the variable x_{N-1} partitions the function $g_N(\cdot)$ into R independent functions, each with $\mathrm{N} - 1$ variables.

TABLE II
MVL Truth Table of Function $g(\cdot)$

x_{N-1}	x_{N-2}	...	x_0	$g_N(\cdot)$
0	0	...	0	k_0
0	0	...	1	k_1
...	...	...	...	...
R-1	R-1	R-1	R-1	k_{R^N-1}

Thus, each sub function is responsible for the contribution of $g_N(\cdot)$ when the variable x_{N-1} takes on the values $[0,1,\dots,R-1]$:

$$g_N(x_0, \dots, \mathrm{x}_{N-2}, \mathrm{x}_{N-1}) = \\ \bigcup_{\mathrm{i}_{N-1}=0}^{R-1} g_N(x_0, \dots, x_{N-2}, i_{N-1}) \tag{6}$$

Since the sub functions $g_N(\cdot, i_{N-1})$ perfectly divide the set of values of the function $g_N(\cdot)$ into R disjoint subsets of values, we can use the function $f_{mux}(x_{N-1}, k_0, k_1, \dots, k_{R-1})$ where the constants k_i are the result of the sub functions $g_N(x_0, \dots, x_{N-2}, \cdot)$:

$$g_N(x_0, \dots, x_{N-1}) = \\ = f_{mux}(x_{N-1}, g_N(x_0, \dots, x_{N-2}, 0), \dots, \\ g_N(x_0, \dots, x_{N-2}, R-1)) \tag{7}$$

Proceeding recursively, the sub functions $g_N(x_0, \dots, x_{N-2}, \cdot)$ are themselves decomposable by the same method into R subsets of disjoint values:

$$g_N(x_0,\dots,x_{N-2},i_{N-1}) = \\ = f_{mux}(x_{N-2}, g_N(x_0,\dots,x_{N-3},0,i_{N-1}),\dots, \\ g_N(x_0,\dots,x_{N-3},R-1,i_{N-1})) \\ \forall\, i_{N-1} \in [0,1,\dots,R-1] \tag{8}$$

The procedure ends when the function g(·) is completely decomposed into $\frac{1-R^N}{1-R}$ unary functions all of which are immediately realizable with the function $f_{mux}(x_0,\cdot)$:

$$g_N(x_0,i_1,\dots,i_{N-2},i_{N-1}) = \\ = f_{mux}(x_0, g_N(0,i_1,\dots,i_{N-1}),\dots, \\ g_N(R-1,i_1,\dots,i_{N-1})) \\ \forall\, i_1,\dots,i_{N-1} \in [0,1,\dots,R-1] \tag{9}$$

The procedure thus shows how it is possible to decompose a generic function $g_N(x_0,\dots,x_{N-1})$ into a hierarchy of subfunctions $f_{mux}(\cdot)$. It is easy to show how the validity of the procedure in the case of *N*-variable functions is also applicable to the case of $N+1$ variable functions: $g_{N+1}(x_0,\dots,x_N)$. In fact, the addition of a variable introduces a level in the hierarchy i.e. of $R-1$ cases that will be treated by the function $f_{mux}(\cdot)$ deputed to discriminate the last variable x_N.

The approach adopted, from the mathematical point of view, finds perfect correspondence in MVL circuits. In Fig.1 it is shown how $\frac{R^N-1}{R-1}$ mux circuits can realize any MVL circuit with N input variables.

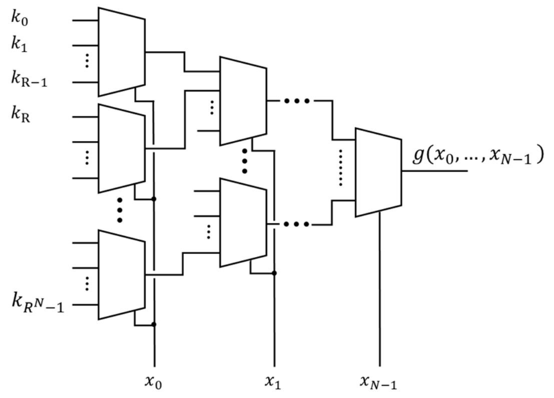


Fig. 1 Circuit that realizes the function $g_N(\cdot)$ through the use of $f_{mux}(\cdot)$

A different mathematical approach was preferred instead of using the notions available on Post algebras of order greater than or equal to two. Considered a discrete and finite set of values $T=\{0,1,\dots,R-1\}$ it is possible to describe any function $f(x_0,\dots,x_{N-1})\colon T\times\dots\times T\to T$, with $x_i\in T\ \forall i$, through the linear combination of the selection functions $S_w(x_j)$ with $w\in\{0,\dots,R-1\}$ and $j\in\{0,\dots,N-1\}$, of the values assumed by variables x_j multiplied by the corresponding coefficients that determine the output value k_i:

$$f(x_0,\dots,x_{N-1}) = \sum_{i=0}^{R^N-1} k_i \cdot \prod_{j=0}^{N-1} S_{c_j(i)}(x_j) \tag{4}$$

where

- k_i is the value taken in the row corresponding to the number *i*, with $k_i\in[0,\dots,R-1]$;
- $c_j(i)$ is the *j*-th digit of the number i, represented in the base R with $j\in[0,\dots,N-1]$;
- $S_{c_j(i)}(x_j)$ is the value selector $c_j(i)$ applied to the operand x_i with $\in[0,\dots,N-1]$.

At this point, to define any function, we just need a number of unary operators as well as the values in the MVL domain and two functions: addition and multiplication. Moreover, it holds that one and only one term in a sum will be different from zero as well as k_i values will multiply productivity that is zero or one (the row selection group). In order to clear the steps that we followed in the proposed research, we show the MVL circuit designing and development life cycle (Fig. 2).

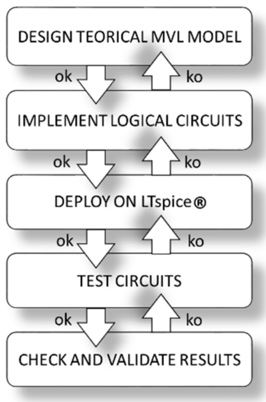

DESIGN TEORICAL MVL MODEL
ok
ko
IMPLEMENT LOGICAL CIRCUITS
ok
ko
DEPLOY ON LTspice®
ok
ko
TEST CIRCUITS
ok
ko
CHECK AND VALIDATE RESULTS

Fig. 2 MVL circuit designing and development life cycle

Without loss of generality, we will consider the field of electronic digital signals as a system for representing and processing information, as is normally the case for a traditional binary computing system. We will also adopt base 4 for our analysis and a maximum voltage supply of 5 volts (Vcc). Nothing prohibits us from proceeding with more levels (the base R) in the future and with lower supply voltage values to minimize power consumption and thermal dissipation. However, this work intends to demonstrate the functional feasibility of the solution.

For issues primarily related to noise isolation, logic levels can be defined by equally dividing the available representation range (5 V):

TABLE III
LEVELS IN BASE 4

Logic level	interval		Videal [V]
	Vmin [V] ≥	Vmax [V] <	
0	0	1.25	0.625
1	1.25	2.50	1.875
2	2.50	3.75	3.125
3	3.75	5.00	4.375

At this point, we only need to realize an electronic circuit capable of performing the unary functions of selection, addition, and multiplication. For our research, we have used the product LTspice® XVII, a simulation software copyrighted by Analog Device Corporation distributed on the internet at the manufacturer's site.

A. The MVL Universal Circuit

We will start with the selection functions, which are in base S_0, S_1, S_2 and S_3 . The functions $S_i(x)$ return one when $x=i$ and zero in all other cases (Table IV). For this task, we rely on a circuit that uses a voltage divider (Fig. 3). The resistors are calculated to obtain the potential of about 0.5 V on the junction nodes corresponding to the recognized level to trigger the buffer.

TABLE IV
UNARY SELECTION FUNCTIONS: SELECTORS

x	***S₀***	***S₁***	***S₂***	***S₃***
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1

Instead, the inverter chain and the binary AND logic gates allow making the $S_i(x)$ signals exclusive.

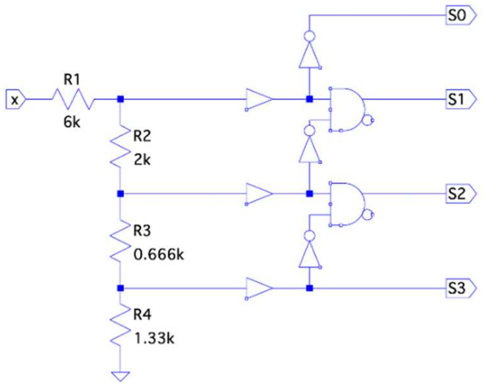


Fig. 3 Selector circuit that recognizes the input value

For our purposes, the functions that we need to implement on the MVL domain are:

- multiplication of a digit k_i by values $S_{c_w(i)}(x_w)$ that can be zero or one;
- sum of terms $k_i \cdot S_{c_{N-1}(i)}(x_{N-1}) \cdot \ldots \cdot S_{c_0(i)}(x_0)$ where only one is greater than zero and the others are zero.

The function $f_{mux}(\cdot)$ can be written using the expression (4):

$$f_{mux}(x, k_0, k_1, k_2, k_3) = \sum_{i=0}^{3} k_i \cdot S_i(x) \tag{10}$$

The operation of multiplication by the constant value k_i is obtained thanks to a voltage-controlled switch that closes and lets the applied voltage pass entirely as output when receiving the value one in the control input. Otherwise, no voltage contribution is provided in output since the switch stay opened.

The sum is obtained due to the principle that only one switch at a time is active, so it is sufficient to connect the switch outputs together. In the simulations, we used an internal component called voltage-controlled switch to which the MYSW model was assigned with the following parameters:

$$Ron = 1 \;\; Roff = 1Meg \;\; Vt = 0.5 \;\; Vh = -0.4 \tag{11}$$

The mux circuit is shown in Fig. 4.

B. Two-operand functions

MVL functions are mathematical applications that operate on a domain of discrete numbers $T = [0, 1, 2, 3]$ and return values in the same domain T:

$$f(x_0, \ldots, x_{N-1}): T \times \ldots \times T \rightarrow T \\ \text{with } x_0 \in T, \ldots, x_{N-1} \in T \tag{12}$$

From a general perspective, the circumstance that a function returns a result as an operation acting over its (discrete) inputs makes it similar to an array. The result is predetermined as a function of the inputs in the array within the function instead of a mathematical calculation outcome. Consequently, we can implement a two-operand function through a $R \times R$ matrix that stores the output values based on the two input variables. The latter, being discrete, represent the row and column indexes of the array, respectively, while the content of the indexed element is the value the function will have to return.

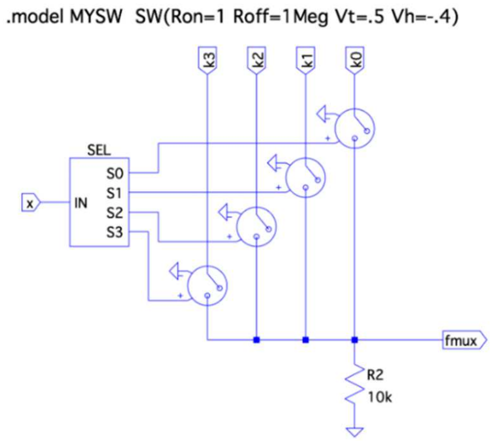


Fig. 4 Four-way MVL mux circuit and one select input

Following this approach, we devised a circuit that uses the selection blocks of Fig. 3 to identify the value of the input variables and an AND matrix to return the correct value on the output (Fig. 5).

Clearly, only one AND at a time will be active and it will not be necessary to encode the zero value on the output since it is the default value. We have adopted a component approach because it allows to reuse the circuits devised. In order to make it easier to understand the circuit that realizes a two-valued MVL function, we will consider a concrete example: the maximum of two elements.

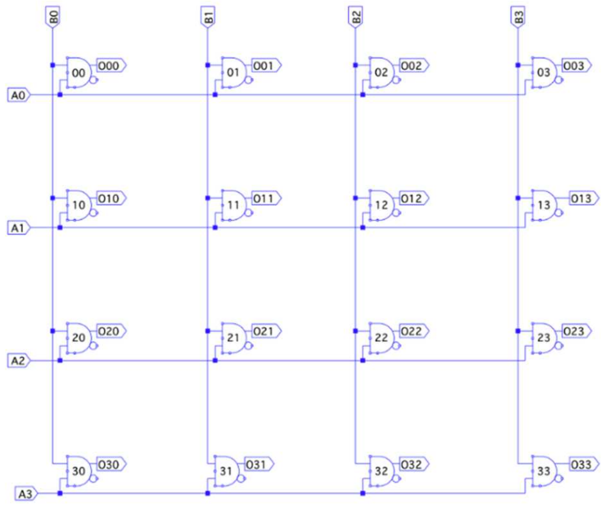


Fig. 5 AND matrix for row selection

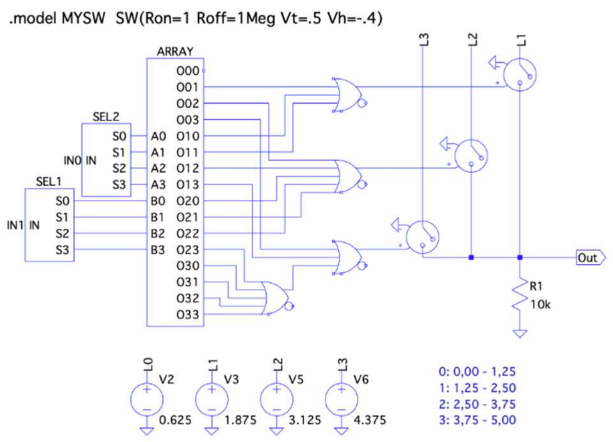
Table V shows the values of the function and the positions at which the related values are enabled to output. If you want to make the final circuit simpler, it is better to group the common values to be provided in output through one or more OR ports, especially if the number of cases is greater than the number of inputs of the used port (Fig. 6).

TABLE V
MVL TRUTH TABLE OF THE MAX FUNCTION

N	A	B	Max(A,B)	0	1	2	3
0	0	0	0	0			
1	0	1	1		1		
2	0	2	2			2	
3	0	3	3				3
4	1	0	1		4		
5	1	1	1		5		
6	1	2	2			6	
7	1	3	3				7
8	2	0	2			8	
9	2	1	2			9	
10	2	2	2			10	
11	2	3	3				11
12	3	0	3				12
13	3	1	3				13
14	3	2	3				14
15	3	3	3				15

C. *Functions with more than two operands*

Functions with more than two operands can be implemented using as many selection blocks as well as the input variables, an AND array that defines the combinations that enable the value to be placed on the output, and R-1 voltage-driven switches (Fig. 7).



```
.model MYSW  SW(Ron=1 Roff=1Meg Vt=.5 Vh=-.4)
```

Fig. 6 Example of max function with two operands

If we have a function with more than three input variables we can use multidimensional arrays. For example, in a combinatorial circuit with 4 input variables, the fourth variable will select one of the 4 possible cubes. So with $R = 4$, four cubes of size $4 \times 4 \times 4$ will be needed.

D. *An example: summing circuit*

As far it has been seen, any function can be wired within the AND matrix in order to obtain the desired result. In this section, we will look at the problem of summing MVL numbers and start with the half-adder circuit (Fig. 7). Before constructing such circuit, it is necessary to define the output values through a table that highlights the circuit behavior, as done in Table V.

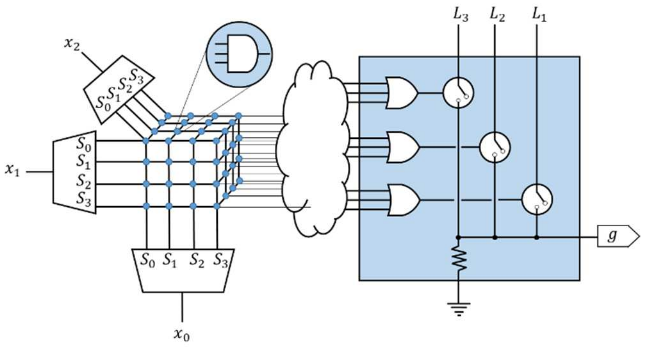
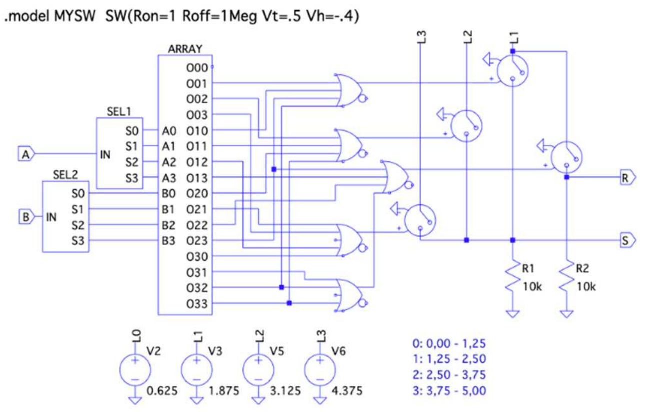


Fig. 7 Example of an MVL function with three operands: $g(x_0, x_1, x_2)$

Once the half-adder has been implemented (Table VI), it will be possible to build a full-adder by composition of two half-adders as it happens in the binary circuit. In the same way, we can build multi-digit summers using the half-adder and propagating the carryover between elements.

TABLE VI
MVL TRUE TABLE OF THE HALF-ADDER

Digit	8	7	6	5	4	3	2	1	0
R	1	1	0	0	0	0	1	1	
A		3	1	0	2	0	0	2	3
B		3	3	2	1	2	1	3	1
Sum	1	3	0	2	3	2	2	2	0



```
.model MYSW  SW(Ron=1 Roff=1Meg Vt=.5 Vh=-.4)
```

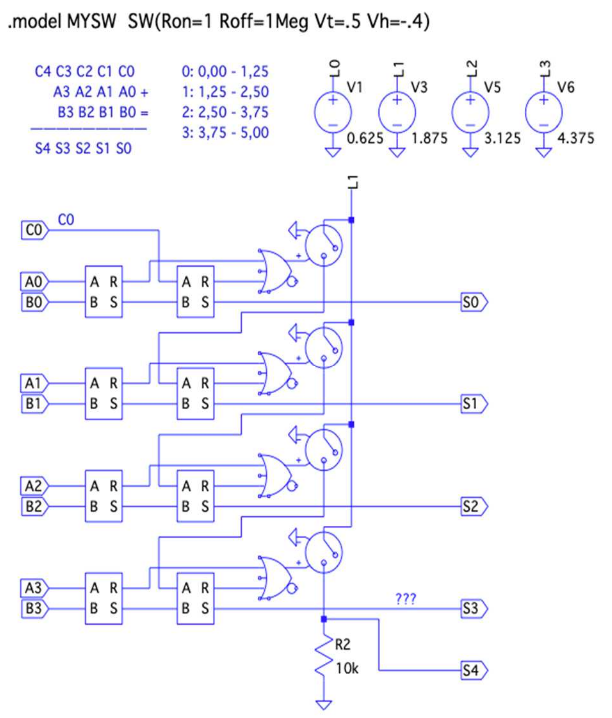
Fig. 8 Half-adder MVL circuit

The circuit on Fig. 9 allows to sum two numbers in base 4 consisting of 8 digits. In the same figure, you can see the simulation that demonstrates the correctness of the result.

III. RESULTS AND DISCUSSION

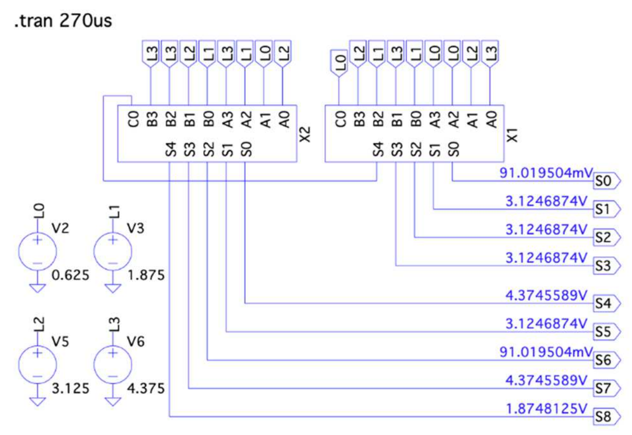
Although the aim of the work is to evaluate the feasibility of the MVL solution from a functional point of view, we also studied the behavior of the proposed circuits by verifying the limits of the adopted components. In the following we define the details of the configurations used. The following parameters have been defined for the AND and OR logic gates:

$$\begin{gathered}\mathrm{Td} = 10\mathrm{n}\ \ \mathrm{Ref} = 0.5\ \ \mathrm{Trise} = 5\mathrm{n} \\ \mathrm{Tfall} = 5\mathrm{n}\ \ \mathrm{Vhigh} = 5\end{gathered} \tag{13}$$



```
.model MYSW  SW(Ron=1 Roff=1Meg Vt=.5 Vh=-.4)
```

Fig. 9 Four-digit number summing circuit



```
.tran 270us
```

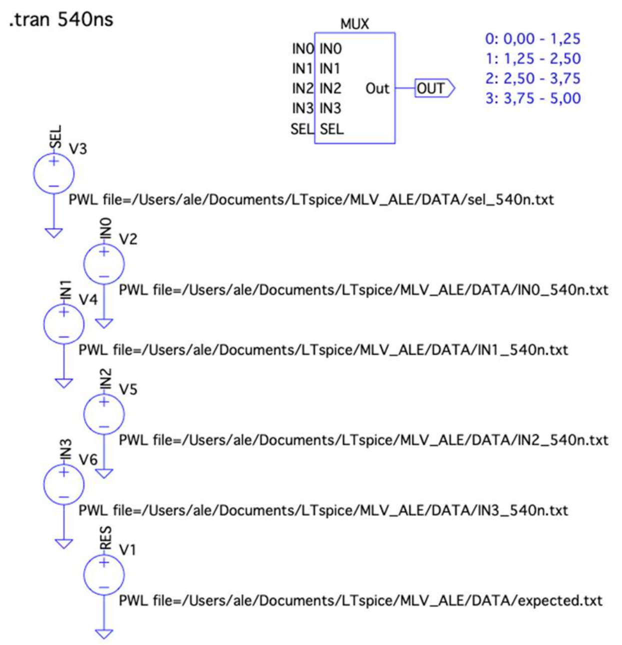
Fig. 10 Eight-digit summing machine using the 4-digit summing machine

While for the buffer and inverter ports the assigned parameters are:

$$\begin{gathered} \mathrm{Td} = 5\mathrm{n} \ \ \mathrm{Ref} = 0.5 \ \ \mathrm{Trise} = 5\mathrm{n} \\ \mathrm{Tfall} = 5\mathrm{n} \ \ \mathrm{Vhigh} = 5 \end{gathered} \tag{14}$$

Because the universal circuit in multi-value logic is the multiplexer with 4 data paths and a selection input, it was chosen for functional and performance testing. In particular, four ideal voltage generators ($IN0, IN1, IN2, IN3$) were used in which time-varying voltage trends were modelled through the PWL directive. This allowed to define 5 different trends of input signals with a frequency equal to 16.67 MHz shown in Table VII. The Table also shows the expected value for the output (OUT).

The results show that the circuit needs an initial transient of about 30 ns to settle and return the correct value at the output. Fig. 7 shows in blue the output voltage and in red the expected value based on the inputs.



```
.tran 540ns
PWL file=/Users/ale/Documents/LTspice/MLV_ALE/DATA/sel_540n.txt
PWL file=/Users/ale/Documents/LTspice/MLV_ALE/DATA/IN0_540n.txt
PWL file=/Users/ale/Documents/LTspice/MLV_ALE/DATA/IN1_540n.txt
PWL file=/Users/ale/Documents/LTspice/MLV_ALE/DATA/IN2_540n.txt
PWL file=/Users/ale/Documents/LTspice/MLV_ALE/DATA/IN3_540n.txt
PWL file=/Users/ale/Documents/LTspice/MLV_ALE/DATA/expected.txt
```

Fig. 11 Mux circuit test with 4 ways and one selection input

TABLE VII
INPUT AND OUTPUT SIGNALS FROM THE MVL MULTIPLEXER

TIME [ns]	IN0 [V]	IN1 [V]	IN2 [V]	IN3 [V]	SEL [V]	EXPECTED OUT [V]
0	0.625	3.125	1.875	4.375	1.875	3.125
60	0.625	3.125	1.875	4.375	1.875	3.125
61	4.375	1.875	3.125	0.625	0.625	4.375
120	4.375	1.875	3.125	0.625	0.625	4.375
121	1.875	0.625	4.375	4.375	3.125	4.375
180	1.875	0.625	4.375	4.375	3.125	4.375
181	4.375	0.625	1.875	3.125	4.375	3.125
240	4.375	0.625	1.875	3.125	4.375	3.125
241	1.875	4.375	4.375	0.625	4.375	0.625
300	1.875	4.375	4.375	0.625	4.375	0.625
301	3.125	0.625	1.875	4.375	3.125	1.875
360	3.125	0.625	1.875	4.375	3.125	1.875
361	0.625	3.125	4.375	1.875	1.875	3.125
420	0.625	3.125	4.375	1.875	1.875	3.125
421	4.375	4.375	0.625	3.125	3.125	0.625
480	4.375	4.375	0.625	3.125	3.125	0.625
481	0.625	1.875	4.375	4.375	4.375	4.375
540	0.625	1.875	4.375	4.375	4.375	4.375

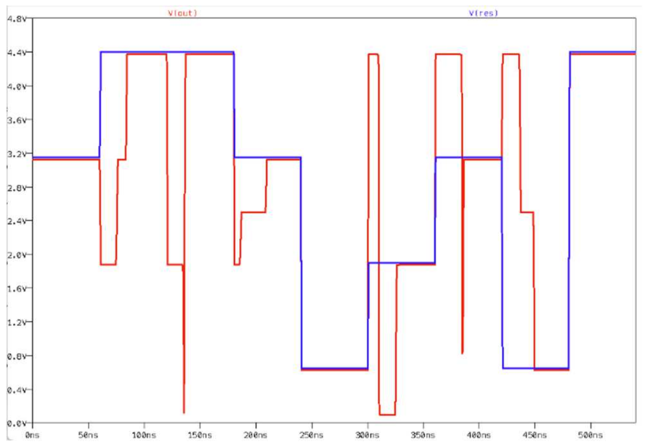


Fig. 12 Difference between mux output signal and expected signal

IV. Conclusion

The idea behind this article is that it is possible to make MVL circuits capable of performing the same functions available to digital circuits. This article shows the operation of circuits that adopt the same representation system as traditional binary digital circuits. However, other techniques could be used to store MVL values, such as the resistance value of a memristor [18], [19]. In fact, since a MVL function is a mathematical application that leads from a set of discrete domains to the same discrete domain, we can assimilate a MVL function to a memory. The input variables act as indices of the memory location in which we find the expected output.

A multi-value memory can be easily realized using arrays of memristors. These devices are widely used in interconnected grids to quickly perform matrix calculations, exploiting their ability to set their resistance within certain values. This is an example of using the resistance of a passive element to store a discrete value. However, with new materials and nanotechnology, nothing prohibits us from associating a multi-value quantity with a physical entity different from the electrical potential in the future [20].

The main shortcoming is the settling time, which would limit the operating frequency of the summing circuit. If this research is to be followed up with an industrial approach, the circuits will have to be engineered to reduce the settling time and increase the operating frequency. The other aspect that is considered important in this research is the scalability of the solution. All the theory described makes no assumptions about the number of levels used. So, the approach remains valid regardless of the base adopted. For reasons of backward compatibility and in order to make these new circuits immediately integrable with current systems, it is desirable to use bases that are powers of two. The upcoming research is subject to build new circuit apparatus that can be used within a complete computing architecture entirely in MVL logic.

Acknowledgment

We are grateful to Prof. Roberto Basili for his careful review and precious suggestions to the manuscript.

References

[1] F. Silvestri, S. Acciarito, G. C. Cardarilli, G. M. Khanal, L. D. Nunzio, R. Fazzolari, M. Re, Fpga implementation of a low-power qrs extractor, volume 512, 2019. doi:10.1007/978-3-319-93082-4_2.

[2] G. C. Cardarilli, L. D. Nunzio, R. Fazzolari, M. Re, Fine-grain reconfigurable functional unit for embedded processors, 2011. doi:10.1109/ACSSC.2011. 6190048.

[3] D. Giardino, G. C. Cardarilli, L. D. Nunzio, R. Fazzolari, A. Nannarelli, M. Re, S. Spano, M-psk demodulator with joint carrier and timing recovery, IEEE Transactions on Circuits and Systems II: Express Briefs 68 (2021). doi:10.1109/TCSII.2020.3041342.

[4] G. C. Cardarilli, L. D. Nunzio, R. Fazzolari, D. Giardino, M. Matta, M. Patetta, M. Re, S. Spanò, Approximated computing for low power neural networks, Telkomnika (Telecommunication Computing Electronics and Control) 17 (2019). doi:10.12928/TELKOMNIKA.v17i3.12409.

[5] Z. T. Sandhie, J. A. Patel, F. U. Ahmed, M. H.Chowdhury, Investigation of multiple-valued logic technologies for beyond-binary era, ACM Comput. Surv. 54 (2021). doi:10.1145/3431230.

[6] B. S. Raghavan, V. S. K. & Bhaaskaran, Design of novel Multiple Valued Logic (MVL) circuits, International Conference on Nextgen Electronic Technologies: Silicon to Software, (2017) 371-378. doi:10.1109/ICNETS2.2017.8067963.

[7] B. Cambou, P. Flikkema, J. Palmer, D. Telesca, C. Philabaum, Can ternary computing improve information assurance? Cryptography 2 (2018) 6. doi:10.3390/cryptography2010006.

[8] D. Etiemble, Evolution of technologies and multi-valued circuits, ArXiv abs/1907.01451 (2019).

[9] S. N. Shah, Addressing the interpretability problem for deep learning using many valued quantum logic (2020). arXiv: 2007.01819.

[10] R. Wang, J.-Q. Yang, J.-Y. Mao, Z.-P. Wang, S. Wu, M. Zhou, T. Chen, Y. Zhou, S.-T. Han, Recent advances of volatile memristors: Devices, mechanisms, and applications, Advanced Intelligent Systems 2 (2020). doi:10.1002/aisy.202000055.

[11] M. Huang, X. Wang, G. Zhao, P. Coquet, B. Tay, Design and implementation of ternary logic integrated circuits by using novel two-dimensional materials, Applied Sciences 9 (2019). doi:10.3390/app9204212.

[12] A. Maksim C. Jae-Woong, K.Jiwan, K. Hyeongjun, J. Sooyoung, K. Kwan-Ho, P. Jin-Hong, Negative differential transconductance device with a stepped gate dielectric for multi-valued logic circuits, Nanoscale Horizons, 10 (2020) 1378-1385. doi: 10.1039/D0NH00163E.

[13] A. Esin, Analysis and design principles of modern control systems based on multi-valued logic models, Upravlenie Bol'shimi Sistemami 88 (2020) 69–98, doi:10.25728/ubs.2020.88.4.

[14] Z. Sun, G. Pedretti, E. Ambrosi, A. Bricalli, W. Wang and D. Ielmini, Solving matrix equations in one step with cross-point resistive arrays, Proceeding of the National Academy of Sciences of the United States of America 116 (2019). doi:10.1073/pnas.1815682116.

[15] M. Jhamb, R. Mohan, Ultra low power design of multi-valued logic circuit for binary interfaces, Journal of King Saud University - Computer and Information Sciences (2021). doi:10.1016/j.jksuci.2021.01.010.

[16] L. Lee, J. Hwang, J. W. Jung, J. Kim, H. I. Lee, S. Heo, M. Yoon, S. Choi, N. V. Long, J. Park, J. W. Jeong, J. Kim, K. R. Kim, D. H. Kim, S. Im, B. H. Lee, K. Cho, M. M. Sung, Zno composite nanolayer with mobility edge quantization for multi-value logic transistors, Nature Communications 10 (2019). doi:10.1038/ s41467-019-09998-x.

[17] A. Simonetta, M. C. Paoletti, Designing digital circuits in multi-valued logic, International Journal on Advanced Science, Engineering and Information Technology 8 (2018) 1166-1172. doi:10.18517/ijaseit.8.4. 5966.

[18] O. Krestinskaya, B. Choubey, A. P. James, Memristive gan in analog, Scientific Reports 10 (2020). doi:10.1038/s41598-020-62676-7.

[19] D. Bhattacharjee, W. Kim, A. Chattopadhyay, R. Waser, V. Rana, Multi-valued and fuzzy logic realization using TaOx memristive devices, Scientific Reports 8 (2018) 8. doi:10.1038/s41598-017-18329-3.

[20] S. B. Jo, J. Kang, J. Cho, Multi-valued logic gates: Recent advances on multi-valued logic gates: A materials perspective, Advanced Science 8 (2021). doi:10.1002/advs.202170040.