電気学会論文誌Ｃ（電子・情報・システム部門誌）
IEEJ Transactions on Electronics, Information and Systems
Vol.136 No.1 pp.22-27 DOI: 10.1541/ieejeiss.136.22

Paper

A Glitch-Free Time-to-Digital Converter Architecture Based on Gray Code

Congbing Li*[a] Non-member, Haruo Kobayashi* Member

(Manuscript received Feb. 25, 2015, revised July 13, 2015)

A glitch-free time-to-digital converter (TDC) based on Gray code is presented. This architecture can reduce hardware, power consumption, as well as chip area significantly compared to a flash type TDC, while keeping comparable performance and glitch-free characteristics. Its proof-of-concept prototype was implemented on FPGA, and the simulation and measurement results validate the effectiveness of the proposed architecture.

Keywords : Timing Measurement, Time to Digital Converter, Gray Code, Glitch Free, FPGA

1. Introduction

A Time-to-Digital-Converter (TDC) measures the time interval between two edges, and fine time resolution can be achieved when the TDC is implemented with an advanced CMOS process. TDC applications include phase comparators of all-digital PLLs, sensor interface circuits, modulation circuits, demodulation circuits, as well as TDC-based ADCs and phase measurement circuits. The TDC will play an increasingly important role in the nano-CMOS era, because it is well suited to implementation with fine digital CMOS processes(1)-(9).

The architecture of a basic flash-type TDC is shown in Fig.1(1)-(4). The flash-type TDC uses a delay line which consists of CMOS inverter buffer delays and D flip-flops (DFF). However, it has a disadvantage on circuit complexity: for a conventional n-bit delay-chain TDC with 2^n quantization levels, at least a total of $2^n - 1$ delay elements and $2^n - 1$ DFFs are required, leading to large circuitry, high power and chip area.

This paper describes a Gray-code based TDC which can reduce the circuit, power and chip area significantly compared to the flash TDC while keeping comparable performance. We have already proposed a residue arithmetic (RA) TDC(8) for the same purpose, but it has drawbacks of glitches due to the delay element mismatches if we do not take care of them(9). The proposed Gray code TDC alleviates the glitch problem.

Section 2 addresses the residue arithmetic TDC and its glitch problem. Section 3 describes the proposed Gray code based TDC architecture; firstly Gray code is reviewed and its generation with a ring oscillator for TDC design is explained, and then the whole TDC architecture based on Gray code is shown. Section 4 shows FPGA implementation and measurement results of the proposed Gray code based TDCs with 4-bit, 6-bit and 8-bit resolutions. Section 5 shows the RTL simulation and FPGA measurement results that the proposed Gray code based TDC has glitch free characteristics even if there are some amounts of mismatches among delay elements. Section 6 provides conclusions.

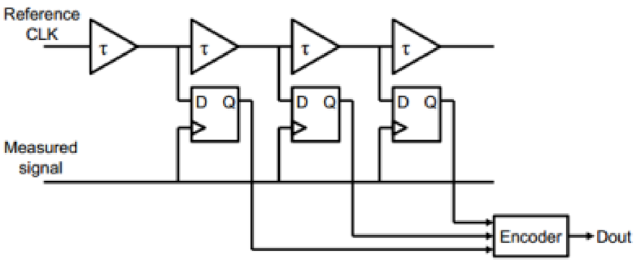


Fig. 1. Flash-type TDC.

2. Residue Arithmetic TDC and its Drawbacks

The drawbacks of the flash architecture motivate the development of residue arithmetic (RA) based TDC(8)(9). This section review the residue arithmetic based TDC and addresses its glitch problem.

2.1 Residue Arithmetic Based TDC

Suppose that m1, …, mr are positive integers and coprime each other. Then there are unique positive integers x for given integers $(a1, a2, \dots, ar)$ which satisfy the following:

$$x \equiv ak\left(mod\, mk\right), \quad k = 1, 2, \dots, r$$

where $0 \le ak < mk, 0 \le x < N \ \left(N = m1 \cdot m2 \cdots mr\right)$

Table 1 shows the case of m1=2, m2=3, m3=5, and we see that each k is mapped to residues of (m1, m2, m3) one to one.

This mathematical basis is called as residue arithmetic and/or modular arithmetic. For example, the above-mentioned algorithm to obtain an integer x from $ak\left(k = 1, 2, \dots, r\right)$ is generally known as remainder theorem(10). Here a large integer x is represented by a set of smaller integers ak , so that computation can be performed independently and in parallel.

We have considered to design a TDC architecture based on the residue arithmetic(9). In the TDC circuit, the signal is "time" instead of "voltage", and the residue can be easily obtained with a ring oscillator. A residue arithmetic based TDC can be conceived by grouping several ring oscillators to operate on the same input. Fig.2 shows an RA-based TDC in the case of m1=2τ, m2=3τ, and m3=5τ. The residues a (mod 2τ), b (mod 3τ), c (mod 5τ) are obtained with three different ring oscillators with the same input.

RA-based TDC can measure the time elapsed between START and STOP incoming pulses and output the residues a, b, c as

a) Correspondence to: Congbing Li. E-mail: t13802483@gunma-u.ac.jp
* Division of Electronics and Informatics, Gunma University
1-5-1, Tenjin-cho, Kiryu, Gunma 376-8515, Japan

Table 1. An integer k and its corresponding residues of (m1, m2, m3).

m1	m2	m3	k
0	0	0	0
1	1	1	1
0	2	2	2
1	0	3	3
0	1	4	4
1	2	0	5
0	0	1	6
1	1	2	7
0	2	3	8
1	0	4	9
0	1	0	10
1	2	1	11
0	0	2	12
1	1	3	13
0	2	4	14
1	0	0	15
0	1	1	16
1	2	2	17
0	0	3	18
1	1	4	19
0	2	0	20
1	0	1	21
0	1	2	22
1	2	3	23
0	0	4	24
1	1	0	25
0	2	1	26
1	0	2	27
0	1	3	28
1	2	4	29

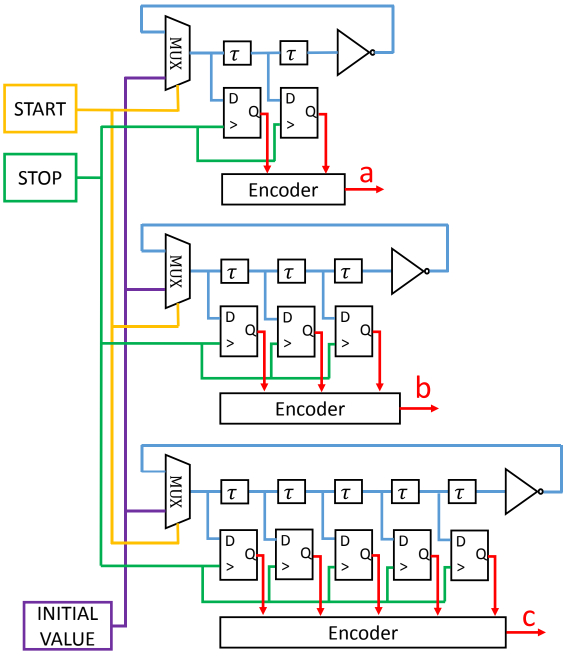


Fig. 2. Residue Arithmetic based TDC architecture.

follows:

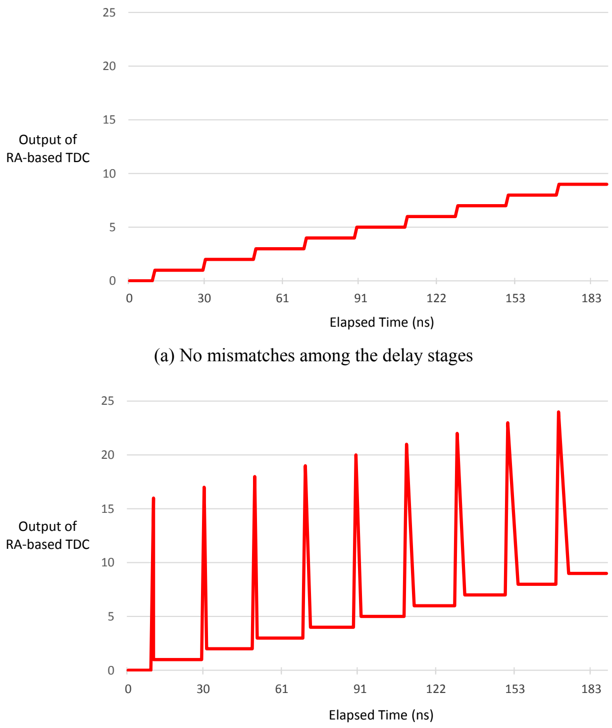
1) When START signal is in LOW state, three ring oscillators are initialized by Initial Value.

2) When START signal goes from LOW to HIGH, three ring oscillators begin to oscillate.

3) When STOP signal is on the rising edge, i.e., LOW-to-HIGH transition, the DFFs are triggered and the value of D is transferred to the output Q.

4) The Q values are transferred into the residues a (mod 2τ), b (mod 3τ), c (mod 5τ) by the encoders.

The corresponding k can be achieved from “a, b, c” based on the residue number system. So the time interval between START and STOP is equal to k×τ.



(a) No mismatches among the delay stages

(b) Mismatches exist among the delay stages (large glitches are observed)

Fig. 3. Simulation results with RA-based TDC without and with mismatches among delay cells in ring oscillators.

In general, RA-based TDC uses M delay cells and M flip-flops (where M=m1+m2+…+mr, additive increase), while the corresponding flash-type TDC uses N delay cells and N flip-flops (where N=m1·m2⋯mr, multiplicative increase). For long measurement time range, the number of delay cells and flip-flops in the proposed TDC decreases rapidly (M<<N) compared with the flash-type TDC, which reduces the hardware and chip area consumption significantly.

2.2 Drawbacks of Residue Arithmetic TDC Although with residue arithmetic TDC architecture small chip area and low power consumption can potentially be achieved, glitches (i.e. out-of-sequence codes) may occur when there are mismatches between the delay stages.

This glitch problem is due to the fact that when k changes from d to d+1, all of m1, m2, …, mr values have to change. However if there are delay mismatches in ring oscillators, some of them may change but others may not in case that the timing difference between START and STOP is on the boundary between d and d+1; this causes a large glitch.

Take RA-based TDC in Fig.2 with τ=20 ns for example. Fig.3(a) shows the output digital codes achieved when all the there are no mismatches among the delay stages. Fig.3(b) illustrates the output digital codes generated when mismatches exist among the delay stages, and we see that glitches occur in the time-domain when there are mismatches among the delay stages: this triggers an instability in the output digital codes.

3. Gray Code Based TDC Architecture

In this section, we show a new type TDC based on Gray code to reduce the hardware and chip power consumption significantly and remove the glitches effectively compared to the flash-type TDC and RA-based TDC, while keeping comparable performance.

Table 2. 4-bit Gray code.

Decimal numbers	Natural Binary Code	4-bit Gray Code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

3.1 Gray Code A Gray code, also known as the reflected binary code, is a binary numeral system where two successive values differ in only one bit (binary digit)(5). Gray code was originally designed to prevent spurious output from electromechanical switches. Table 2 illustrates the difference between Natural Binary and Gray codes.

With Gray code, only one of G3, G2, G1, G0 in 4-bit case changes state from one position (decimal k) to another (decimal k+1). This feature prevents certain data errors which can occur with natural binary code during state changes.

For example, look at the natural binary code sequence in Table 2, the number 7 represented as 0111. As it changes to 8, every bit changes state, to 1000. Every value changes from either 1 to 0, or 0 to 1. If all of these bits changed instantaneously, and synchronously, from one state to the other, there would never be any problem. However, in a highly capacitive circuit (or sluggish system response), some bits would flip before others. Since every bit is changing in this example, depending on the order they change, it might produce an output of any value from this collection of bits. In other words, if for the bits that should be changed for a small amount of the TDC input change, some are changed and the others are not, the TDC output error can be very big in natural binary code.

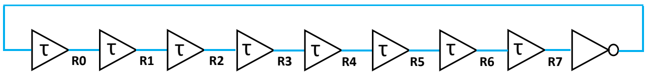
In the 4-bit Gray code sequence in Table 2, between any two adjacent numbers, only one bit changes at a time. Even from position 7 to 8, Gray code only changes one bit state. The sort of error mentioned above is not possible with Gray code, so the data is more reliable. In other words, even if the bit that should be changed does not change for a small amount of the TDC input change, the error is only 1 LSB in Gray code.

Furthermore, this characteristic allows a circuit to perform some error checking, i.e. if more than one bit changes for a small amount of the input change, the data must be incorrect.

3.2 Natural Time-Domain Gray Code Bit Generator

In a ring oscillator, between any two adjacent states, only one output changes at a time. This characteristic is very similar to Gray code.

For example, as illustrated in Fig.4, two successive states of the 8-stage ring oscillator differ in only one output. The output R3 is the same as the Gray code bit G2, and R7 is the same as G3.



State	8-stage Ring Oscillator Output								4-bit Gray Code			
	R0	R1	R2	R3	R4	R5	R6	R7	G3	G2	G1	G0
0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	1
2	1	1	0	0	0	0	0	0	0	0	1	1
3	1	1	1	0	0	0	0	0	0	0	1	0
4	1	1	1	1	0	0	0	0	0	1	1	0
5	1	1	1	1	1	0	0	0	0	1	1	1
6	1	1	1	1	1	1	0	0	0	1	0	1
7	1	1	1	1	1	1	1	0	0	1	0	0
8	1	1	1	1	1	1	1	1	1	1	0	0
9	0	1	1	1	1	1	1	1	1	1	0	1
10	0	0	1	1	1	1	1	1	1	1	1	1
11	0	0	0	1	1	1	1	1	1	1	1	0
12	0	0	0	0	1	1	1	1	1	0	1	0
13	0	0	0	0	0	1	1	1	1	0	1	1
14	0	0	0	0	0	0	1	1	1	0	0	1
15	0	0	0	0	0	0	0	1	1	0	0	0

Fig. 4. 8-stage ring oscillator and 4-bit Gray code.

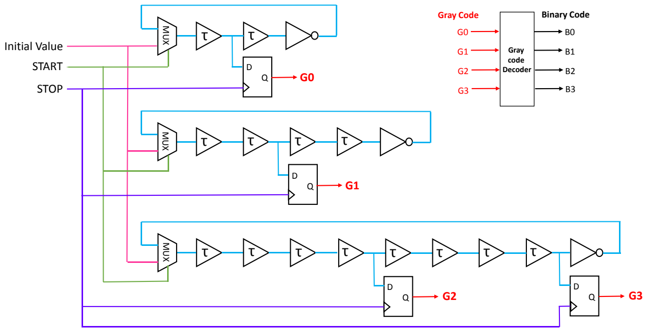


Fig. 5. Proposed Gray code TDC architecture in 4-bit case.

Ring oscillator is a natural time-domain Gray code bit generator. For any given Gray code, its each bit can be generated by a certain ring oscillator.

3.3 Proposed Gray Code Based TDC Architecture As a ring oscillator is a natural time-domain Gray bit code generator(8), a Gray code TDC architecture can be conceived by grouping a few ring oscillators to operate on the same input.

Fig.5 shows the proposed Gray code TDC in 4-bit case. The Gray code bit G0 is generated by a 2-stage ring oscillator, and G1 is generated by a 4-stage ring oscillator, and G2, G3 are generated by 8-stage ring oscillators.

This TDC can measure the time elapsed between START and STOP incoming pulses and output the binary digital representation of the time interval as follows:

1) When START signal is in LOW state, three ring oscillators are initialized by Initial Value.

2) When START signal goes from LOW to HIGH, three ring oscillators begin to oscillate.

3) When STOP signal is on the rising edge, i.e. LOW-to-HIGH transition, the DFFs are triggered and the value of D is transferred to the output Q. Each Q stands for a certain Gray code bit.

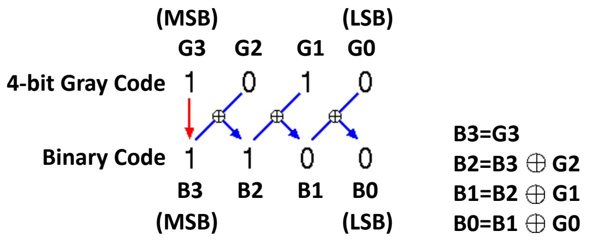


Fig. 6. Gray code decoder.

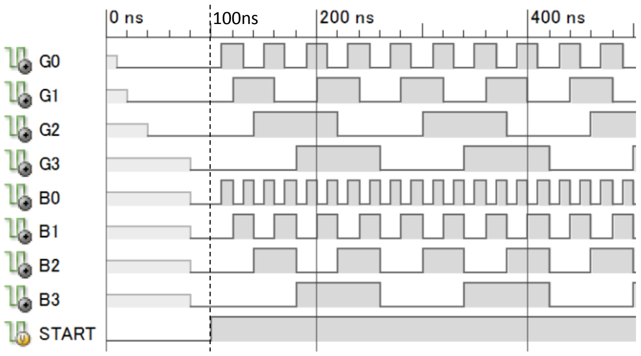


Fig. 7. RTL simulation waveforms of 4-bit Gray code TDC.

Table 3. Gray Code TDC vs. Flash-type TDC.

	Number of Delay Cells	Number of DFFs	Maximum Stage of RO
Gray Code TDC	2^n-2	n	2^{n-1}
Flash-type TDC	2^n	2^n	2^n

4) All the generated Gray code bits are delivered to Gray code decoder, as illustrated in Fig.6, and transferred into binary code, which represents the time interval between START and STOP.

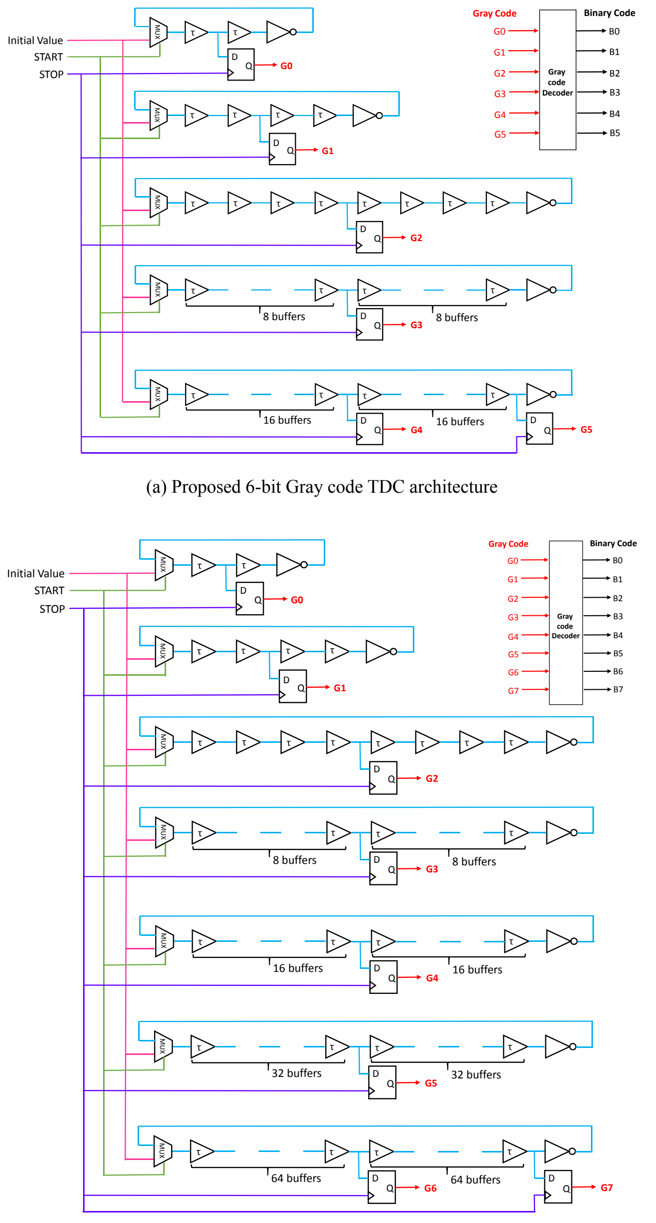
RTL simulation was conducted to verify the characteristics of Gray code TDC in Fig.5. The delay of each buffer is equal to 10 ns, and START signal goes from LOW to HIGH at 100 ns. RTL simulation waveforms are illustrated in Fig.7. We can see that the proposed 4-bit Gray code TDC works as expected in the time domain.

Note that the proposed Gray code TDC uses only 14 delay cells and 4 flip-flops, and the maximum stage of the ring oscillator is 8, while the corresponding flash TDC requires 16 delay cells and 16 flip-flops, and the maximum stage of the ring oscillator is 16.

In general, for a measurement range of 2^n, the proposed Gray code TDC uses 2^n-2 delay cells and n flip-flops (linear growth), and the maximum stage of the ring oscillator is 2^{n-1}, while the corresponding flash-type TDC uses 2^n delay cells and 2^n flip-flops (exponential growth), and the maximum stage of the ring oscillator is 2^n.

For large measurement range, the number of flip-flops in the proposed TDC decreases rapidly ($n \ll 2^n$) compared with flash-type TDC, which reduces the hardware and chip area consumption significantly.

Furthermore, the maximum stage of the ring oscillator is also reduced by half in the proposed TDC. The use of shorter delay lines reduces also the integral non-linearity caused by mismatches between the delay stages.



(a) Proposed 6-bit Gray code TDC architecture

(b) Proposed 8-bit Gray code TDC architecture

Fig. 8. Proposed 6-bit and 8-bit Gray code TDC architectures.

Similarly, 6-bit and 8-bit Gray code TDC architectures can be conceived by a group of ring oscillators (Fig.8).

4. FPGA Implementation of Gray Code TDC

A proof-of-concept 4-bit Gray code TDC in Fig.5 is implemented on FPGA (Fig.9). For the 4-bit Gray code TDC, inputs “START” and “Initial Value” are connected to user push buttons, and “STOP” is connected to 200 MHz FPGA clock. Outputs “B3 B2 B1 B0” are delivered to user LED. Each buffer is realized by a delay flip-flop with 100 MHz clock frequency, i.e. the delay of each buffer is equal to 10 ns.

The measurement starts by pushing the START push button, which produces a rising edge “START” signal. ChipScope is applied to probe the internal signal of FPGA[11][12]. We can see from Fig.10(a) that the proposed 4-bit Gray code TDC works with good linearity as expected. Similarly, proof-of-concept 6-bit and

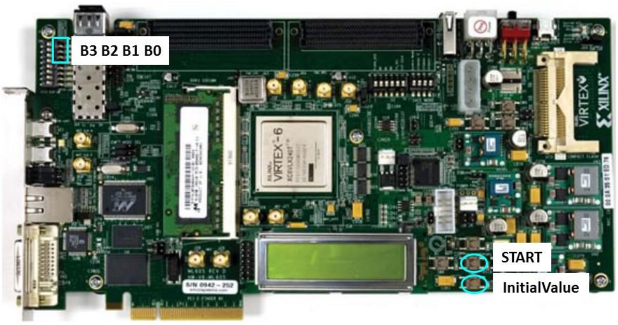
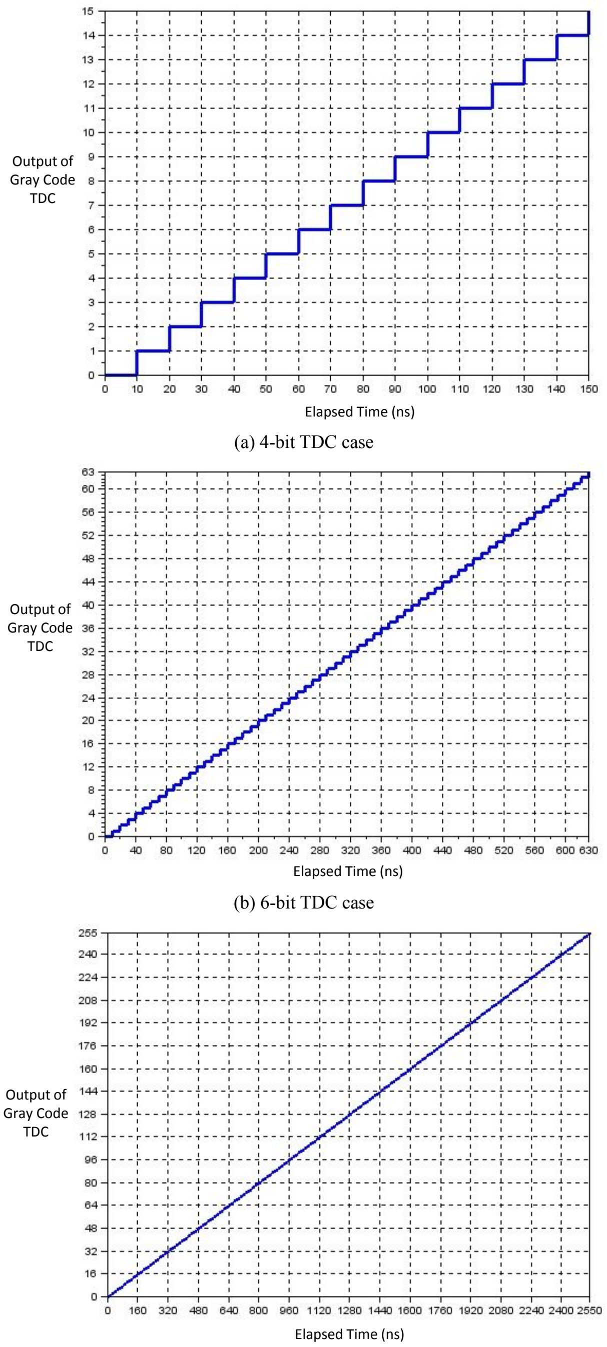


Fig. 9. FPGA Implementation of 4-bit Gray code TDC.



(a) 4-bit TDC case

(b) 6-bit TDC case

(c) 8-bit TDC case

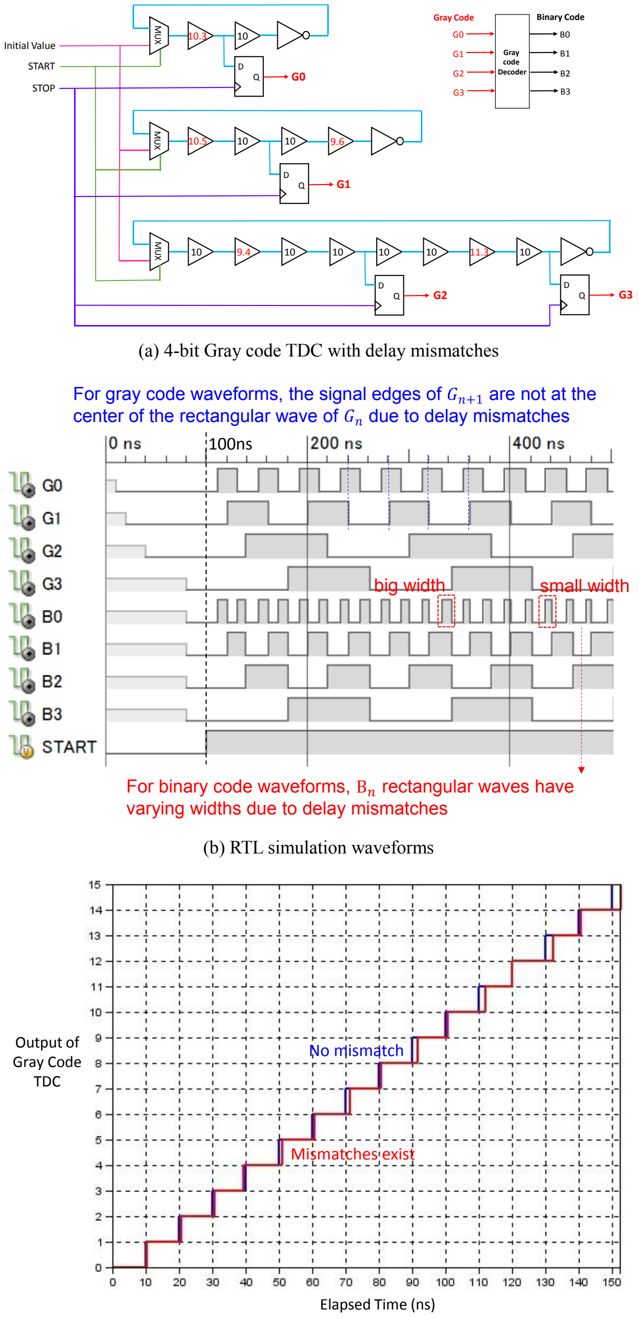
Fig. 10. FPGA measurement results of 4-bit, 6-bit and 8-bit Gray code TDCs.

8-bit Gray code TDC architectures were implemented on FPGA and measured (Fig.10(b), (c)).

5. Glitch-Free Characteristics

The proposed Gray code TDC has a unique characteristic where only one output of the DFFs changes state with each clock pulse, so it can provide a glitch-free binary code sequence, i.e. no out-of-sequence code, even there are some amounts of mismatches among the delay stages.

RTL simulation was conducted to verify this characteristic. In Fig.11(a), mismatches happen among the delay stages in 4-bit Gray code TDC. Fig.11(b) shows the RTL simulation waveforms. From Fig.7, we can see that in no mismatch condition, for Gray code, the waveform G_{n+1} always has signal edges at the center of



(a) 4-bit Gray code TDC with delay mismatches

For gray code waveforms, the signal edges of G_{n+1} are not at the center of the rectangular wave of G_n due to delay mismatches

For binary code waveforms, B_n rectangular waves have varying widths due to delay mismatches

(b) RTL simulation waveforms

(c) FPGA measurement results

Fig. 11. 4-bit Gray code TDC with delay mismatches.

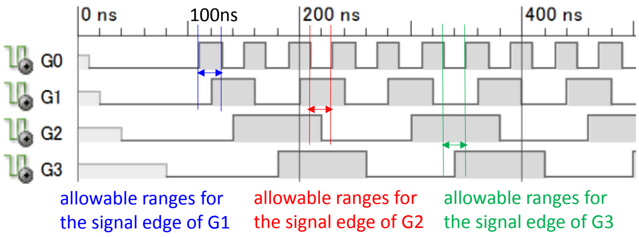


Fig. 12. Allowable ranges for the signal edges of G1/G2/G3 in 4-bit Gray code TDC.

the rectangular wave of G_n, and for binary code, the widths of B_n rectangular waves are always the same. Comparatively, from Fig.11(b), we can see that under mismatch condition, for Gray code, the signal edges of G_{n+1} are not at the center of the rectangular wave of G_n, and for binary code waveforms, the widths of rectangular waves varies due to delay mismatches.

Similarly, a proof-of-concept 4-bit Gray code TDC in Fig.11(a) is implemented on FPGA. Buffers are realized by combinations of delay flip-flops with 100 MHz/200 MHz clock frequency and IODELAY blocks with 200 MHz REFCLK. We can see from Fig.11(c) that even though the code width varies due to the mismatches, the proposed Gray code TDC can still output a glitch-free binary code sequence.

As this glitch-free advantage is achieved from the unique characteristics of Gray code, where only one output changes at a time between any two adjacent states, for Gray code waveforms, the signal edges of G_n $(n>0)$ should be within the range of the corresponding rectangular wave of G0, as illustrated in Fig.12. Otherwise, the unique characteristics of Gray code will no longer exist, i.e. glitches will appear.

The allowable maximum delay mismatch can be calculated based on above analysis. For example, for 4-bit Gray code TDC in Fig.11(a), the allowable maximum delay mismatch between G0 ring oscillator stage and G1 ring oscillator stage is equal to 10 ns.

6. Conclusion

This paper proposed Gray-code based TDC architecture, which reduces hardware and power drastically compared to the flash TDC while keeping performance, and also having glitch-free characteristics which is advantage over the residue arithmetic TDC. RTL simulation and FPGA measurement results validated the effectiveness. We conclude this paper by remarking that our TDC design work used only RTL (without SPICE) simulation and FPGA (instead of full custom CMOS) implementation, which would be suitable for mixed-signal SoC design in nano-CMOS era[13].

Acknowledgements

We would like to thank K. Katoh for his guidance to FPGA implementation as well as STARC for their kind support of this project.

References

(1) Y. Arai and T. Baba : "A CMOS Time to Digital Converter VLSI for High-Energy Physics", IEEE Symposium on VLSI Circuits (1988)

(2) R. B. Staszewski and P. T. Balsara : "All-Digital Frequency Synthesizer in Deep-Submicron CMOS, Wiley (2006)

(3) K. Katoh, Y. Kobayashi, T. Chujyo, J. Wang, E. Li, C. Li, and H. Kobayashi : "A Small Chip Area Stochastic Calibration for TDC Using Ring Oscillator", Journal of Electronic Testing: Theory and Applications, Springer, Vol.30, No.6, pp.653-663 (2014)

(4) T. Chujo, D. Hirabayashi, K. Katoh, C. Li, Y. Kobayashi, J. Wang, K. Sato, and H. Kobayashi : "Experimental Verification of Timing Measurement Circuit With Self-Calibration", IEEE International Mixed-Signals, Sensors and Systems Test Workshop, Porto Alegre, Brazil (2014)

(5) Y. Osawa, D. Hirabayashi, N. Harigai, H. Kobayashi, K. Niitsu, and O. Kobayashi : "Phase Noise Measurement Techniques Using Delta-Sigma TDC", IEEE International Mixed-Signals, Sensors and Systems Test Workshop, Porto Alegre, Brazil (2014)

(6) S. Uemori, M. Ishii, H. Kobayashi, D. Hirabayashi, Y. Arakawa, Y. Doi, O. Kobayashi, T. Matsuura, K. Niitsu, Y. Yano, T. Gake, T. J. Yamaguchi, and N. Takai : "Multi-bit Sigma-Delta TDC Architecture with Improved Linearity", Journal of Electronic Testing : Theory and Applications, Springer, Vol.29, No.6, pp.879-892 (2013)

(7) T. Komuro, J. Rivoir, K. Shimizu, M. Kono, and H. Kobayashi : "ADC Architecture Using Time-to-Digital Converter", IEICE Trans., Vol.J90-C, pp.126-133 (2007)

(8) C. Li, K. Katoh, H. Kobayashi, J. Wang, S. Wu, and S. N. Mohyar : "Time-to-Digital Converter Architecture with Residue Arithmetic and its FPGA Implementation", 11th International SoC Design Conference, Jeju, Korea (2014)

(9) B. Wu, S. Zhu, Y. Zhou, and Y. Chiu : "A 9-bit 215-MS/s Folding-Flash Time-to-Digital Converter Based on Redundant Remainder Number System", IEEE Custom Integrated Circuits Conference, San Jose, CA (2014)

(10) W. A. Chren Jr. : "Low-Area Edge Sampler Using Chinese Remainder Theorem", IEEE Trans. Instrumentation and Measurement, Vol.48, No.4, pp.793-797 (1999)

(11) http://datagenetics.com/blog/november32014/index.html

(12) Xilinx : "Using Xilinx ChipScope Pro ILA Core with Project Navigator to Debug FPGA Applications", [Online] Available: www.xilinx.com

(13) H. Kobayashi, H. Aoki, K. Katoh, and C. Li : "Analog/Mixed-Signal Circuit Design in Nano CMOS Era", IEICE Electronics Express, Vol.11, No.3, pp.1-15 (2014)

Congbing Li

(Non-member) received the B.Eng. degree in communication engineering from Beijing University of Posts and Telecommunications (BUPT) in 1999, the M.Eng. degree in communications and information systems from BUPT in 2003. He was a researcher of digital technology research center, R&D H.Q., SANYO Electric Co., Ltd. Currently he is a Ph.D. course student in electronic and information engineering in Gunma University, Japan.

Haruo Kobayashi

(Member) received the B.S. and M.S. degrees in information physics from University of Tokyo in 1980 and 1982 respectively, the M.S. degree in electrical engineering from University of California, Los Angeles (UCLA) in 1989, and the Ph.D. degree in electrical engineering from Waseda University in 1995. He joined Yokogawa Electric Corp. Tokyo, Japan in 1982, and was engaged in research and development related to measuring instruments. In 1997, he joined Gunma University and presently is a Professor in Division of Electronics and Informatics there. His research interests include mixed-signal integrated circuit design & testing, and signal processing algorithms. He received the Yokoyama Award in Science and Technology in 2003.