Potential Functionality of Multi-valued Tunneling Phase Logic Devices

Hossam A. H. Fahmy* Martin Morf† Richard A. Kiehl ‡

Abstract

The possibility for obtaining high functionality in multi-valued logic gates based on tunneling phase logic (TPL) is examined. Using assumptions based on previous results for binary and ternary TPL, the case of a 3-input 4-phase gate is analyzed. It is found that such gates can equivalently perform a rich combination of the functions XOR, AND, OR, and NOT, and could thereby provide high functionality in a single element.

1 Introduction

The majority of digital circuits use voltage levels to indicate the different logic values. Beside the voltage, current-mode as well as charge-mode (like Charge Coupled Devices [1, 2] or Single Electron Logic [3, 4]) circuits also exist but are less popular. Two other possibilities can be achieved using single electron devices, namely the use of the frequency of a waveform [5](appendix B) or the use of its phase with Tunneling Phase Logic (TPL) [6].

The use of the phase of a waveform to represent logic values in digital circuits has been proposed some time ago [7, 8, 9]. It may also provide the ability to perform multi-valued logic [10] in a simple manner.

Tunneling Phase Logic has been proposed as a viable approach to Terascale integration. However, circuit issues and logic gate design using TPL devices are not yet well understood. The main problem is interconnections between devices. One possible solution to reduce the number of wires needed is to use multi-valued logic. The goal fo this study is to gain an understanding of the potential functionality of multi-valued TPL circuits so that we can estimate the leverage of such an approach.

Two-phase and three-phase logic has been analyzed in detail before using traditional devices [7, 9] and using TPL [6, 11]. The work presented here consists of generalizing this to whatever number of phases that can be practically implemented using TPL. A particular case that is investigated in detail is that of the use of 4 phases to represent 4 logic values. A 3 input gate (each one can be any of the 4 phases) has been studied and the resulting functions are found to be quite rich from the logic functionality point of view.

2 Basic operation

A complete analysis of the operation of TPL gates has not yet been performed, however, some basic properties of the gates have been obtained in computer simulations. In particular, previous results show that a binary TPL gate with a capacitively coupled input signal can perform an inverter operation [6], while ternary TPL gates act as an inverter-equivalent gate which performs a rotation operation [11]. The analysis of TPL gates as a function of fan-in and the operation of gates with four or more phase states has not yet been carried out. In this study, we make two assumptions about this operation in order to allow us to gain some understanding of potential functionality of a gate with 3 inputs and 4 phase states. First, we assume that the input signal is simply the sum of several inputs, which gives a resultant phase and amplitude. Second, we assume that a 4 state gate performs the same rotation rule as found for binary and ternary gates.

3 The 4 phase logic

When the TPL is operated so that it can lock to one of four phases of the pump frequency an odd number of inputs must be used. Otherwise the inputs may cancel each other and a zero output would occur, which is not one of the valid logic values (phases). A C-language code has been written to give the resulting truth table for a TPL gate with any number of phases and inputs. This code gives the results when no rotation is assumed and in complex notation, i.e. the sum of the inputs reduced to the closest valid logic value representation in complex numbers. For the current analysis, it is used to generate the truth table for the 4 phase-3 inputs case. Then, different rotations can be assumed, as in the case of 3 phase logic, and the effect can be investigated manually. Once the general rule of ro-

*Solid State and Photonics Lab, Stanford University, Stanford, California, USA

†Computer Systems Lab, Stanford University, Stanford, California, USA

‡Solid State and Photonics Lab, Stanford University, Stanford, California, USA

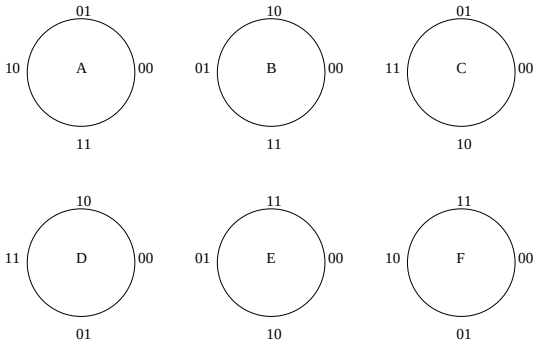

Figure 1: The six possible coding schemes in 4 phase logic.

tation for any number of phases is known, this code can be used to investigate the possibility of using more phases and inputs.

In order to assess the usefulness of the gate it is useful to know its boolean function equivalent. This can be done by giving a binary code $(00, 01, 10, 11)$ to the 4 different logic values and relating the bits representing the output of the gate to the bits representing the inputs by logic functions. It is important to note that the 4 phase case is 4 way symmetric, i.e. if the binary coding assigned to the 4 values is rotated by $\pi/2, \pi$ or $3\pi/2$ the same functions will result. Hence, the total number of different coding is equal to the number of ways the 4 binary codes can be distributed on the 4 values divided by 4. This is : $(4 \times 3 \times 2 \times 1)/4$ which is equal to 6. The six different coding schemes are presented in Fig. 1.

It is interesting to note that the resulting truth table (provided in the appendix) differs in the complex notation result for each coding. However, if the binary code is used instead of the complex number then codings A and F are the same; B and E are the same and C and D are the same. This means that the phase representation is actually somewhat richer (more complex) than the binary code representation. Nonetheless, we will focus on the binary representation in this study. In this truth table, no rotation is assumed, i.e. the binary result is just the code corresponding to the reduced sum.

Simple boolean equations can describe the relations between the outputs and the inputs for different coding schemes. In the following the subscript 0 is used to denote the least significant bit while 1 denotes the most significant bit.

3.1 Logic functions with different rotation cases

In the case of no rotation the logic functions are:

$$A_1 = F_1 \quad = \quad b_1 a_1 + c_1 b_1 + c_1 a_1$$

Table 1: Multiplexer representation for some of the functions

c	b	MUX(1)	MUX(2)
00	00	0	0
00	01	$a_1 \oplus a_0$	a_1
00	10	a_0	$a_1 \oplus a_0$
00	11	a_1	a_0
01	00	$a_1 \oplus a_0$	a_1
01	01	1	0
01	10	$\overline{a_1}$	$\overline{a_0}$
01	11	a_0	$\overline{a_1 \oplus a_0}$
10	00	a_0	$a_1 \oplus a_0$
10	01	$\overline{a_1}$	$\overline{a_0}$
10	10	0	1
10	11	$\overline{a_1 \oplus a_0}$	a_1
11	00	a_1	a_0
11	01	a_0	$\overline{a_1 \oplus a_0}$
11	10	$\overline{a_1 \oplus a_0}$	a_1
11	11	1	1

$$\begin{aligned}
A_0 = F_0 &= MUX(1) \\
B_1 = E_1 &= MUX(2) \\
B_0 = E_0 &= b_0 a_0 + c_0 b_0 + c_0 a_0 \\
C_1 = D_1 &= b_1 a_1 + c_1 b_1 + c_1 a_1 \\
C_0 = D_0 &= b_0 a_0 + c_0 b_0 + c_0 a_0
\end{aligned}$$

where $MUX(1)$ and $MUX(2)$ are the multiplexer functions shown in table 1.

If a rotation of π is performed on every result then for a coding scheme like A, the 00 output with no rotation becomes 10 with the rotation, 01 becomes 11, 10 becomes 00 and finally 11 becomes 01. In all these substitutions, the least significant bit remains the same while the most significant bit is inverted. Hence A_1 with a π rotation is the inverse of A_1 without rotation while A_0 remains the same. A set of such simple observations can be constructed for all the outputs and this gives the following functions for the case of π rotation:

$$\begin{aligned}
A_1 = F_1 &= \overline{b_1 a_1 + c_1 b_1 + c_1 a_1} \\
A_0 = F_0 &= MUX(1) \\
B_1 = E_1 &= MUX(2) \\
B_0 = E_0 &= \overline{b_0 a_0 + c_0 b_0 + c_0 a_0} \\
C_1 = D_1 &= \overline{b_1 a_1 + c_1 b_1 + c_1 a_1} \\
C_0 = D_0 &= \overline{b_0 a_0 + c_0 b_0 + c_0 a_0}
\end{aligned}$$

Similarly, the case of $\pi/2$ rotation can be analyzed and gives:

$$\begin{aligned}
A_1 = F_1 \quad = \quad & (c_1 \oplus c_0)(b_1 \oplus b_0) + (c_1 \oplus c_0)(a_1 \oplus a_0) \\
& + (b_1 \oplus b_0)(a_1 \oplus a_0)
\end{aligned}$$

$$
\begin{aligned}
A_0 = F_0 &= \overline{MUX(1)} \\
B_1 = E_1 &= \overline{MUX(2)} \\
B_0 = E_0 &= (c_1 \oplus c_0)(b_1 \oplus b_0) + (c_1 \oplus c_0)(a_1 \oplus a_0) \\
&\quad + (b_1 \oplus b_0)(a_1 \oplus a_0) \\
C_1 = D_1 &= b_0a_0 + c_0b_0 + c_0a_0 \\
C_0 = D_0 &= \overline{b_1a_1 + c_1b_1 + c_1a_1}
\end{aligned}
$$

While the case of $3\pi/2$ gives:

$$
\begin{aligned}
A_1 = F_1 &= \overline{A_1(\pi/2)} \\
A_0 = F_0 &= \overline{MUX(1)} \\
B_1 = E_1 &= \overline{MUX(2)} \\
B_0 = E_0 &= \overline{A_1(\pi/2)} \\
C_1 = D_1 &= \overline{b_0a_0 + c_0b_0 + c_0a_0} \\
C_0 = D_0 &= b_1a_1 + c_1b_1 + c_1a_1
\end{aligned}
$$

It is clear that the functions performed by the codings A, B, E and F are richer than those performed by C and D specially in the case of $\pi/2$ and $3\pi/2$. This is because each output bit depends on all the input bits and inherently contains a large number of logic functions (XOR, AND, OR and NOT). The 3 input 4 phase TPL can be also used to implement the majority gate which is an important building block for arithmetic circuits (to generate the carry in adders). The function given by A_1 for example in the case of $\pi/2$ rotation is nothing but the majority of the terms between parentheses. The majority is also the function given by the C and D codings for all the rotations.

Since the gates have the ability to perform the XOR and majority functions, they are very suitable to implement arithmetic circuits and large adders and multipliers efficiently. These types of circuits are the basic building blocks required for image processing chips based on 2D Fast Fourier Transform and other circuits proposed for Terascale integration applications [12].

4 Conclusion

The possibility for obtaining high functionality in a 3-input 4-phase multi-valued logic gate has been examined using assumptions based on previous results for binary and ternary TPL. It has been shown that such gates can potentially provide high functionality in a single element. While the detailed results are preliminary in that further work is needed to determine the actual rotation rule and to check other assumptions of the present analysis, these results indicate that multi-valued TPL logic is a promising approach for realizing powerful computation in compact circuits.

References

[1] M. H. Abd-El-Barr, Z. G. Vranesic, and S. G. Zaky, "Algorithmic synthesis of MVL functions for CCD implementation," *IEEE Transactions On Computers*, vol. 40, pp. 977–986, Aug. 1991.

[2] H. G. Kerkhoff and M. L. Tervoert, "Multiple-valued logic charge-coupled devices," *IEEE Transactions On Computers*, vol. C-30, pp. 644–652, Sept. 1981.

[3] P. D. Tougaw and C. S. Lent, "Logical devices implemented using quantum cellular automata," *Journal of Applied Physics*, vol. 75, pp. 1818–1825, Feb. 1994.

[4] S. Bandyopadhyay and V. P. Roychowdhury, "Computational paradigms in nanoelectronics: Quantum coupled single electron logic and neuromorphic networks," *Japanese Journal of Applied Physics*, vol. 35, pp. 3350–3362, June 1996.

[5] H. Fahmy, "Novel digital structures utilizing single electron devices," Master's thesis, Electronics and Communications Dept. Faculty of Engineering, Cairo University, 1997.

[6] T. Ohshima and R. A. Kiehl, "Operation of bistable phase-locked single-electron tunneling logic elements," *Journal of Applied Physics*, vol. 80, pp. 912–923, July 1996.

[7] J. V. Neumann, "Non-linear capacitance or inductance switching, amplifying and memory organs," *US Patent No. 2,815,488*, Dec. 1957.

[8] R. L. Wigington, "A new concept in computing," *Proceedings of the IRE*, pp. 516–523, Apr. 1959.

[9] R. H. Dennard, H. Y. Joliusburger, N. Nakagawa, and G. E. Simaitis, "Logical systems utilizing phase locked subharmonic oscillators," *US Patent No. 3,234,470*, Feb. 1966.

[10] K. C. Smith, "The prospects for multivalued logic: A technology and applications view," *IEEE Transactions On Computers*, vol. C-30, pp. 619–634, Sept. 1981.

[11] F. Liu, F.-T. An, and R. A. Kiehl, "Ternary single electron tunneling logic element," *Applied Physics Letters*, 1999.

[12] J. Earl E. Swartzlander, "Assessment of signal processor architectures and integrated circuit device requirements for computing at the trillion device level," 1998. Unpublished.

Appendix: Truth table for 4 phase logic

c	b	a	A	B	C	D	E	F
00	00	00	3 00	3 00	3 00	3 00	3 00	3 00
00	00	01	2+i 00	1 00	2+i 00	2-i 00	1 00	2-i 00
00	00	10	1 00	2+i 00	2-i 00	2+i 00	2-i 00	1 00
00	00	11	2-i 00	2-i 00	1 00	1 00	2+i 00	2+i 00
00	01	00	2+i 00	1 00	2+i 00	2-i 00	1 00	2-i 00
00	01	01	1+2i 01	-1 01	1+2i 01	1-2i 01	-1 01	1-2i 01
00	01	10	i 01	i 10	1 00	1 00	-i 10	-i 01
00	01	11	1 00	-i 11	i 01	-i 01	i 11	1 00
00	10	00	1 00	2+i 00	2-i 00	2+i 00	2-i 00	1 00
00	10	01	i 01	i 10	1 00	1 00	-i 10	-i 01
00	10	10	-1 10	1+2i 10	1-2i 10	1+2i 10	1-2i 10	-1 10
00	10	11	-i 11	1 00	-i 10	i 10	1 00	i 11
00	11	00	2-i 00	2-i 00	1 00	1 00	2+i 00	2+i 00
00	11	01	1 00	-i 11	i 01	-i 01	i 11	1 00
00	11	10	-i 11	1 00	-i 10	i 10	1 00	i 11
00	11	11	1-2i 11	1-2i 11	-1 11	-1 11	1+2i 11	1+2i 11
01	00	00	2+i 00	1 00	2+i 00	2-i 00	1 00	2-i 00
01	00	01	1+2i 01	-1 01	1+2i 01	1-2i 01	-1 01	1-2i 01
01	00	10	i 01	i 10	1 00	1 00	-i 10	-i 01
01	00	11	1 00	-i 11	i 01	-i 01	i 11	1 00
01	01	00	1+2i 01	-1 01	1+2i 01	1-2i 01	-1 01	1-2i 01
01	01	01	3i 01	-3 01	3i 01	-3i 01	-3 01	-3i 01
01	01	10	-1+2i 01	-2+i 01	i 01	-i 01	-2-i 01	-1-2i 01
01	01	11	i 01	-2-i 01	-1+2i 01	-1-2i 01	-2+i 01	-i 01
01	10	00	i 01	i 10	1 00	1 00	-i 10	-i 01
01	10	01	-1+2i 01	-2+i 01	i 01	-i 01	-2-i 01	-1-2i 01
01	10	10	-2+i 10	-1+2i 10	-i 10	i 10	-1-2i 10	-2-i 10
01	10	11	-1 10	-1 01	-1 11	-1 11	-1 01	-1 10
01	11	00	1 00	-i 11	i 01	-i 01	i 11	1 00
01	11	01	i 01	-2-i 01	-1+2i 01	-1-2i 01	-2+i 01	-i 01
01	11	10	-1 10	-1 01	-1 11	-1 11	-1 01	-1 10
01	11	11	-i 11	-1-2i 11	-2+i 11	-2-i 11	-1+2i 11	i 11
10	00	00	1 00	2+i 00	2-i 00	2+i 00	2-i 00	1 00
10	00	01	i 01	i 10	1 00	1 00	-i 10	-i 01
10	00	10	-1 10	1+2i 10	1-2i 10	1+2i 10	1-2i 10	-1 10
10	00	11	-i 11	1 00	-i 10	i 10	1 00	i 11
10	01	00	i 01	i 10	1 00	1 00	-i 10	-i 01
10	01	01	-1+2i 01	-2+i 01	i 01	-i 01	-2-i 01	-1-2i 01
10	01	10	-2+i 10	-1+2i 10	-i 10	i 10	-1-2i 10	-2-i 10
10	01	11	-1 10	-1 01	-1 11	-1 11	-1 01	-1 10
10	10	00	-1 10	1+2i 10	1-2i 10	1+2i 10	1-2i 10	-1 10
10	10	01	-2+i 10	-1+2i 10	-i 10	i 10	-1-2i 10	-2-i 10
10	10	10	-3 10	3i 10	-3i 10	3i 10	-3i 10	-3 10
10	10	11	-2-i 10	i 10	-1-2i 10	-1+2i 10	-i 10	-2+i 10
10	11	00	-i 11	1 00	-i 10	i 10	1 00	i 11
10	11	01	-1 10	-1 01	-1 11	-1 11	-1 01	-1 10
10	11	10	-2-i 10	i 10	-1-2i 10	-1+2i 10	-i 10	-2+i 10
10	11	11	-1-2i 11	-i 11	2-i 11	2+i 11	i 11	-1+2i 11
11	00	00	2-i 00	2-i 00	1 00	1 00	2+i 00	2+i 00
11	00	01	1 00	-i 11	i 01	-i 01	i 11	1 00
11	00	10	-i 11	1 00	-i 10	i 10	1 00	i 11
11	00	11	1-2i 11	1-2i 11	-1 11	-1 11	1+2i 11	1+2i 11
11	01	00	1 00	-i 11	i 01	-i 01	i 11	1 00
11	01	01	i 01	-2-i 01	-1+2i 01	-1-2i 01	-2+i 01	-i 01
11	01	10	-1 10	-1 01	-1 11	-1 11	-1 01	-1 10
11	01	11	-i 11	-1-2i 11	-2+i 11	-2-i 11	-1+2i 11	i 11
11	10	00	-i 11	1 00	-i 10	i 10	1 00	i 11
11	10	01	-1 10	-1 01	-1 11	-1 11	-1 01	-1 10
11	10	10	-2-i 10	i 10	-1-2i 10	-1+2i 10	-i 10	-2+i 10
11	10	11	-1-2i 11	-i 11	-2-i 11	-2+i 11	i 11	-1+2i 11
11	11	00	1-2i 11	1-2i 11	-1 11	-1 11	1+2i 11	1+2i 11
11	11	01	-i 11	-1-2i 11	-2+i 11	-2-i 11	-1+2i 11	i 11
11	11	10	-1-2i 11	-i 11	-2-i 11	-2+i 11	i 11	-1+2i 11
11	11	11	-3i 11	-3i 11	-3 11	-3 11	3i 11	3i 11