American Journal of Engineering Research (AJER)
e-ISSN: 2320-0847 p-ISSN : 2320-0936
Volume-14, Issue-4, pp-26-36
www.ajer.org
Research Paper Open Access

A 127-Step Phase Interpolator for CDR Applications with Enhanced Linearity Using a Novel Discrete Current Modulation Technique

Erick J. Arenas Mendoza[1], Guillermo Espinosa Flores-Verdad[1], Víctor R. González Díaz[2], Gisela de la Fuente Cortés[2]
National Institute of Astrophysics Optics and Electronics, Puebla, México[1]. Autonomous University of Puebla, Puebla, México[2].
Corresponding Author: Erick J. Arenas Mendoza

ABSTRACT : *This paper presents a novel Phase Interpolator topology, where the phase-tracking loop is implemented using a new technique through discrete current modulation profile. The PI provides 127 discrete steps. The PI is digitally controlled by a circuit topology that enables both coarse and fine phase adjustments, significantly enhancing linearity while minimizing Differential Nonlinearity (DNL) and Integral Nonlinearity (INL) errors.*

The design and layout were implemented using 65nm CMOS technology, targeting an operating frequency of 2.5GHz for the PI. This setup achieves an angular resolution of approximately 2.81° per code, with a standard delay of 3.125ps. The proposed system was validated through post-layout simulations, incorporating the PI, Counters and binary2thermo logic. Additionally, Verilog-AMS simulations were used to model Counters and binary2thermo in order to design the corresponding digital logic operations.

KEYWORDS *Clock Data Recovery (CDR), Phase Interpolator (PI), Differential Non-Linearity (DNL), Dynamic Code Change (DCC), Integral Non-Linearity (INL), Modulation Profile (MP).*

Date of Submission: 13-04-2025 Date of acceptance: 28-04-2025

I. INTRODUCTION

The rapid advancement of high-speed serial links has significantly increased the demand for high data throughput in recent years. Modern data transmission systems, particularly those involving high-speed multichannel serial communication—such as CPU-to-CPU connections or data transfers to storage devices—often do not include a synchronous clock with the transmitted data. Consequently, the received data may experience asynchronization and noise interference [1]. To mitigate these issues, Clock Data Recovery (CDR) circuits are employed, enabling accurate data recovery through signal retiming in plesiochronous operation [2]. Among various CDR architectures, the all-digital Phase Interpolator (PI)-based CDR is particularly attractive for multichannel applications [3, 4, 5]. Unlike traditional Phase-Locked Loop (PLL)-based CDRs, PI-based architectures eliminate the need for a PLL, thereby reducing lock time and minimizing cycle-to-cycle jitter accumulation [5, 6]. Furthermore, the digital nature of PI-based CDRs offers improved resilience to process, voltage, and temperature (PVT) variations compared to their analog counterparts. Additionally, in contrast to Delay-Locked Loop (DLL)-based CDRs, PI-based CDRs can shift the clock phase over an unrestricted range, making them particularly advantageous in handling limited frequency offsets between the transmitter (TX) and receiver (RX) [5].

While several studies have explored all-digital quarter-rate PI-based CDRs [3, 4, 5, 7, 8], none have incorporated fine phase control adjustments to enhance system linearity

This paper is organized as follows: Section II introduces the proposed Phase Interpolator (PI) core design with enhanced linearity, detailing the coarse and fine control scheme and circuit implementation. Section III discusses the Dynamic Code Change (DCC) mechanism using a modulation profile. Section IV presents the layout construction and the postlayout results of the interpolators and the digital cells using Verilog-AMS. Finally, conclusions are drawn in Section V.

II. Phase Interpolator Design

The linear approximation phase interpolator response is illustrated in Figure 1. In this graph, the linear approximation response is compared with the ideal phase-to-code mapping to form a perfect circular trajectory, ensuring an ideal phase fit [9]. This linear approximation can be described mathematically by Equation 1.

There are various methods to implement a linear approximation algorithm, utilizing either fully digital circuitry or an analog core such as a Gilbert cell. State-of-the-art implementations of these algorithms can be found in [10, 11, 12, 13]. To summarize the behavior of a linear approximation phase interpolator, we define the functions $a(t)$ and $b(t)$, which establish the relationship between the input control code and the interpolated phase [14].

$$\mathrm{a}(t)\sin\omega\mathrm{t} + \mathrm{b}(t)\cos\omega\mathrm{t} = \sqrt{a(t)^2 + b(t)^2}\cos\left(\omega\mathrm{t} - \tan^{-1}\frac{a(t)}{b(t)}\right) \quad (1)$$

An important consideration regarding phase interpolators is that their control can be either digital or analog, regardless of the core implementation. As discussed in [9] and [15], digitally controlled phase interpolators (PIs) are often limited by speed and low phase resolution. However, this work aims to address the issue of low phase resolution and enhance linearity by implementing a coarse and fine control scheme in the digital domain. Resolution and linearity are two critical parameters in Clock Data Recovery (CDR) systems, both of which are directly influenced by the design and performance of the phase interpolator circuit.

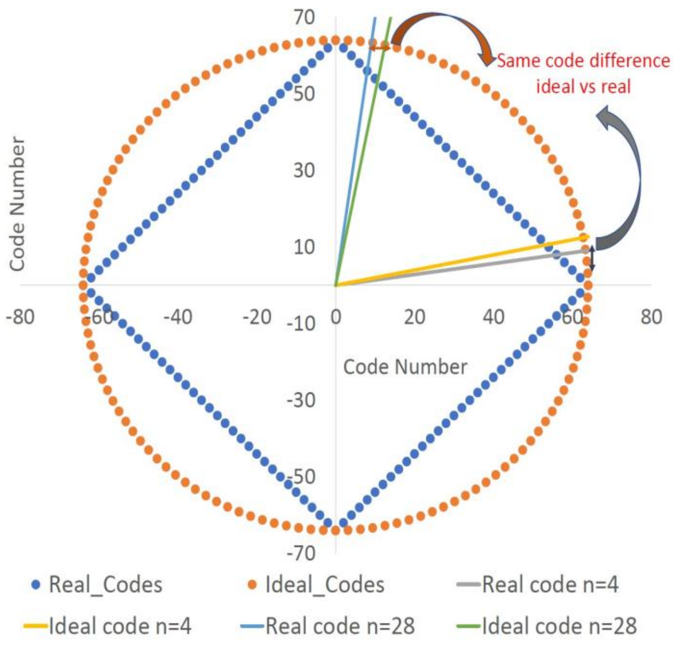

Figure 1: Ideal code versus linear approximation.

Phase Interpolator's Coarse and Fine Control:

The distribution between the ideal code (with a step size of 2.8125°) and the linear approximation code is illustrated in Figure 1.

The common digital approximation technique in a phase interpolator topology using current as a control code is presented in Figure 2. This circuit is based on the well-known Gilbert cell [14]. The phase adjustment is carried out through the current code by taking the four clock inputs in the interpolator and weighting them to create a standard delay at its output.

In order to use the $a(n)$ and $b(n)$ codes in the general interpolator circuit, it is important to show the relationship between amplitude at the interpolator and current in Fig. 2. In equation 2, this relationship is presented, and the correspondence between the branch currents is illustrated in equation 3 [14].

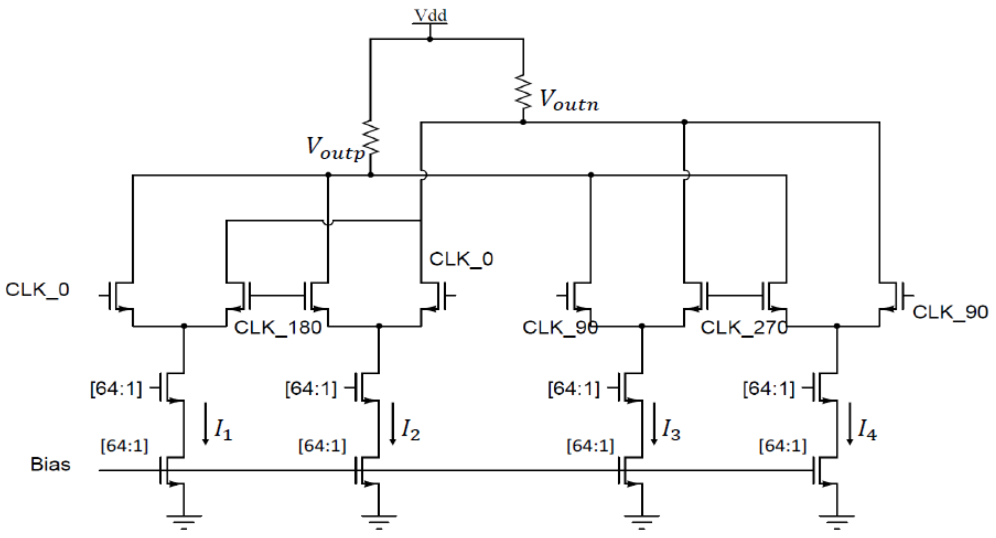

Figure 2: Linear approximation interpolator using current as control code.

$$b(n) \propto (I_1 - I_2)$$
$$a(n) \propto (I_3 - I_4) \quad (2)$$

$$I_1 + I_2 = \frac{stp_{max}}{2} I_0$$
$$I_3 + I_4 = \frac{stp_{max}}{2} I_0 \quad (3)$$

Where I_0 is the current by step and $\frac{stp_{max}}{2}$ is the maximum number of current by steps used. Figure 3 shows the ideal behavior of the current for I_1, I_2, I_3, and I_4 in the first 32 codes across the step by code, using $stp_{max} = 128$. Therefore, $a(n)$ and $b(n)$ can be built by the current difference, as shown in 2.

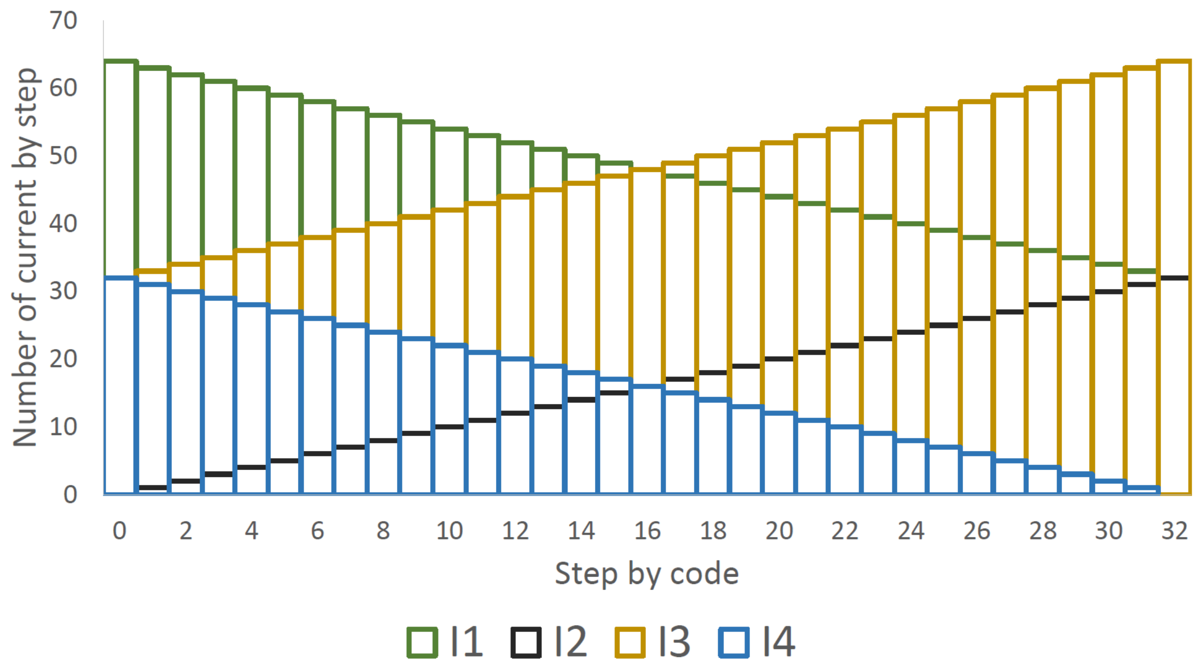

Figure 3: Current behavior in common interpolator topology

Proposal for fine control in a phase interpolator:
Figure 1 shows four symmetrical regions, each consisting of 32 codes. It is observed that if a better match can be achieved in the first 32 codes, the same combination can be repeated for all the other regions. Thus, the behavior of the first 32 codes has been analyzed, e.g., in the first 15 codes, the difference between the ideal phase step and the linear approximation phase step is positive. Therefore, a higher phase is required to reach the ideal phase step. At code 16, the ideal step matches the real step (45˚). On the other hand, in the second set of 15 codes (codes 17 to 31), the difference between the ideal phase step and the linear approximation phase step is negative, indicating that less phase is required to reach the ideal step. Finally, at code 32, the ideal step matches the linear approximation step (90˚). This same behavior is repeated for the next 96 steps.

Taking into consideration these observations, the following proposal has been made: to increment the phase in the first 15 codes, the behavior in $a(n)$ is incremented by one code until the eighth code, and then it is decreased by one code until all eight codes are eliminated. This response is illustrated in Figure 4. Similarly, to decrease the phase in the next set of 15 codes after code 16, the change in $b(n)$ is decreased by one code until the eighth code, and then it is increased by one code until all eight diminishing codes are eliminated. This response is also depicted in Figure 4. Table 1 presents the proposed code for all 128 steps, and the magnitudes of $a(n)$ and $b(n)$ for each code boundary are indicated in the table. The code starts at 0 with a phase of 0˚ and finishes at code 127 with a phase of 357.2297˚.

One way to evaluate the difference between the linear approximation code and proposal code is by calculating the Error per Code (EPC) or Integral Non-linearity (INL) [16], which represents the comparison between the difference of the real and ideal phases with respect to the ideal step size of 2.8125˚. Equation 6 provides the mathematical representation of the error per code [16].

$$INL = EPC = \frac{Phase_{real} - Phase_{ideal}}{Ideal\ phase\ step} \qquad (4)$$

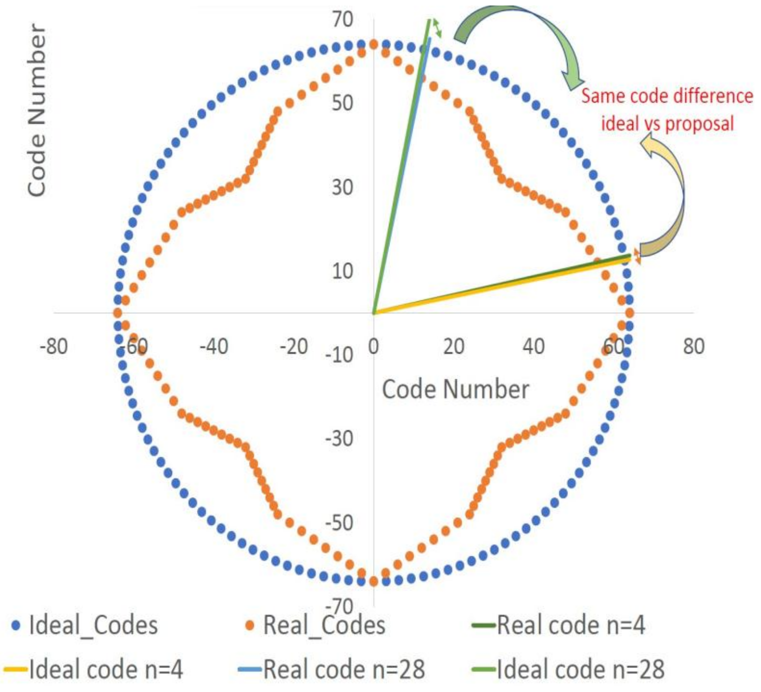

Figure 4: Proposal code change versus ideal code

Table 1: Proposal code change

$b(n)$	$a(n)$	code
$64-2n$	$3n$	$0 \le n \le 8$
$64-2n$	$n+16$	$9 \le n \le 16$
$48-n$	$2n$	$17 \le n \le 24$
$96-3n$	$2n$	$25 \le n \le 32$
$96-3n$	$128-2n$	$33 \le n \le 40$
$16-n$	$128-2n$	$41 \le n \le 48$
$64-2n$	$80-n$	$49 \le n \le 56$
$64-2n$	$192-3n$	$57 \le n \le 64$
$2n-192$	$192-3n$	$65 \le n \le 72$
$2n-192$	$48-n$	$73 \le n \le 80$
$n-112$	$128-2n$	$81 \le n \le 88$
$3n-288$	$128-2n$	$89 \le n \le 96$
$3n-288$	$2n-256$	$97 \le n \le 104$
$n-80$	$2n-256$	$105 \le n \le 112$
$2n-192$	$n-144$	$113 \le n \le 120$
$2n-192$	$3n-384$	$121 \le n \le 128$

The Differential and Integral Non-Linearity plots (DNL and INL) for the first 32 codes are shown in Figure 5. As can be observed from Figure 5, the number of codes that have a magnitude greater than one is sixteen for the INL with linear approximation, whereas with the proposed code, this number reduces to only six. Furthermore, this number can be reduced with some techniques explained in the next section.

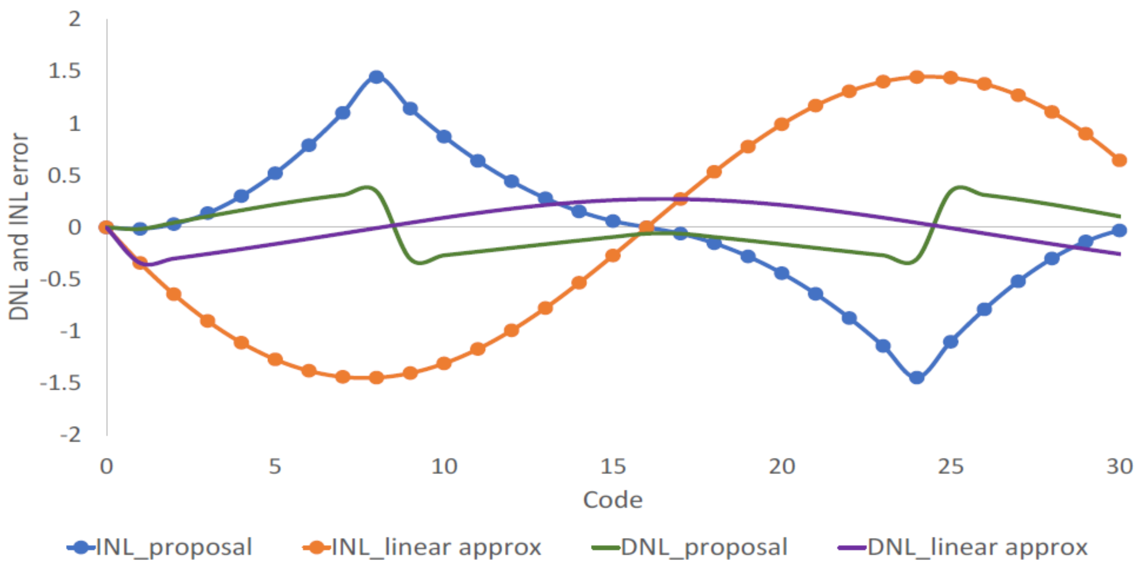

Figure 5: DNL and INL plots

Circuit implementation:
The proposed code can be used to create a transistor-level implementation. In order to achieve a correct implementation, the explanation provided in the previous section is utilized. Figure 6 illustrates how the eight codes used for ne control are implemented in the interpolator. It can be observed that p-type transistors are employed to increase and decrease the eight codes, enabling fine control in the phase interpolator, e.g., in the first 16 codes, the p-type transistors on the right side of the Figure start incrementing the current of I_8, causing the total current by step on the right side to decrease by one for each step by code until reaching eight codes $I_4 - I_8$. Simultaneously, the total code in $a(n) \propto I_3 + I_8 - I_4$ is incremented by $3n$ instead of $2n$, as described in Table 1. Once the eighth code is reached, the p-type transistors on the right side start decrementing by one until the increment in the current vanishes. As a result, the total code from 9 to 16 in $a(n)$ becomes $n + 16$. This approach allows the phase in $a(n)$ for the first 16 codes to rise from the code. The same principle is applied to the remaining 112 codes, taking into account the correct position of current by step in I_5, I_6, I_7, and I_8, as can be observed in Figure 7.

Indeed, as evident in Figure 6, an offset current has been introduced at the source of the differential pairs. This adjustment is made to prevent a substantial step difference between codes with enable currents close to zero and other codes with higher enable currents. On the other hand, one of the most important parts of this circuit implementation is the variable resistor (which can also be seen as a variable current source in the p-mirror transistor). By tuning the variable resistor, it becomes possible to create a dynamic code (with a default value of 1) for fine control in the phase interpolator. Both techniques are very useful to reduce not only DNL but also INL in order to improve the linearity of the system.

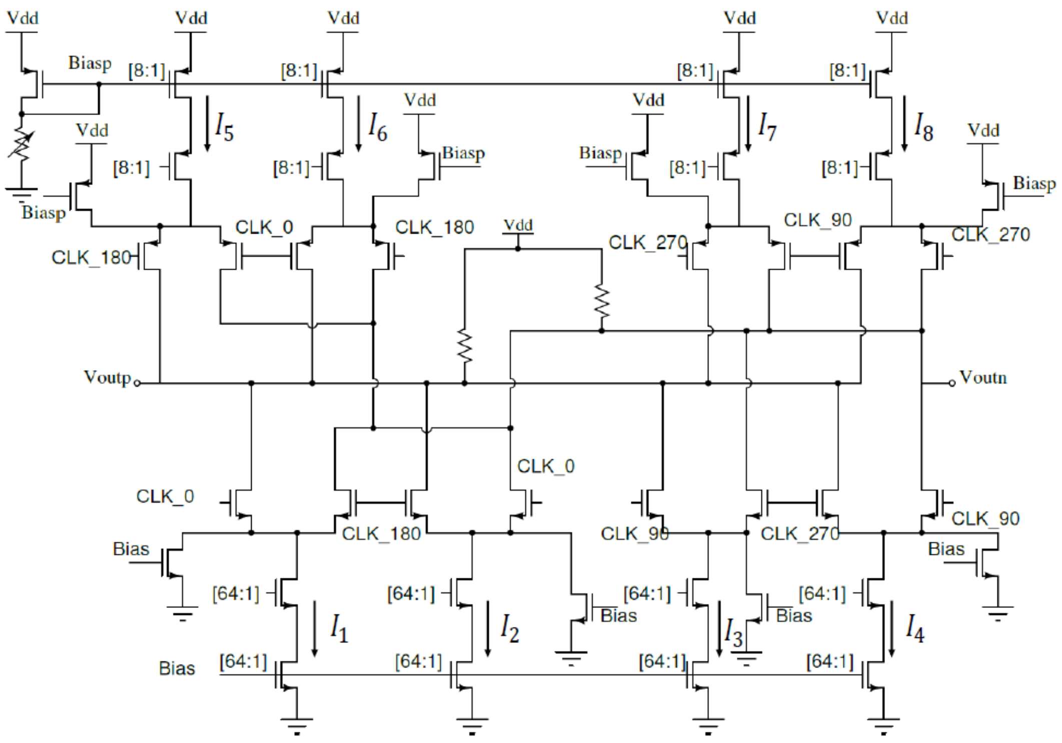

Figure 6: Circuit implementation proposal.

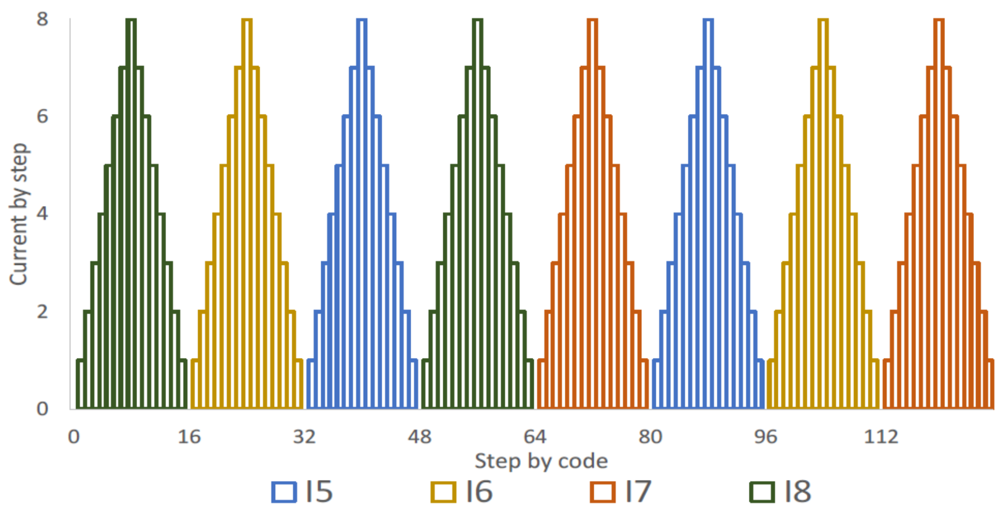

Figure 7: Distribution of currents I_5, I_6, I_7, and I_8. Discrete triangular signal for eight PIs (modulation profile construction).

III. Dynamic Code Change using a Modulation Profile

After demonstrating the novelty of the topology implementation, a second proposal is developed to achieve a better phase fit using both concepts explained in the previous sections: the dynamic code and the distribution of currents, which are used to build a modulation profile. The equations used to achieve a perfect phase fit using the Dynamic Code Change (DCC) of the proposed algorithm are shown in Equations 5 and 6.

$$tan^{-1}\left(\frac{code(y)+DDC(P_{code})}{code(x)}\right) = IPS - case1 \qquad (5)$$

$$tan^{-1}\left(\frac{code(y)}{code(x)+DDC(P_{code})}\right) = IPS - case2 \quad (6)$$

Where the Ideal Phase Summation (IPS) is the sum of the ideal phase for each code, and the case1 or case2 depend if IPS is in the first sixteen codes (case1) or the second sixteen codes (case2). Figure 8 shows the ideal DCC pattern and a proposed modulation profile with 6% of the maximum ideal DCC (1.045) as the step. If the ideal DCC is implemented in the fine control of this PI, a theoretical zero DNL and INL errors are gotten. In Figure 9 the new code change pattern is compared with the ideal code.

Although the ideal DCC modulation pattern has a theoretical zero error, in practical implementation the ideal modulation pattern is not followed due to mismatch between current sources, process variations and randomness in the fabrication process. Taking all of these into account, the digital pattern created by this proposal in the previous sections (Figure 7) is leveraged to generate a discrete modulation profile (DMP) where the only adjustment parameter is the discrete current step for the modulation profile.

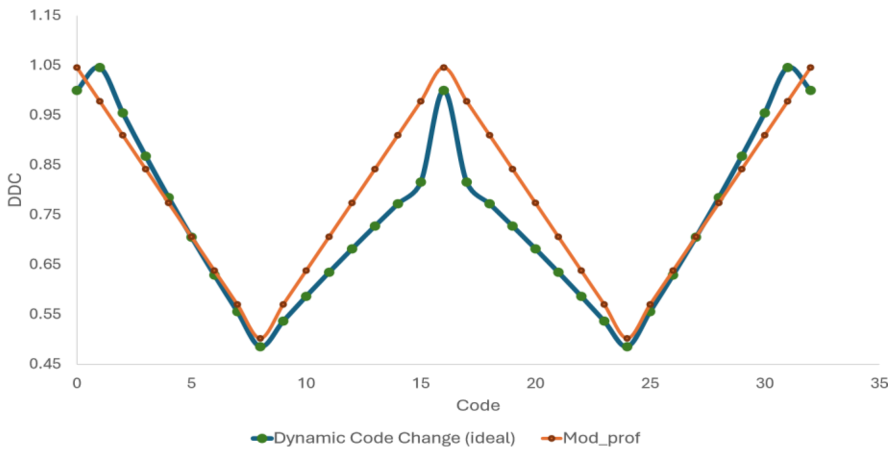

Figure 8: Ideal DCC modulation pattern and proposed modulation profile.

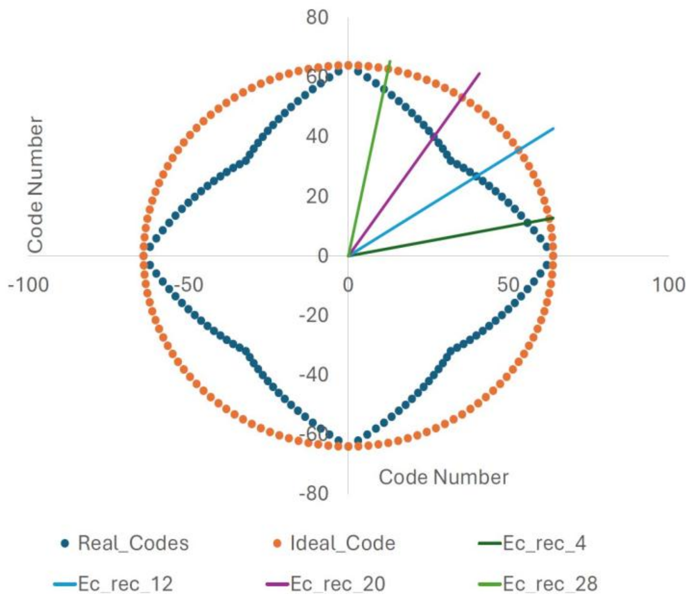

Figure 9: New code change pattern based on ideal DCC vs ideal code.

IV. Simulation Results

To guarantee proper operation and high-speed performance, the current mirror transistors must be biased in the saturation region. To mitigate shortchannel second-order effects, all transistors in the current mirror stage are designed with channel lengths greater than the minimum allowed by the technology. The proposed operating current for achieving saturation is 10μA per code. Since the proposed topology operates in a manner similar to a Gilbert cell, the phase interpolator functions in large-signal operation. Consequently, the most critical parameters for bandwidth are the *RC* filter in the PI output stage and the current amplitude of each code.

The capacitance value is set to the minimum allowable for the technology, which is 100fF. This capacitance is sufficient to filter out undesirable harmonic frequencies while maintaining high-speed performance, preventing signal degradation at the interpolator output.

Layout implementation:

In Figure 10, the proposed layout is shown. The design of the physical implementation requires that metal layers not overlap, or the overlap should not be between two close metal layers, in order to avoid an increment of parasitic capacitance, all of this due to the high operation frequency and the small resolution of the interpolator. The sizes of this proposed implementation are: length=144.9μm and height=32.4μm, where it has been included the interpolator core, current mirrors of N-type transistors and current mirrors of P-type transistors, all of them can be seen in Figure 6. The transistors were interdigitated in two sections for both differential amplifiers in order to avoid crosstalk due to the high frequency signals.

Postlayout simulations:

The results, including the DNL and INL errors obtained from the postlayout netlist for the first 32 codes are presented in Figure 11. These simulations were done over three process corners (tt, ss, ff) and three temperatures (-40°, 60°, and 120°), the step current is set at 10μA (640μA for all 64 codes), and the dynamic code is established at 0.875 (8.75μA per code for fine control). The operation frequencies are set at 2.5GHz. The inputs for the simulation consist of sinusoidal signals with phases of 0°, 90°, 180°, and 270°. This implementation was built using UMC 65nm CMOS Technology, employing a Calibre view as layout extraction to validate the proposed methodology.

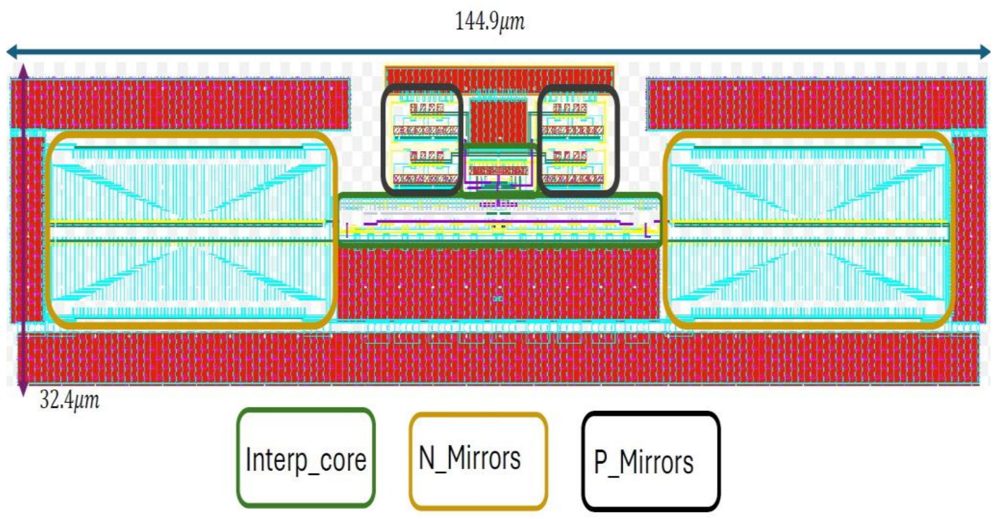

Figure 10: Phase Interpolator layout

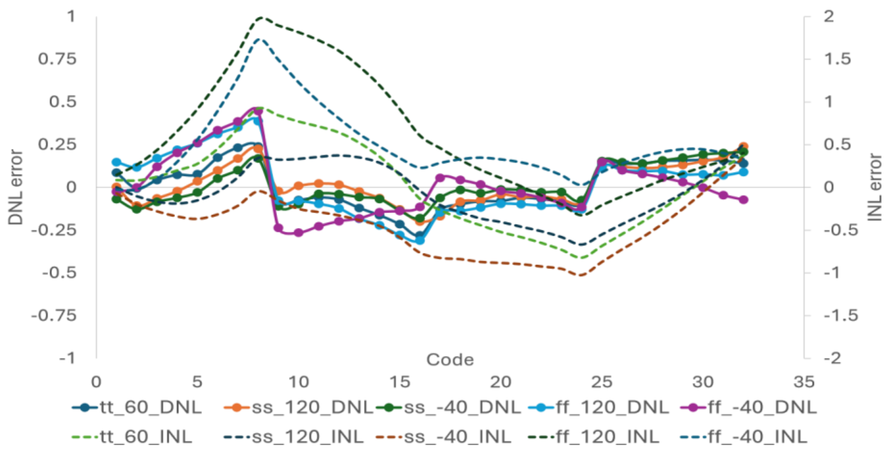

Figure 11: DNL and INL versus Code over five corners

Finally, the postlayout simulation in the typical corner, enabling the proposed modulation profile and with settings of 70uA as dynamic current (dynamic code of 0.875) and discrete current step of 4.25uA for DNL and INL error is shown in Figure 12.

As seen in Figure 12, the max $|DNL|$ is 0.192 LSB and the max $|INL|$ is 0.386 LSB, a reduction in comparison of the original proposal of 17.59% for DNL and 58.27% for INL. This proposal leaves open the possibility for further improvement by making the discrete current step variable instead of static through a calibration system. Table 2 summarizes the performance of this article and compares it with previously reported works.

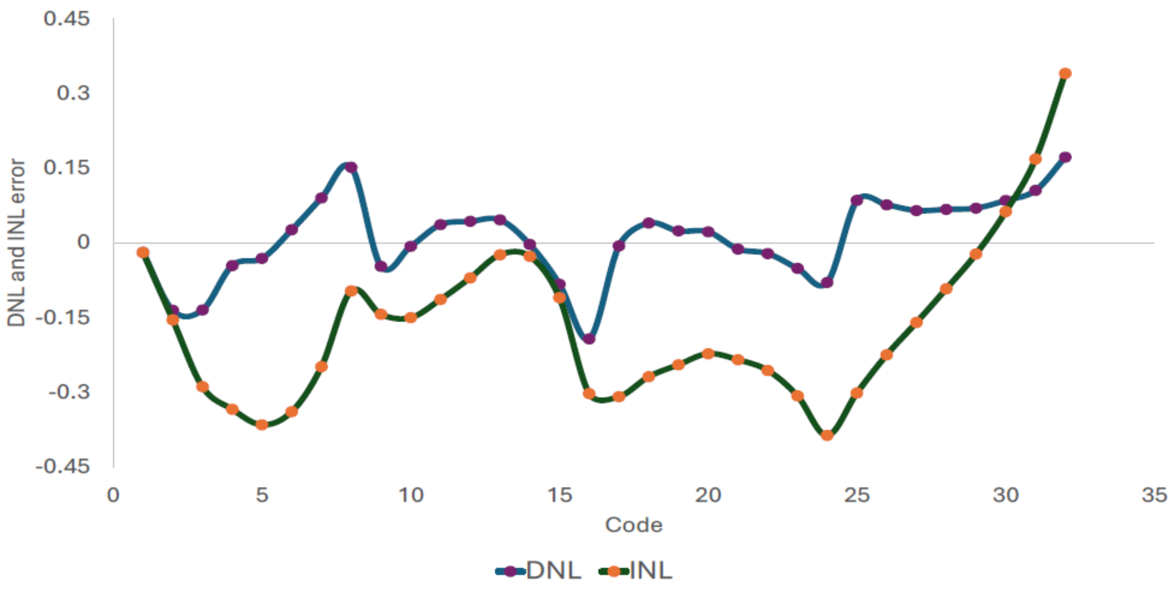

Figure 12: DNL and INL error using MP

Table 2: Comparison with other works

References	[17]	[1]	[5]	[18]	[3]	[19]	This work
Technology (nm)	65	65	65	65	65	65	65
Resolution (steps)	12	64	64	na	512	128	128
Frequency (GHz)	4	5	8	3	2.5	3.5	2.5

Power (*mW*)	na	15	13	2.3	10	7.58	1.6
Min step (*ps*)	20.83	3.12	1.95	na	0.781	2.23	3.12
DNL (LSB)	0.21	0.27	0.47	na	0.703	na	0.192
INL (LSB)	na	na	na	1.4	2.5	1.45	0.386

V. Conclusions

This paper presented a methodology to enhance the linearity of a phase interpolator using a Gilbert cell-based implementation in 65*nm* CMOS technology with a 1.2*V* supply voltage and an operating frequency of 2.5*GHz*. The circuit was characterized using a digital measurement system to evaluate the standard delay between consecutive codes, revealing DNL and INL errors of 23.3% and 92.5%, respectively, in the typical corner. The minimum phase step achieved was 3.12*ps*, with a resolution of 128 steps.

To further enhance phase linearity, a modulation profile scheme employing dynamic code adaptation was introduced. This approach significantly reduced DNL and INL errors to 19.2% and 38.6%, respectively, demonstrating its effectiveness in improving interpolator precision.

REFERENCES

[1] S. Hu, C. Jia, K. Huang, C. Zhang, X. Zheng, Z. Wang, A 10gbps cdr based on phase interpolator for source synchronous receiver in 65nm cmos, in: 2012 IEEE International Symposium on Circuits and Systems (ISCAS), IEEE, 2012, pp. 309 312.

[2] B. Razavi, Design of CMOS phase-locked loops: from circuit level to architecture level, Cambridge University Press, 2020.

[3] X. Zheng, C. Zhang, F. Lv, F. Zhao, S. Yuan, S. Yue, Z. Wang, F. Li, Z. Wang, H. Jiang, A 40-gb/s quarter-rate serdes transmitter and receiver chipset in 65-nm cmos, IEEE Journal of Solid-State Circuits 52 (11) (2017) 2963 2978.

[4] L. Rodoni, G. von Buren, A. Huber, M. Schmatz, H. Jackel, A 5.75 to 44 gb/s quarter rate cdr with data rate selection in 90 nm bulk cmos, IEEE Journal of Solid-State Circuits 44 (7) (2009) 1927 1941.

[5] G. Wu, D. Huang, J. Li, P. Gui, T. Liu, S. Guo, R. Wang, Y. Fan, S. Chakraborty, M. Morgan, A 1 16 gb/s all-digital clock and data recovery with a wideband high-linearity phase interpolator, IEEE Transactions on very large scale integration (VLSI) systems 24 (7) (2016) 2511 2520.

[6] G. Souliotis, A. Tsimpos, S. Vlassis, Phase interpolator based clock and data recovery with jitter optimization, IEEE Open Journal of Circuits and Systems (2023).

[7] M.-S. Chen, Y.-N. Shih, C.-L. Lin, H.-W. Hung, J. Lee, A fullyintegrated 40-gb/s transceiver in 65-nm cmos technology, IEEE Journal of solid-state circuits 47 (3) (2011) 627 640.

[8] X. Zheng, Design of high-speed serdes transceiver for chip-to-chip communications in cmos process., Ph.D. thesis, University of Lincoln (2018).

[9] R. Kreienkamp, U. Langmann, C. Zimmermann, T. Aoyama, H. Siedho , A 10-gb/s cmos clock and data recovery circuit with an analog phase interpolator, IEEE Journal of Solid-State Circuits 40 (3) (2005) 736 743.

[10] C.-Y. Chen, M. Q. Le, M. Wakayama, Phase-interpolator based pll frequency synthesizer, uS Patent 7,162,002 (Jan. 9 2007).

[11] S. Katare, S. V. Iyer, G. Tong, L. R. Munagala, M. Nagarajan, Y. Bangda, A low power high linearity phase interpolator design for high speed io interfaces, in: 2014 International SoC Design Conference (ISOCC), IEEE, 2014, pp. 92 93.

[12] Y.-H. Liu, C.-L. Li, T.-H. Lin, A 200-pj/b mux-based rf transmitter for implantable multichannel neural recording, IEEE Transactions on Microwave Theory and Techniques 57 (10) (2009) 2533 2541.

[13] M. Lin, Z. Wen, L. Chen, X. Li, Phase interpolation technique based on high-speed serdes chip cdr, in: 2015 5th International Conference on Computer Sciences and Automation Engineering (ICCSAE 2015), Atlantis Press, 2016, pp. 160 165.

[14] K.-A. Tran, A spread-spectrum clock generator using phase interpolation for emi reduction, Ph.D. thesis, Massachusetts Institute of Technology (2014).

[15] H. Hayati, M. Ehsanian, A 5-gbps cmos burst-mode cdr circuit with an analog phase interpolator for pons, Informacije MIDEM 45 (1) (2015) 39 46.

[16] A. AbdelHadi, M. Allam, S. Ibrahim, A high-linearity 6-ghz phase interpolator in 28-nm cmos technology, in: 2017 Japan-Africa Conference on Electronics, Communications and Computers (JAC-ECC), IEEE, 2017, pp. 41 44.

[17] B. Abiri, R. Shivnaraine, A. Sheikholeslami, H. Tamura, M. Kibune, A 1-to-6gb/s phase-interpolator-based burst-mode cdr in 65nm cmos, in: 2011 IEEE International Solid-State Circuits Conference, IEEE, 2011, pp. 154 156.

[18] A. T. Narayanan, M. Katsuragi, K. Kimura, S. Kondo, K. K. Tokgoz, K. Nakata, W. Deng, K. Okada, A. Matsuzawa, A fractional-n subsampling pll using a pipelined phase-interpolator with an fom of-250 db, IEEE Journal of Solid-State Circuits 51 (7) (2016) 1630 1640.

[19] Z. Wang, P. R. Kinget, A very high linearity twin phase interpolator with a low-noise and wideband delta quadrature dll for high-speed data link clocking, IEEE Journal of Solid-State Circuits 58 (4) (2022) 1172 1184.