Multi-Valued Logic Digital Circuits for Realizing a Complete Computer Architecture

Alessandro Simonetta[1], Maria Cristina Paoletti[1] and Alessio Venticinque[2]

[1]Department of Enterprise Engineering, University of Rome Tor Vergata, Rome, Italy
[2]Department of Electrical and Information Engineering, University of Naples Federico II, Napoli, Italy

Abstract

The objective of this paper is to lay the foundation for the construction of a computer architecture using entirely multivalue logic (MVL). To achieve this ambitious result, it is necessary to know how to design all the digital components that normally form the basis of a computer's operation. These are combinational circuits dedicated to mathematical computation and sequential circuits in charge of storing information. In the paper, a general methodology is proposed that can be used for the construction of digital circuits capable of working independently of the basis of representation of multivalue logic and the physical quantity used for encoding logical states.

Keywords

MVL, Multi-valued logic, CPU, Computer architecture, digital circuit, adder look ahead

1. Introduction

The subject of multivalue logic as a natural evolution of representation in the binary system is a much debated issue [1], [2]. Although various solutions, including technological ones, have been proposed, but at present there is no calculator capable of operating in multivalued logic [3], [4], [5]. However, with the arrival of memresistors [6] and a multiplicity of multi-state semiconductor components, the scenario is evolving and the theory developed in [7] could find easy application. The theory we are going to describe is based on the extension of Boolean algebra and the binary numbering system toward an n-valued (discrete) numbering system [8], [9]. In this paper we will start from the theoretical definition and gradually go on to realize all the digital components that characterize a traditional computer architecture.

2. Method

The proposed method use the teory in [10]. Assuming that we have a domain of discrete values $\mathbb{T} = \{0, 1, .., n-1\}$, we can easily show that any function f:

$$f : \mathbb{T} \times ... \times \mathbb{T} \to \mathbb{T} \tag{1}$$

can be expressed as a linear combination of the input variables:

$$f(x_o, x_1, ..., x_{p-1}) = \sum_{i=0}^{n^p-1} k_i \cdot \prod_{j=0}^{p-1} S_{c_j}(i)(x_j) \tag{2}$$

where:

- k_i is the value taken at the i-th position in the combination, with $k_i \in \mathbb{T}$;
- $c_{j(i)}$ is the j-th digit of the number i, represented in base n with $j \in \{0, ..., p-1\}$;
- $S_{c_j(i)}$ is the value selector $c_j(i)$ applied to operand x_j with $j \in \{0, ..., p-1\}$ e $i \in \{0, ..., n^p - 1\}$;

Thus, to express any MVL function f, p+2 basis function are sufficient: p selection, addition and multiplication functions.

2.1. The selection functions

The selection functions S_i with $i \in \{0, ..., n-1\}$, that we can call *Selectors* too, are unary functions. We can define as many selectors as there are symbols in the domain $\mathbb{T}$.

$$S_i : \mathbb{T} \to \mathbb{B} \tag{3}$$

A selector is able to check whether the value of the operand matches a value in the set $\mathbb{T}$:

$$S_i(t) = \begin{cases} 1 & se \;\; t = i \\ 0 & se \;\; t \neq i \end{cases} \tag{4}$$

with $i, t \in \{0, ..., n-1\}$. Using the selection functions, any combination of the input variables can be identified. For example, to check if all variables $x_0, ..., x_{p-1}$ have

ICYRIME 2022: International Conference of Yearly Reports on Informatics, Mathematics, and Engineering. Catania, August 26-29, 2022
alessandro.simonetta@gmail.com (A. Simonetta);
mariacristina.paoletti@gmail.com (M. C. Paoletti)
0000-0003-2002-9815 (A. Simonetta); 0000-0001-6850-1184 (M. C. Paoletti); 0000-0003-3286-3137 (A. Venticinque)

the value $n-1$ except x_0 which must be equal to 1, we can use the following function:

$$\phi = S_1(x_0) \cdot S_{p-1}(x_1) \cdot ... \cdot S_{p-1}(x_{p-1}) \quad (5)$$

which returns the value 1 exclusively in the case where the input variables meet the requirement:

$$(x_0, ..., x_{p-1}) = (1, n-1, ..., n-1) \quad (6)$$

2.2. Sum

Let us consider the summation of equation 2, we observe that only one term in the summation will be valued at k_w corresponding to the coding in base n of w. Indeed, all the terms of the summation are null except the one corresponding to the combination identified by the selection functions:

$$f(x_o, x_1, ..., x_{p-1}) = k_w \quad (7)$$

2.3. Multiplication

With reference to the generic term of the summation of the equation 2, it can be expressed by a function ϕ:

$$\phi : \mathbb{T} \times \mathbb{B} \times ... \times \mathbb{B} \to \mathbb{T} \quad (8)$$

that is equivalent to θ:

$$\theta : \mathbb{T} \times \mathbb{B} \to \mathbb{B} \quad (9)$$

thus:

$$t \cdot b_0 \cdot b_1 \cdot ... \cdot b_{p-1} = t \cdot (b_0 \cdot b_1 \cdot ... \cdot b_{p-1}) = t \cdot b \quad (10)$$

Indeed, it is sufficient to make a function with two operands: one MVL and the other binary. This function returns as output the value of the multivalue input (if the binary value is unity) or zero in the other case:

$$\theta(b, t) = \begin{cases} 0 \ \ se \ \ b = 0 \\ t \ \ se \ \ b = 1 \end{cases} \quad (11)$$

with $t \in \mathbb{T}$ and $b \in \mathbb{B}$.
Recalling that the logical conjunction operation returns a result that is equivalent to multiplication between bits (the result is one if and only if all the operands are 1), we can conclude that it is possible to use the AND operator to compute the value b in the binary domain. At this point it is possible to use that bit to calculate the MVL function $\theta(b, t)$. Without loss of generality and with the purpose of making the paper easier to read, we will use the symbol normally used for multiplication to denote the function θ conscious of the fact that it is multiplication between values pertaining to different domains ($\mathbb{B} \subseteq \mathbb{T}$).

3. MVL digital circuits

In this section, we will see how it is possible to construct all the circuits that form the basis of a traditional computer architecture [11], [12]. Thus, we will start with combinational circuits, those in which the output is a function of the inputs, and then we will consider the smallest memory element, the D Latch. To accomplish this task we will use the same architectural choice made in [10], and supported by studies in [13], thus a logic based on 4 values that coincides with the base-4 numbering system. The four logic values will be represented through the voltage levels as reported in table 1.

Table 1
MVL levels

Logic level	Interval		V_{ideal}
	$V_{min} \geq$	$V_{max} <$	
L_0	0	1.25	0.625
L_1	1.25	2.50	1.875
L_2	2.50	3.75	3.125
L_3	2.50	5.00	4.375

We also studied the behavior of the proposed circuits by verifying the limits of the adopted components [14], [15], [16]. In particular, the details for the configuration of the components are the following:

- *AND* and *OR* logic gates:
  - Td = 10n
  - Ref = 0.5
  - Trise = 5n
  - Tfall = 5n
  - Vhigh = 5
- *inverted gates*:
  - Td = 5n
  - Ref = 0.5
  - Trise = 5n
  - Tfall = 5n
  - Vhigh = 5

3.1. LTSpice

The electronic circuits that we will describe, were designed and validated using the product LTspice® XVII by Analog Device Corporation [17], distributed on the developer's website. This simulator has a graphical interface to build the schematics and allows the user to test the circuits. The tool also allows new features to be defined and imported as elements in the various schemes. Therefore, we chose to reuse the circuit diagrams proposed in [10] regarding the selectors, multiplexer and half-adder to design new ones.

3.2. Combinational Circuits

Table 2
MVL true table: adding single MVL digits

A_i	B_i	C_i	S_i	C_{i+1}	g_i	p_i
0	0	0	0	0	0	0
0	0	1	1	0	0	0
0	1	0	1	0	0	0
0	1	1	2	0	0	0
0	2	0	2	0	0	0
0	2	1	3	0	0	0
0	3	0	3	0	0	0
0	3	1	0	1	0	1
1	0	0	1	0	0	0
1	0	1	2	0	0	0
1	1	0	2	0	0	0
1	1	1	3	0	0	0
1	2	0	3	0	0	0
1	2	1	0	1	0	1
1	3	0	0	1	1	0
1	3	1	1	1	1	0
2	0	0	2	0	0	0
2	0	1	3	0	0	0
2	1	0	3	0	0	0
2	1	1	0	1	0	1
2	2	0	0	1	1	0
2	2	1	1	1	1	0
2	3	0	1	1	1	0
2	3	1	2	1	1	0
3	0	0	3	0	0	0
3	0	1	0	1	0	1
3	1	0	0	1	1	0
3	1	1	1	1	1	0
3	2	0	1	1	1	0
3	2	1	2	1	1	0
3	3	0	2	1	1	0
3	3	1	3	1	1	0

with $A_i, B_i, S_i \in \mathbb{T}$ and $C_i, C_{i+1}, g_i, p_i \in \mathbb{B}$

MVL look-ahead adder In this section we will study the behavior of the circuit MVL look-ahead adder of 4-digit base-4 words. In [7] a half-adder consisting of two inputs and two outputs in base 4 has already been presented, and starting from this circuit the authors design a full-adder in [10]. The most obvious problem with this circuit is the response time, which depends on the propagation delay of the carryover in the various half-adders. The time grows as the number of digits of operands and thus half-adders used to construct the adder increases [18], [19]. Indeed, in a look-ahead adder the carryover is not affected by the delay of circuit elements placed in cascade. Consider two MVL words of length q: $A = \{A_{q-1}, ..., A_0\}$ and $B = \{B_{q-1}, ..., B_0\}$, if the calculated carryovers are $C = \{C_q, ..., C_1\}$ and C_0 is the potential carryover of a previous least significant word, we can calculate the sum word $S = \{S_q, ..., S_0\}$

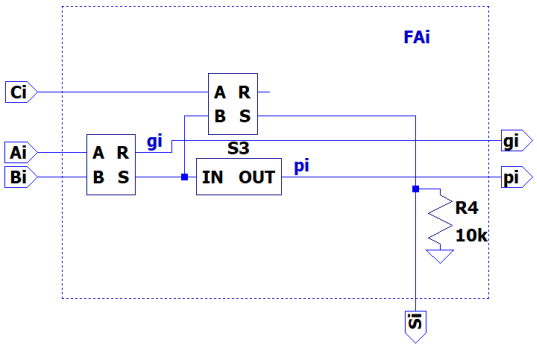

Figure 1: Look-ahead full adder single digit detail

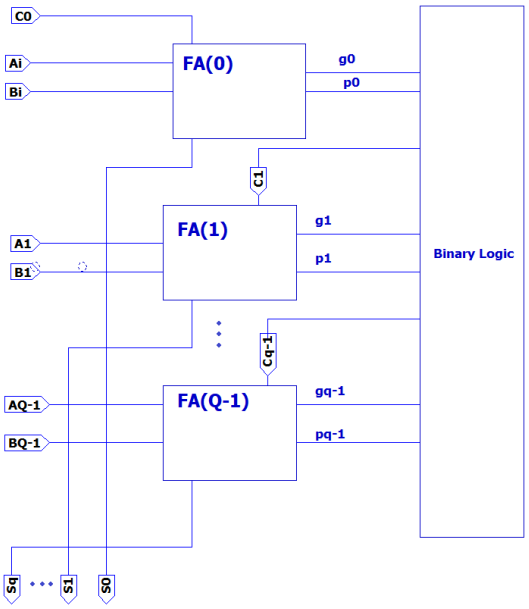

Figure 2: Look-ahead full adder blocks scheme

as indicated in table 3.

Table 3
Adding two MVL words of q length

C_q	C_{q-1}	...	C_0	
	A_{q-1}	...	A_0	+
	B_{q-1}	...	B_0	=
S_q	S_{q-1}	...	S_0	

If by the variables p_i and g_i we denote the generated and propagated carryover, respectively, we can build the truth table (table 2) related to the sum of two generic

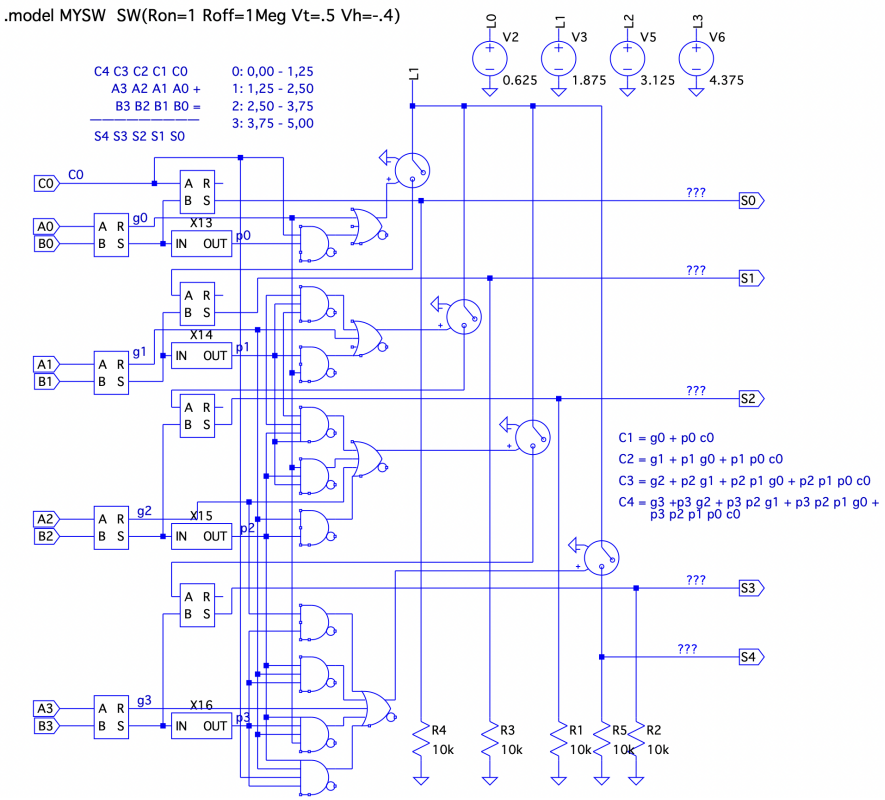

Figure 3: Look-ahead adder with 4 MVL digits

MVL digits. The values of p_i and g_i are given by the relation 12 for a cirucite in which the signals are in base n.

$$g_i = \begin{cases} 0 & if \;\; A_i + B_i \leq n-1 \\ 1 & if \;\; A_i + B_i > n-1 \end{cases} \tag{12}$$

$$p_i = \begin{cases} 0 & if \;\; A_i + B_i = n-1 \\ 1 & if \;\; A_i + B_i \neq n-1 \end{cases} \tag{13}$$

In the same way, in the case of the MVL, we can determine in advance the values of the carryovers C_{i+1} from the variables g_i and p_i . Using the symbols $\sum$ and $\prod$, for logical disjunction (OR) and logical conjunction (AND), respectively, we can write the equation 14.

$$C_{i+1} = g_i + C_0 \prod_{j=0}^{i} p_j + \sum_{k=0}^{i-1} g_k \prod_{j=k+1}^{i} p_j \tag{14}$$

However, the values of the carryovers (C_{i+1}), determined in such a way, are in the binary domain, instead we need the value corresponding to the logical level L_1 of the table 1. Then a transduction element is needed that, through a switch driven by a binary signal, can send the value corresponding to L_1 to the second half-adder. The adder circuit will use two half-adders to calculate $S_i = C_i + A_i + B_i$; the variable g_i will be calculated from the first half-adder while p_i will be calculated using the selection function to understand whether $A_i + B_i = n-1$ (figure 1). This component will be replicated for each couple of input in order to build a look-ahead adder as in figure 2. In figure 3 a look-ahead adder circuit with 4 MVL digits is made. This circuit can be used to perform MLV word sum with lengths multiple of 4: for example, we can add up 8 MVL digits by cascading two circuits (figure 4). The behavior of the circuit is verified through a concrete example: suppose we add the value $A = 310200023$ with $B = 33212131$ and that the carryover C_0 will be 0. The expected result is the number S=130232220 as can be easily seen from the calculation shown in the table 4.

Decoder A decoder is a combinational circuit with r inputs and n^r outputs. It returns as output the decoded form of a coded word given as input.
Starting with the most basic decoder, which decodes a single digit in the four possible values, we will construct

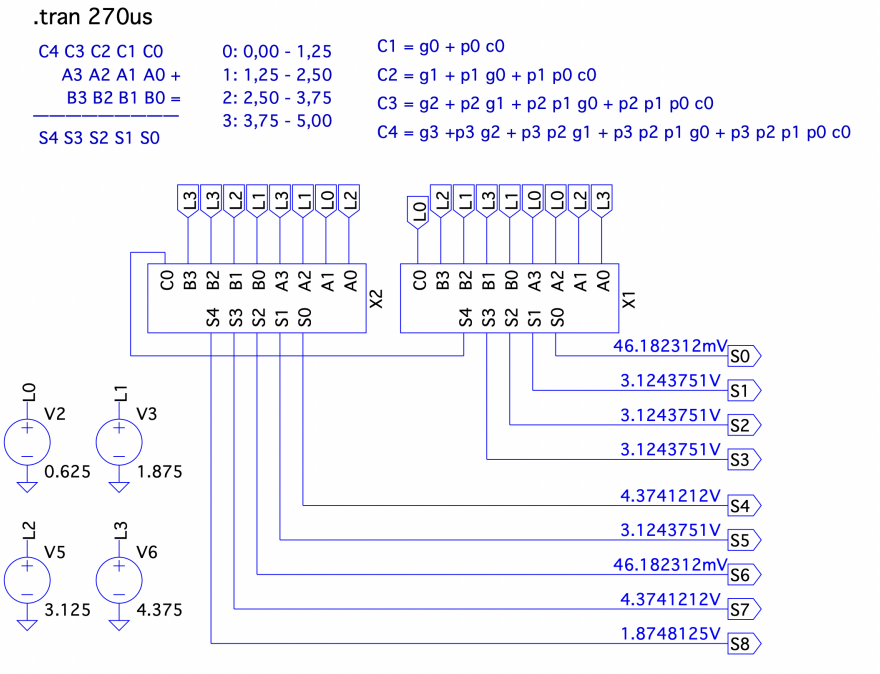

Figure 4: Look-ahead adder with 8 MVL digits

Table 4
An example of two words sum of 8 MVL digits

1	1					1	1		
	3	1	0	2	0	0	2	3	+
	3	3	2	1	2	1	3	1	=
1	3	0	2	3	2	2	2	0	

the decoder that decodes two-digit words. Making a single-digit decoder is a simple task because it is sufficient to use the selection functions by applying them to the same input (figure 5).

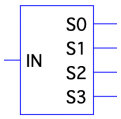

Figure 5: MVL Decoder 1x4

$$S_i(IN) = \begin{cases} L_1 \;\; if \;\; IN = i \\ L_0 \;\; if \;\; IN \neq i \end{cases} \tag{15}$$

Based on the value of the input *IN*, it will select the corresponding output (level L_1), while all other outputs will be zero (level L_0), according to what reported in table 5.

Table 5
MVL True Table

IN	S_0	S_1	S_2	S_3
L_0	L_1	L_0	L_0	L_0
L_1	L_0	L_1	L_0	L_0
L_2	L_0	L_0	L_1	L_0
L_3	L_0	L_0	L_0	L_1

Since we have two one-value decoders, it is possible to construct a decoder with $r = 2$ and $4^2 = 16$ outputs (figure 6) using an AND gates array that realizes all possible combinations of pairs relative to the outputs of the two decoders. The figure 6 shows the circuit with the components *X2* and *X3* representing single-digit decoders. In this case, if the input is the number 13_4 ($IN_1 = L_1$ and $IN_0 = L_3$) the only enabled digit in output would be O_7 (logic level L_1) .

Encoder The encoder is a combinational circuit too, that receives as input n^r MVL digits and returns as output r MVL digits. It performs the opposite function to the decoder: it receives a decoded representation as input and provides the corresponding encoding as output. The figure 14 shows an encoder with 16 inputs and 2 outputs. Each IN_i input passes through a S_i switch that detects the L_1 level at the input (equation 16).

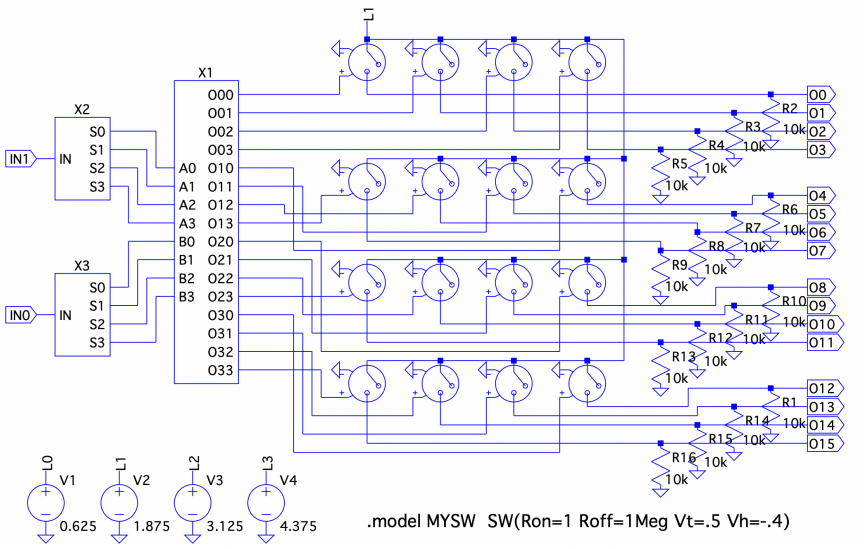

Figure 6: MVL Decoder 2x16

$$S_1(IN_i) = \begin{cases} L_1 \ \ if \ \ IN_i = L_1 \\ L_0 \ \ otherwise \end{cases} \tag{16}$$

To make the circuit simpler, we grouped the selector switch outputs that would determine the same value on the output signals $OUT1$ and $OUT2$. Note that the output of each enabled switch causes its closure, which brings the output level value. For example, if $IN7 = L_1$ the output of selector $SEL8$ enables the OR gates that determine $OUT1 = L_1$ and $OUT0 = L_3$.

Demultiplexer The demultiplexer (or simply *demux*) is a circuit typically used to restore multiplexing produced by a multiplexer in communications. Its treatment in MVL can be found in [10]. It consists of n^r data inputs, r selection signals and an output. It takes a value equal to the input line corresponding to the decoding of the data word. The figure 8 shows the simplest MVL demux with 4 data lines and selection input SEL of data lines. To make the circuit, we used a selector block (module *X2*) that contains the 4 basic selectors. The output of the block determines which signal OUT_i of the circuit will end up in the IN input. The equation 17 describe the behavior of the output.

$$OUT_i = \begin{cases} IN \ \ if \ \ S_i = L_1 \\ L_0 \ \ otherwise \end{cases} \tag{17}$$

For example, if $SEL = L_3$ there will be $S3 = L_1$. The output signal from the selectors block $X2$ enables the corresponding switch by propagating the IN signal to the output OUT_3.

Shifter In this section we will look exclusively at parallel-type shifters, which are normally used to perform mathematical operations (e.g. multiplication and division by base). In general, having a summation and a shifter makes it possible to perform multiplication and division operations for general numbers as long as they are expressed in terms of the base [20]. An example of multiplication between 17 and a generic number x, with $x \in \mathbb{N}$, performed through the use of shifting and addition, is given in the equation 18.

$$17 \cdot x = (4^1 + 4^0) \cdot x = 4^1 \cdot x + 4^0 \cdot x \tag{18}$$

Right Shifter The shifter right moves the digits to the right by as many positions as specified in a dedicated line. For integers this corresponds to performing a division of the number with respect to the base. Since these are components that can be used in series to perform operations on multiple digits, we must provide for receiving and sending these digits to adjacent modules (figure 9). In the following examples, we will use 4-digit words, and the lines of communication with adjacent modules will be 3. Indeed, the variable SEL indicating the number of digits to be moved can take 4 values (including zero).

Right Rotate Shifter In this section we wanted to show the simplicity of implementation of a 4-digit word rotation shifter. A selection input SEL determines the number of shifts in the word (figure 11).

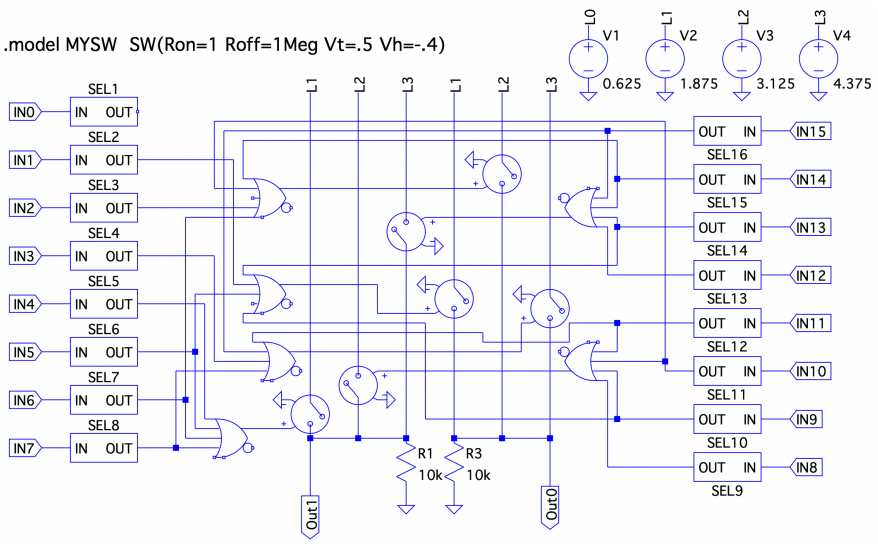

Figure 7: Encoder 16x2

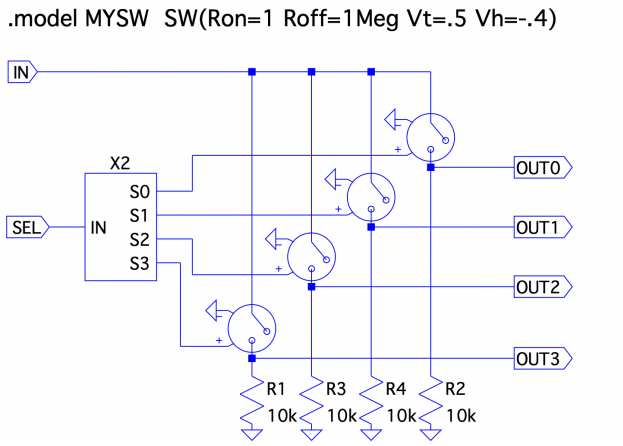

Figure 8: MVL Demux 4x1

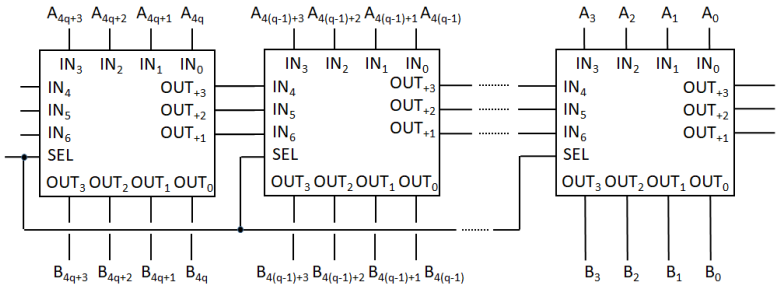

Figure 9: Collegamento in cascata di più shifter right

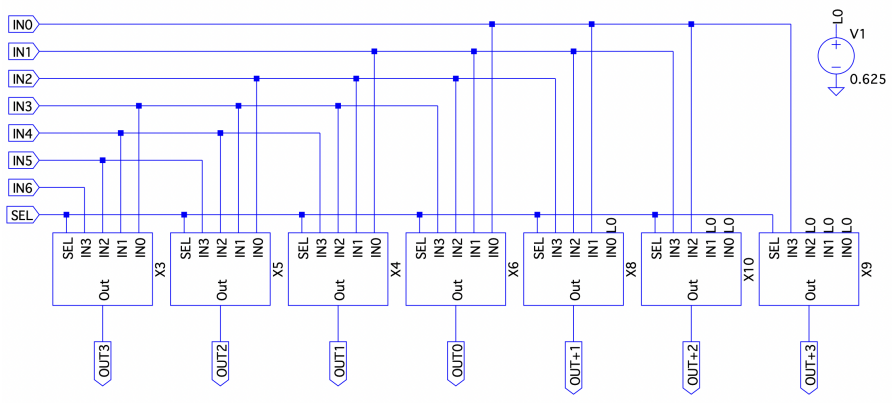

Figure 10: Right Shifter

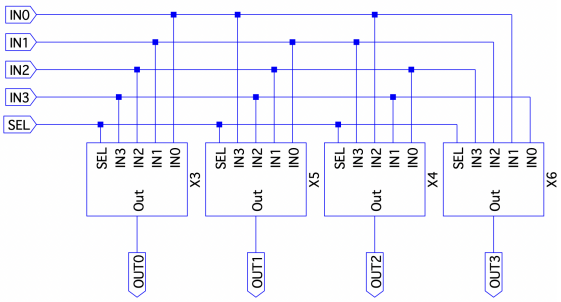

Figure 11: Right Rotate Shifter

Left Shifter Similar to what we described for the right shifter, the left shifter moves the input signals to the left. For integers it corresponds to a multiplication by the base. The figure 12 shows the circuit with 7 data inputs and a selection signal SEL. In this case, each module can expect the arrival of three digits from the one located to its left. Meanwhile, the module under consideration can shift three values of its base word to the left.

In this case if $SEL = 0$, the most significant outputs are set to L_0, enabling the input IN_0 of the first three multiplexers on the left of the figure ($X8$,$X10$, $X9$). The outputs OUT_6, OUT_5 and OUT_4 are set to level zero. In this case the inputs IN_1,IN_2,IN_3 do not compete with the outputs. As the value of the signal SEL increases, the inputs will be shifted to the left until IN_3 moves from the output OUT_3 to OUT_6 ($SEL = 3$) and three new signals will be present on OUT_0, OUT_1 and OUT_3.

3.3. Sequential Circuits

A digital circuit is said to be *sequential* if the output depends on the inputs applied and the state of the circuit. In contrast, in combinational circuits the output is uniquely determined by the values of the inputs. Thus, a sequential circuit has the ability to store information through feedback in the circuits that allows the informative content calculated in the past to be brought back into the input due to the non-instantaneous propagation times in the semiconductors. We will then realize the D latch circuit in MVL logic.

D Latch The schema of the D latch is shown in figure 13, and it consists of two cascaded multiplexers in which the output of the first one determines the selection of the input of the second one. The data signal D lands in all data inputs of the first mux except the input IN_0. Indeed, in IN_0 comes the output data of the second mux. A clock signal CLK, which intermittently takes on the values L_0 and L_1, allows selection between the new data present in D ($SEL \geq L_1$) and the one previously calculated ($SEL = L_0$).

To verify the operation of the D latch, we used an articulated data signal (table 6) so as to check that the circuit had the desired behavior.

4. Limits and Future Works

New circuits in multivalue logic may be a step toward addressing the computational problems of certain applications such as artificial intelligence and blockchain. The amount of time, energy, and resources to deliver certain services is increasing more and more, and traditional circuits seem unresponsive to current needs [21],[22], [23]. We presented several theoretical solutions, and many of

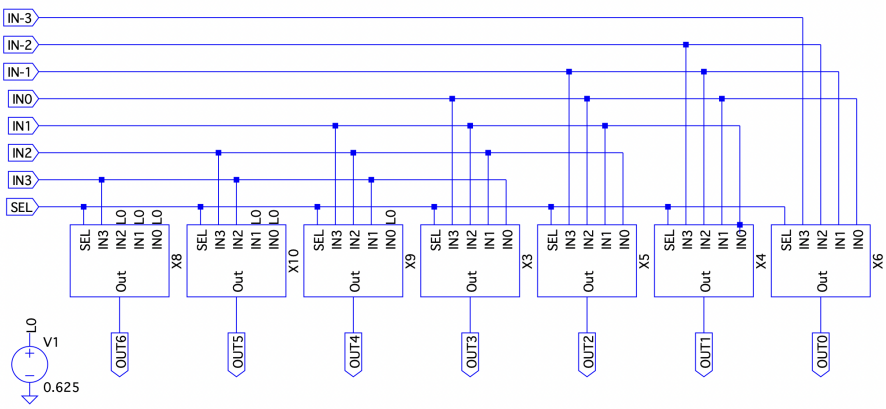

Figure 12: Left Shifter

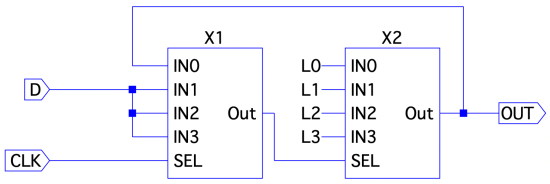

Figure 13: D Latch with MVL MUX 4+1

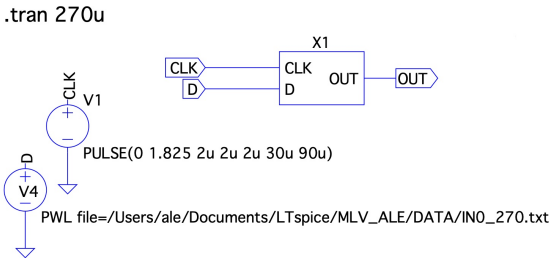

Figure 14: Testing D Latch

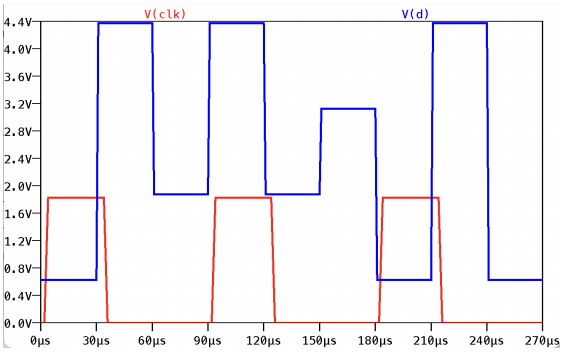

Figure 15: Testing D Latch: inputs

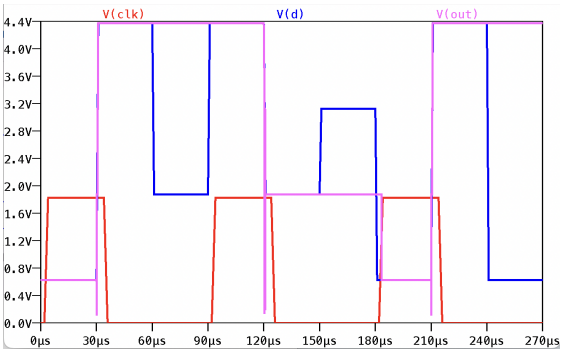

Figure 16: Testing D Latch: outputs

them are aligned with new technological solutions [24], [25], such as mem-resistors [26] or programmable resistor arrays in complex layers [27]. These last components were developed by Massachusetts Institute of Technology (MIT) to create neural networks to support a new area of artificial intelligence called *analog deep learning*.
The lack of integration of models developed with such technologies is one of the limitations that must be overcome to make the use of this logic effective.
The first step in achieving the target of usability of such logic within circuits is their integrability within current systems in digital logic. To this end, it is desirable to employ bases that are powers of two; future efforts will focus on this type of logic.
The proposed work has two important advantages over what has been produced in the literature on MVL systems:

Table 6
Test Input Signals

Time	Volts
0u	0.625
30u	0.625
31u	4.375
60u	4.375
61u	1.875
90u	1.875
91u	4.375
120u	4.375
121u	1.875
150u	1.875
151u	3.125
180u	3.125
181u	0.625
210u	0.625
211u	4.375
240u	4.375
241u	0.625
270u	0.625

- The system we proposed has the ability to work independently of the adopted base. Obviously, the choice of this value must be fixed through the trade-off between expressive power and the internal complexity of the circuits to be made. Just as an example, being able to have a computer that works in hexadecimal would make it possible to reduce the size of memory communication lines, and to perform calculations directly in hexadecimal base by a factor of 4. On the other hand, the number of switches in the decoder shown in figure 6 would increase from 16 to 256 elements. In addition, having designed schemes that works independently of the numbering system base adopted allows the solution to be guaranteed to survive over time in relation to technological evolution. Indeed, this allows the designer to build architectures that are scalable over time.
- The ability to easily integrate with digital systems currently available on the market. This aspect is a strength for any new technology that must not undermine what has been achieved in the past but lead it to a new world.

5. Conclusion

The idea behind this paper is that it is possible to make MVL circuits capable of performing the same functions available to digital circuits. A multi-valued logic system reduces the significant amount of design effort, reduces power consumption, increases the data rate between devices, and minimizes the cost of signals [28, 29, 30, 31].
The paper shows circuits that adopt the same representation system as traditional binary digital ones, but implement them using MVL signals (values). Among the strengths of the presented solution, there is the scalability. Indeed, the presented approach is independent of the base used and thus of the number of levels.
However, the gap with hardware technologies and techniques to support MVL, such as storing MVL values, is far from being closed. The previously mentioned inorganic material, which makes the resistor extremely energy efficient, is an example solution that could accelerate the industrialization of such circuits.
One challenge for the further use of multi-valued logic in circuit design is the creation of an effective computer-aided design package. At present, many multi-core architectures have been developed and implemented, but although parallel computing resulting from multi-core architectures is a solution to increase the performance of a computer [32, 33, 34], with it arise various issues related to the competition and collaboration of cores, especially when the latter are used in the neural networks applications to the solution of different types of problems [35, 36, 37, 38]. We can think of the problem of inaccurate interrupts or information sharing in caches. Such issues are absent in a single computer operating with a computing model close to that of a human.

References

[1] E. Dubrova, Multiple-Valued Logic in VLSI: Challenges and Opportunities, Proceedings of NORCHIP'99 (1999).
[2] D. Etiemble, 45-year CPU Evolution: one law and two equations, Second Workshop on Pioneering Processor Paradigms, Vienna, (2018).
[3] R. A. Jaber, B. Owaidat, A. Kassem, A. M. Haidar, A novel low-energy cntfet-based ternary half-adder design using unary operators, in: 2020 International Conference on Innovation and Intelligence for Informatics, Computing and Technologies (3ICT), 2020, pp. 1–6. doi:10.1109/3ICT51146.2020.9311953.
[4] R. Zhang, N. Uetake, T. Nakada, Y. Nakashima, Design of programmable analog calculation unit by implementing support vector regression for approximate computing, IEEE Micro 38 (2018) 73–82. doi:10.1109/MM.2018.2873953.
[5] A. Navidi, R. Sabbaghi-Nadooshan, M. Dousti, Tqcasim: An accurate design and essential simulation tool for ternary logic quantum-dot cellular automata, Scientia Iranica (2021). URL: http://scientiairanica.sharif.edu/article_22243.html. doi:10.24200/sci.2021.53471.3256.
[6] X. Wang, P. Li, C. Jin, Z. Dong, H. H. C. Iu,

General modeling method of threshold-type multivalued memristor and its application in digital logic circuits, International Journal of Bifurcation and Chaos 31 (2021) 2150248. URL: https://doi.org/10.1142/S0218127421502485. doi:10.1142/S0218127421502485.

[7] 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. URL: http://ijaseit.insightsociety.org/index.php?option=com_content&view=article&id=9&Itemid=1&article_id=5966. doi:10.18517/ijaseit.8.4.5966.

[8] W. Alexander, The ternary computer, Electronics and Power 10 (1964) 36–39. doi:10.1049/ep.1964.0037.

[9] B. Choi, K. Shukla, Multi-valued logic circuit design and implementation, International Journal of Electronics and Electrical Engineering 3 (2014). doi:10.12720/ijeee.3.4.256-262.

[10] A. Simonetta, M. C. Paoletti, M. Muratore, A new approach for designing of computer architectures using multi-value logic, International Journal on Advanced Science, Engineering and Information Technology 11 (2021) 1440–1446. URL: http://ijaseit.insightsociety.org/index.php?option=com_content&view=article&id=9&Itemid=1&article_id=15778. doi:10.18517/ijaseit.11.4.15778.

[11] "Nagarathna, S. Jamuna, "a brief review on multiple-valued logic-based digital circuits", in: T. "Senjyu, P. N. Mahalle, T. Perumal, A. Joshi (Eds.), "ICT with Intelligent Applications", "Springer Singapore", 2022.

[12] N. I. Chernov, N. V. Butyrlagin, N. N. Prokopenko, V. Y. Yugai, Synthesis and circuitry of multi-valued digital current logic elements, in: 2021 International Conference on Electrical Engineering and Photonics (EExPolytech), 2021, pp. 104–107. doi:10.1109/EExPolytech53083.2021.9614916.

[13] N. K. Naware, D. S. Khurge, S. U. Bhandari, Review of quaternary algebra & its logic circuits, 2015 International Conference on Computing Communication Control and Automation (2015) 969–973.

[14] D. Rene, Random Testing of Digital Circuits: Theory and Applications, CRC Press, 2020, p. 496.

[15] D. Kaufmann, Formal verification of multiplier circuits using computer algebra, it - Information Technology (2022). URL: https://doi.org/10.1515/itit-2022-0039. doi:doi:10.1515/itit-2022-0039.

[16] D. J. Preston, P. Rothemund, H. J. Jiang, M. P. Nemitz, J. Rawson, Z. Suo, G. M. Whitesides, Digital logic for soft devices, Proceedings of the National Academy of Sciences 116 (2019) 7750–7759. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1820672116. doi:10.1073/pnas.1820672116.

[17] Analog Devices, LTspice, 2022. URL: https://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html.

[18] S. Raksha, Carry Look-Ahead Adder – Working, Circuit and Truth Table, 2020. URL: https://technobyte.org/carry-look-ahead-adder-working-circuit-truth-table/(May2022).

[19] WatElectronics, What is Carry Lookahead Adder : Block Diagram and Its Working, 2021. URL: https://www.watelectronics.com/carry-lookahead-adder/(May2022).

[20] A. S. Tanenbaum, Structured Computer Organization, Pearson, 2012, p. 808.

[21] X. Chen, Y. Wardi, S. Yalamanchili, Power regulation in high performance multicore processors, in: 2017 IEEE 56th Annual Conference on Decision and Control (CDC), 2017, pp. 2674–2679. doi:10.1109/CDC.2017.8264047.

[22] G. C. Cardarilli, L. D. Nunzio, R. Fazzolari, M. Re, F. Silvestri, S. Spanò, Energy consumption saving in embedded microprocessors using hardware accelerators, TELKOMNIKA (Telecommunication Computing Electronics and Control) (2018).

[23] G. Cardarilli, L. Di Nunzio, R. Fazzolari, M. Re, R. Lee, Butterfly and inverse butterfly nets integration on altera nios-ii embedded processor, in: Conference Record of the 44th Asilomar Conference on Signals, Systems and Computers, Asilomar 2010, Conference Record - Asilomar Conference on Signals, Systems and Computers, 2010, pp. 1279–1283. doi:https://doi.org/10.1109/ACSSC.2010.5757737, 44th Asilomar Conference on Signals, Systems and Computers, Asilomar 2010 ; Conference date: 07-11-2010 Through 10-11-2010.

[24] H. Yoo, C.-H. Kim, Multi-valued logic system: new opportunities from emerging materials and devices, J. Mater. Chem. C 9 (2021) 4092–4104. URL: http://dx.doi.org/10.1039/D1TC00148E. doi:10.1039/D1TC00148E.

[25] Z. Zhang, A. Xu, H. T. Ren, G. Liu, X. Cheng, Reconfigurable multivalued memristor fpga model for digital recognition, International Journal of Circuit Theory and Applications (2022). URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cta.3377.

[26] B. Mouttet, Dynamic systems model for ionic memresistors based on harmonic oscillation, arXiv: Mesoscale and Nanoscale Physics (2011).

[27] A. Zewe, New hardware offers faster computation for artificial intelligence, with much less

energy, 2022. URL: https://news.mit.edu/2022/analog-deep-learning-ai-computing-0728.

[28] R. A. Jaber, A. Kassem, A. M. El-Hajj, L. A. El-Nimri, A. M. Haidar, High-performance and energy-efficient cnfet-based designs for ternary logic circuits, IEEE Access 7 (2019) 93871–93886. doi:10.1109/ACCESS.2019.2928251.

[29] K. Kobashi, R. Hayakawa, T. Chikyow, Y. Wakayama, Multi-valued logic circuits based on organic anti-ambipolar transistors, Nano Letters (2018). doi:10.1021/acs.nanolett.8b01357.

[30] G. Magistris, C. Rametta, G. Capizzi, C. Napoli, Fpga implementation of a parallel dds for wide-band applications, in: CEUR Workshop Proceedings, volume 3092, 2021, pp. 12–16.

[31] M. H. Moaiyeri, Z. M. Taheri, M. R. Khezeli, A. Jalali, Efficient passive shielding of mwcnt interconnects to reduce crosstalk effects in multiple-valued logic circuits, IEEE Transactions on Electromagnetic Compatibility 61 (2019) 1593–1601. doi:10.1109/TEMC.2018.2863378.

[32] S. Illari, S. Russo, R. Avanzato, C. Napoli, A cloud-oriented architecture for the remote assessment and follow-up of hospitalized patients, in: CEUR Workshop Proceedings, volume 2694, 2020, pp. 29–35.

[33] G. Lo Sciuto, S. Russo, C. Napoli, A cloud-based flexible solution for psychometric tests validation, administration and evaluation, in: CEUR Workshop Proceedings, volume 2468, 2019, pp. 16–21.

[34] G. De Magistris, R. Caprari, G. Castro, S. Russo, L. Iocchi, D. Nardi, C. Napoli, Vision-based holistic scene understanding for context-aware human-robot interaction, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 13196 LNAI (2022) 310–325. doi:10.1007/978-3-031-08421-8_21.

[35] C. Napoli, F. Bonanno, G. Capizzi, Exploiting solar wind time series correlation with magnetospheric response by using an hybrid neuro-wavelet approach, Proceedings of the International astronomical union 6 (2010) 156–158.

[36] G. Capizzi, G. Lo Sciuto, M. Woźniak, R. Damaševicius, A clustering based system for automated oil spill detection by satellite remote sensing, in: Artificial Intelligence and Soft Computing: 15th International Conference, ICAISC 2016, Zakopane, Poland, June 12-16, 2016, Proceedings, Part II 15, Springer, 2016, pp. 613–623.

[37] N. Brandizzi, S. Russo, R. Brociek, A. Wajda, First studies to apply the theory of mind theory to green and smart mobility by using gaussian area clustering, in: CEUR Workshop Proceedings, volume 3118, 2021, pp. 71–76.

[38] G. L. Sciuto, G. Susi, G. Cammarata, G. Capizzi, A spiking neural network-based model for anaerobic digestion process, in: 2016 International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM), IEEE, 2016, pp. 996–1003.