ISSN:0975-9646

Balanced- Ternary Logic for Improved and Advanced Computing

Shamshad Ahmad, Mansaf Alam
Deptt. Computer Science, JMI, New Delhi.

Abstract:Logical Systems are the essence of our computing machines. They determine the design, understanding and most importantly the performance of the compuuting machines. Contemporary and traditional binary logic is way too old and limited in application and scope. This paper is meant to present balanced ternary logic as the most suitable logical system for our modern computing machines in terms of performance, simplicity, cost and the future prospects that it can bestow upon our modern computing machines. The paper also deals with the fundamental logical gates and operations in balance ternary logic.

INTRODUCTION:

For decades we have been using computing machines which work on the concept of fundamental switches having only two possible states often represented by - 0 and 1, or ON and OFF.[1] We have witnessed innumerable advancements and improvements in the realm of these devices but none to have so strong an impact so as to question this fundamental property of our "modern" computers itself. Consequently, we have been working and improving upon the same binary logic based systems which has obvious limits.

This paper, however, presents another logical system, balanced ternary logic for our computing machines and its advantages over any other proposed logic.

Modern computing devices based on binary logic implement Boolean logic in which the fundamental components of the internal circuitry are required to have only two differentiable states φ_1 φ_2 [2]from where it gets the title as base-2 as well. Binary logic became popular due to the presence of simple and readily available two state switches and also extensive works on the same logic and machines. Based on two state-logic, modern computers lack various aspects expected from a good logical system which we could recover in our machines by implementing balanced ternary logic.

Any logical system can have an associated power set $\mathbf{P_L}$ to hold elements $\{\varphi_1, \varphi_2, \varphi_3, \ldots, \varphi_n\}$ where $n(P_L)$ gives the base $\mathbf{b_L}$ of that particular logic. Likewise, (balanced) ternary logic, also called "flip-flap-flop"[4] may be associated with a power set defined as P_{BT} to hold elements **as** $\mathbf{P_{BT}} = \{\varphi_1, \varphi_2, \varphi_3\}$ and has $\mathbf{n(P_T)=3}$ which is ultimately $\mathbf{b_{BT}}$. This one extra state φ and the fact that $\mathbf{b_{BT}} = \mathbf{3}$, make balanced ternary logic, the supreme logic.

This paper is essentially a work on balanced ternary logic; we will see how it has an essential advantage over other possible logics including binary logic as well. The paper also defines the basic logical gates, the basic logical setups for this logic and also simple arithmetical operations. Balanced ternary logic has the least hardware complexity [5] and thus economical [6]. Unlike binary representation of information, there is no difference between "signed" and "unsigned" numbers when signed numbers are represented in this logic. To further add to its beauty, the amount of conditional instructions, using this logic, decrease twice which necessarily improves the performance and simplifies things.

Special proofs and arguments in the paper shall further embark upon the efficiency, simplicity, versatility and overall supremacy of ternary logic over any other proposed logic. What if the fundamentals of our machines has better accordance with the Nature and informal human thinking? We need not worry; ternary logic has such advantages as well. Using three different states, it can, on the fundamental level itself have states referring to 'True' 'False' and interestingly 'Unknown'[7] which is very relevant to informal human thinking. Moreover, in terms of versatility, balanced ternary logic, with a clever designing, can also provide an economical space for the conventional binary logic as well [12].

BALANCED TERNARY LOGIC:

Balanced ternary logic is a non-binary, multi-valued logic and special case of ternary logic in which the P_T is represented as $\mathbf{P_{BT}}$ where $\mathbf{P_{BT}} = \{1,0,-1\}$, though, for shorthand, we use $\mathbf{P_{BT}} = \{+, 0, -\}$It is noteworthy that (φ) are simple states and that their representation as symbols and integral values carries no physical significance. Here, we define 'Trit' as the name given to the basic unit of information in a balanced ternary logic based machine. We also define 'tryte' for our purpose as a collection of six trits capable of holding 729 unique values.

I - Complexity:

Besides it other numerous advantages, ternary logic turns out to be the least complex logic for computing. A proposed method for measuring complexity, $\mathbf{C_L}$ of a logic would be by examining the maximum number of information represented by fixed **n** number of **φ** for any particular logic. Clearly, the logical system representing the least number of information using **n φ** would be least economical and would be the most complex. Conversely, the logical system that would represent the highest number of information for **n** number of **φ** would win and will be the most simple (least complex) and most economical [6] of all the proposed logics and will have a base equal to **b**.

Let **N** number of numerical information be represented by any logical system with base **b**. We can define a function f(b) to find **N** to give the maximum possible number of information that could be represented using **n φ**.

$$\mathbf{f(b) = b \wedge (n / b)}$$

Clearly, f(b), at the highest value of N, would correspond to the base **b** of most efficient logic. It is seen that with every

possible value of constant **n**, the f(b) for b>1 has the highest corresponding to **b**=3 only, which is in fact the base of ternary logic. It is feasible to restrict all the values of **b** to be a positive whole number as the value of **b** has to be a counting number only. To verify our result, we find the solution to f(b)'=0 for b>1

The solution to the above function is **e**, the base of natural logarithm which again rounds off to the nearest integer three or $\mathbf{b_{BT}}$.

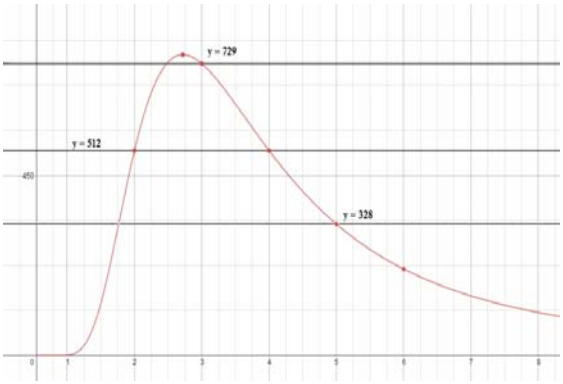

Fig:1 Plot for describing the effective b along with highlighted f(b) of similar multi-valued logic.

II – Information Handling and Representation:

Now that it had been established that balanced ternary logic is the most 'economical' and the 'simple' logic for modern computing devices, we consider the fundamental purpose of computers- information. As stated earlier, $\mathbf{P_{BT} = \{+, 0, -\}}$ which is just a simple representation for **φ** for balanced ternary and that it has no physical significance. However, this representation under balanced ternary logic shall have deep impact upon our understanding and the design of any particular machine itself.

Most of computing is all dealing with information, essentially represented by φ and inferred as numbers to perform actions over them. Therefore, a good and simple way of handling and representation of information becomes crucial.

As per fundamentals, a 16-bit microcomputer with on-board memory cannot access more than 65 Kilobytes of memory directly. However, a similar microcomputer with memory based on (balanced) ternary logic can directly access as much as 43 Mbytes of equivalent memory which is a gain of more than 656 times of memory capacity [10]. Also every binary representation of numbers should have a reserved piece of explicit information to differentiate between "signed" and "unsigned" number, balanced ternary, however, provides more compact way of representing numerical information without such explicit piece of information as it is symmetrical and has elements which are negatives of each other like 1 and -1.

Often, numerical information is stored in a machine using a particular logic with base **b** using the following notation:

$$\mathbf{d_n b^{n-1} + . . + d_3 b^2 + d_2 b^1 + d_1 b^0}$$

Where **d**, the coefficient is the corresponding digit in the information represented using a given logical system. In balanced ternary logic, **d** may have any value from $\mathbf{P_{BT}}$. Here, we however rely upon the integral values of the elements of $\mathbf{P_{BT}}$ for a better visualization and understanding of ternary representation. The further simplicity of the balanced-ternary representation is evident in arithmetical operations based on balanced ternary logic.

Addition of two ternary numbers can be easily performed using the below mentioned ternary adders. Another special feature of ternary logic is that subtraction can also be performed like addition only that the number to be subtracted will have original value but with every **d** inverted. Thus in balanced ternary logic the most widely performed operations become simple. Converting numbers represented in balanced ternary from positive to negative is easily achieved by inverting every $\mathbf{d_n}$ to $\mathbf{d_n'}$, also referred to as inversion, as shown in the table below.

D	D'	D	D'
+	-	-	+
0	0	0	0
-	+	+	-

Table:1 Inversion operation on trits in balanced ternary.

Understanding Balanced Ternary Arithmetic:

Having shown that balanced-ternary logic is the best possible logical system for modern computing devices in terms of simplicity and performance, we will try to set up the basic idea of fundamental logical gates. Though earlier attempts have been made to define the fundamental logic gates like AND and OR as in Kleene and Łukasiewicz logic [12][8]; however, we have proposed our own TAND and TOR logics which we have found to be more practical, mathematically rationale and symmetrical with balanced-ternary logic. We have assumed that 0 state might possess either of + and – and that the result of implementing any logic gate must be as precise as possible. The latter assumption rules out that we might simply put 0 for every resultant trit. Also, in designing these gates, have assumed that + and – are just the negation of each other both in application and logical understanding. Here, we have defined the truth tables for these logical gate that are more consistent with the arithmetical operations and mathematical rationale based on our assumptions.

I-TERNARY 'OR' (TOR) :

Ternary OR gate implements logical disjunction. The resultant outputs on implementing the proposed TOR gate has been shown in the following truth table. Though, it is just a co-incidence that if none of the inputs is 0, and if + is treated as binary 1 and – as binary 0, we get a OR truth table similar to binary logic's, we can exploit it for binary logic as well.

TOR	+	**0**	-
+	+	+	+
0	+	**0**	-
-	+	-	-

Table:2, representing the truth table for TOR gate for balanced ternary logic.

II- TERNARY 'AND' (TAND) :

Ternary AND gate implements logical conjunction. The resultant outputs on implementing the proposed TOR gate has been shown in the following truth table.

TAND	+	0	-
+	+	0	0
0	0	0	0
-	0	0	-

Table:3 representing the truth table for TOR gate for balanced ternary logic.

It is noteworthy, as assumed earlier, in the above table that the resultant value of the above defined gates has been on the basis that they have a hundred percent chance on the implementation of these logical gates. Now that these two fundamental gates have been defined, the other two other frequently used gates- NAND and NOR gates for balanced ternary logic may be defined using the above logic gates by simply inverting (inversion, defined earlier) results of corresponding logics using Table:1.

III- HALF ADDER FOR BALANCED TERNARY LOGIC:

Based on the above defined fundamental gates, we build a balanced ternary half adder to take in two trits and process them to give sum trit S and carry trit C. It uses two TAND gates, a TOR gate, a inverter (shown in block) for inversion, and a special switch sw which allows the input 'through' it only when both the externally controlling inputs are non-0 otherwise it results in 0. The resultant outputs for corresponding inputs have been shown in table 4.

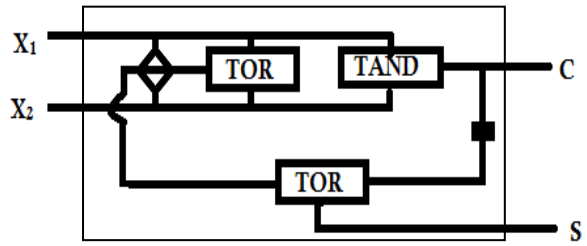

Fig:2 Balanced-Ternary Half Adder logic diagram.

INPUT		OUTPUT	
X_1	X_2	S	C
-	-	+	-
+	+	-	+
-	+	0	0
+	-	0	0
0	0	0	0
0	+	+	0
0	-	-	0
+	0	+	0
-	0	-	0

Table:4 Resultant outputs on implementing balanced ternary half adder on every possible set of inputs.

IV- FULL ADDER:

A balanced-ternary full adder [fig 3] is designed upon the above defined ternary half adder. It takes in three trits as input; essentially designed upon three ternary half adders, it returns two trits- sum S_o and carry C_o. To construct balanced ternary full adder, we assume half-adders **THA** defined earlier as its fundamental structures. It uses three ternary half adders to perform the operations successfully. The resultant values after implementing this balanced-ternary full adder for corresponding inputs is shown in table 5.

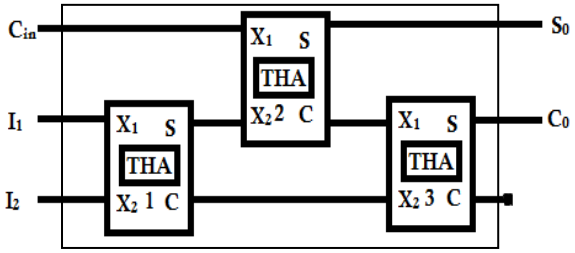

Fig:3 balanced ternary full adder logic diagram.

Note here that that C from THA 3 has been left out as unconsidered and it still causes no problem. We also found that in every possible case, its state will remain 0 only.

Input			Output	
I_1	I_2	C_{in}	S_0	C_0
-	-	-	0	-
+	+	+	0	+
-	-	0	+	-
+	+	0	-	+
-	-	+	-	0
+	+	-	+	0
-	0	-	+	-
+	0	+	-	+
-	0	0	-	0
+	0	0	+	0
-	0	+	0	0
+	0	-	0	0
-	+	-	-	0
+	-	+	+	0
-	+	0	0	0
+	-	0	0	0
-	+	+	+	0
+	-	-	-	0
0	-	-	+	-
0	+	+	-	+
0	-	0	-	0
0	+	0	+	0
0	-	+	0	0
0	+	-	0	0
0	0	-	-	0
0	0	+	+	0
0	0	0	0	0

Table:5 Resultant outputs on implementing balanced ternary full adder on every possible set of inputs.

'NATURAL' NATURE OF BALANCED TERNARY:

The implementation of ternary logic gives the machines more prospects for informal and natural thinking over prevailing binary logic which has only two recognizable states corresponding to true or false only, at the fundamental level. Although, instances of informal thinking and improved cognitive nature can be achieved on binary machines as well but it is not pure or native. However, in ternary logic, due to the fact that balanced ternary provides an extra value to be translated into anything else other than formal 'true' and 'false' like 'maybe'. In fact, ternary logic is the most economical logic that could achieve this ability up to the fundamental level of computing without any explicit definition.

For balanced ternary logic, we can assume a system Ă with elements having the general logical meaning they carry {True, False, may be or unknown} which correspond to different values on mapping to the set $\mathbf{P_{BT}}$ [11]. We should try to map natural propositions, p to a specific element in in the system A. Clearly, if we successfully map Natural Propositions to A, we can easily figure out mapping m for every p in P_{bt} as follows:

Let $\mathbf{p} : Ă \rightarrow P_{bt}$

Then, $\mathbf{m(p)}$= {1, if p is true }
{-1, if p is false}
{0, if p can't be mapped to 1 or -1}

It is clear that balanced ternary logic provides a fundamental representation to all the natural propositions which may be 'true', 'false' or 'maybe'. It can be inferred that due to the informal thinking flexibility provided by balanced ternary logic, it will also provide improved suitable logical system for improved cognitive thinking as well.

If a machine be designed properly with the implementation of balanced ternary logic, it would also perform far better than contemporary binary machines in AI domains as well. For instance, balanced ternary logic based machines have big hopes on their side to performance extremely well on natural thinking tests such as Turing test [14] as they have an extra state to invoke something other than the conventional 'true' and 'false'. If designed properly upon balanced ternary logic, machines could achieve somewhat human-like thinking. Beyond any doubt they will also improve techniques such as Natural Language Processing (NLP) as well.

Converting chunks of text into more formal representations such as first-order logic structures make it easier for computer programs to manipulate. Natural language understanding involves the identification of the intended semantic from the multiple possible semantics which can be derived from a natural language expression which usually takes the form of organized notations of natural languages concepts [13]. True/ false/ Unknown or Maybe, provide more prospects for improved sematic formalization and thus improved NLP.

CONCLUSION:

Implementation of ternary logic has huge advantages over prevailing binary logic. Implementation of any logic becomes easy once we understand that the fundamental structure of any hardware for a specific machine doesn't depends on its driving force, for instance electric current but on its(here, current's) effects. It allows a neat and smart representation of information and also is very consistent with its logical arithmetic. -The implementation of balanced ternary logic will unlock innumerable horizons for the future prospect of computing devices. To conclude, balanced ternary logic is the overall best logical system for improved computing when all the factors including performance, efficiency, durability and robustness are taken under consideration.

REFERENCES:

[1] Boole, George (2003) [1854]. An Investigation of the Laws of Thought. Prometheus Books. ISBN 978-1-59102-089-9.

[2] Paul, Tomassi (1999). Logic. Routledge. p. 124. ISBN 978-0-415-16696-6.

[3] Lou Goble (2001). The Blackwell guide to philosophical logic. Wiley-Blackwell. p. 309. ISBN 978-0-631-20693-4.

[4] Knuth, D. E.; The art of computer programming, Vol.2. Seminumerical algorithms. -Addison-Wesley, 1969.

[5] Liu D, Swenson C. Trading speed for low power by choice of supply and threshold voltages. IEEE J Solid-State Circuit 1993.

[6] Hurst , S.L, "Multiple-valued logic- its status and its future", IEEE Transactions on Computers, vol. C-33, no.12, pp.1160-1179, December 1984.

[7] The Penguin Dictionary of Mathematics. 2nd Edition. London, England: Penguin Books. 1998. p. 417

[8] Lucasiewicz, Jan O Logice Tr´ojwarkoscioewj English translation: On three-valued logic, in L. Borkowski (ed.), Selected works by Jan Lukasiewicz, North-Holland, Ams-12terdam, 1970, pp. 87-88

[9] Cignoli R. and D'Ottaviano I. and Mundici D. Algebra das l´ogicas de Lucasiewicz´,Universidade Estadual de Campinas, UNICAMP 1995.

[10] Stokes, Jon (2007). Inside the machine: an illustrated introduction to microprocessors and computer architecture. No Starch Press. p. 66. ISBN 978-1-59327-104-6.

[11] Jorge Pedraza, Fundamentals of Ternary Logic, June 2009

[12] Kleene, Josephson 1967. Mathematical Logic. John Wiley. Dover reprint, 2001. ISBN 0-486-42533-9.

[13] Yucong Duan, Christophe Cruz (2011), Formalizing Semantic of Natural Language through Conceptualization from Existence. International Journal of Innovation, Management and Technology (2011) 2 (1), pp. 37-42.

[14] Turing, Alan Computing Machinery and Intelligence, Mind, 1950