High SFDR Pipeline ROM-less DDFS Design on FPGA Platform Using Parabolic Equations

Chua-Chin Wang, *Senior Member, IEEE*, Hsiang-Yu Shih, Wei Wang†, *Senior Member, IEEE*,

Department of Electrical Engineering
National Sun Yat-Sen University
Kaohsiung, Taiwan 80424
Email: ccwang@ee.nsysu.edu.tw

***Abstract*— A 4-stage pipeline ROM-less direct digital frequency synthesizer (DDFS) with equal division interpolation is proposed in this work. To attain higher SFDR (spurious free dynamic range) and faster clock rate, the hardware cost and delay using different segments with various interpolation equations are analyzed systematically to explore the optimal solution. The parabolic equations with proper selection of coefficients and factorized operation orders based on optimized hardware cost and delay are finally utilized to enhance SFDR. The proposed design is demonstrated by the physical implementation on Altera FPGA platform. The average SFDR is measured to be 68.4242 dBc with 1.1659 dBc deviation over 33 times of experiments. The measured SFDR is proved to outperform many previous DDFS works even if they were implemented on silicon.**

***Keywords* -ROM-less DDFS, spurious free dynamic range (SFDR) parabolic polynomial interpolation, pipeline structure, frequency synthesizer**

I. Introduction

Frequency synthesizers have been one of the most important roles in recent communication systems, mobile phones, medical bio-sensing devices, and global positioning system (GPS). They are mainly in charge of generating digital or analog signals with various frequencies. DDFS was firstly proposed in 1971's [1], where the amplitude data are stored in ROM-based look-up table. A generic DDFS structure is shown in Fig. 1. Since there is no need of the feedback loop and VCO (voltage-controlled oscillator), DDFSs are able to achieve fast frequency switching and the wide output range more easily than the PLL-based solutions. However, the bottleneck of DDFS implementation is the generation of a pure sinusoidal output. When the demand of high resoltuion of the sinusoidal output is required, the size of the ROM increases exponentially resulting in the major drawback of the ROM-based implementation architecture.

By contrast, many researches have also reported ROM-less DDFS designs, e.g., [9], [10], where different algorithms were used instead of ROM tables to realize the phase-to-amplitude converter (PAC). These ROM-less DDFS works demanded complicated polynomials to carry out the phase-to-sine mapper. However, any polynomial whose order is higher than three was found inefficient to be implemented if considering the feasibility of hardware realization [5]. Besides, the spurious free dynamic range (SFDR) of sine wave will be limited. Recently, several ROM-less DDFS [2], [3], [4], [6] has been developed to realize the PAC, i.e., the critical path of DDFS. However, these DDFSs based on 2nd-order polynomials still can not achieve high SFDR performance without sacrificing the speed.

To increase SFDR and reduce power dissipation, the 2nd order polynomials with equal division in a pipeline architecture is proposed in this design. Most important of all, different orders of mathmetical operators of the 2nd-order polynomial are fully analyzed to find out the optimal SFDR, output frequency and power consumption.

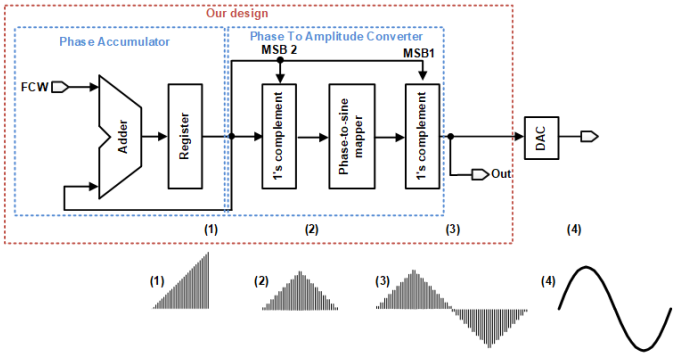

Fig. 1. Generic DDFS structure

II. DDFS Design Using Factorized Parabolic Equations

The sine wave is governed by the non-linear equation, $f(x) = sin(x)$. Theoretically, synthesizing a full sinusoidal wave ($0 \sim 2 \cdot \pi$) can be achieved by synthesizing one quarter of the full cycle, namely ($0 \sim \frac{1}{2} \cdot \pi$), and then bilateral sysmetry with respect to a vertical line and x-axis, as shown in Fig. 2, is used to generate the other 3 quaters. DDFS means to take advantage of digital logic and algorithms to generate a wave

Prof. C.-C. Wang is the contact author. He is with Department of Electrical Engineering, National Sun Yat-Sen University, Kaohsiung, Taiwan 80424. (e-mail: ccwang@ee.nsysu.edu.tw).

†Prof. W. Wang is with College of Electronics Engineering, Chongqing University of Posts and Telecommunications (CQUPT), 400065, Chongqing, P. R. China.

978-1-5386-4441-6/18/$31.00 ©2018 IEEE

form close to the real sinusoidal wave. Thus, the interpolation approach using low-order polynomials (or called equations) is widely adopted to speed up the sine wave generation.

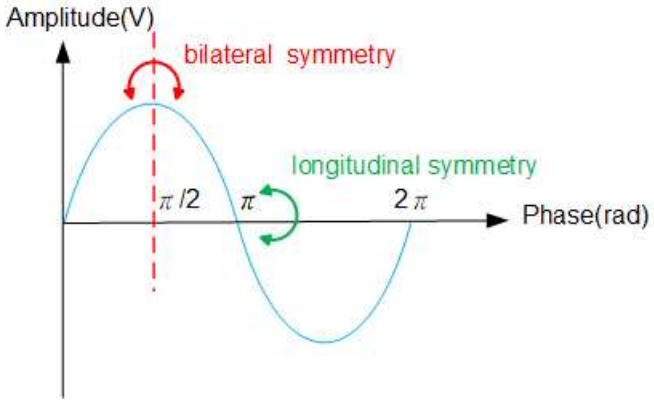

Fig. 2. Symmetry of sine wave

A. Interpolation schemes for DDFS

The selection of interpolation schemes will directly affect error, speed, and particularly SFDR. Three common interpolation schemes using low-order polynomials are linear interpolation (1st-order polynomials), quasi-linear interpolation (combination of 1st-order and 2nd-order polynomials), and parabolic interpolation (2nd-order polynomials), [7], [8]. The linear interpolation apparently has the edge of simplicty with poor accuracy. By contrast, the parabolic interpolation taking advantage of the similarity between $sin(x)$ and the parabolic function. However, the price to pay is high computation complexity. As for the quasi-linear interpolation, the synthesis of the sine wave close to the origin is based on linear polynomials, and that of the region close to $\frac{1}{2}\pi$ is based on the parabolic polynomials. That is, the quasi-linear interpolation manages to fully take advantage of 1st and 2nd polynomials to attain the minimal error. However, this scheme needs more sophisticated logic control and region division considerations.

Besides the selection of different interpolation schemes, how the entire region to be synthesized is segmented is another issue to be resolved. Theoretically, more segments are utilized and synthesized will result in higher accuracy. However, the design complexity will increase, and the speed will be reduced. For instance, will the 8-segment division of $0 \sim \frac{1}{2}\pi$ have a better SFDR than 4-segment? Or the SFDR performance of equal segements better than that of unequal segments? How to partition the entire $0 \sim \frac{1}{2}\pi$ into unequal segments so that a better SFDR will be achieved? All of these puzzles are expected to be answered substantially.

To explore the scenarios given different interpolation approaches and different segmentations, SFDR and error simulations by MATLAB are summarized in Table I. The SFDR and maximum error results of parabolic interpolation with 8 segments is found to be better than that of quasi-linear interpolation with 32 segments. Besides, if the design complexity, area overhead, and switching speed are taken into consideration, the parabolic interpolation with 8 segments seems to be a better option.

TABLE I

PERFORMANCE GIVEN DIFFERENT SEGMENTATION AND INTERPOLATION

	Parabolic interpolation		Quasi-linear interpolation		Linear interpolation	
Segment	Max. Error	SFDR	Max. Error	SFDR	Max. Error	SFDR
4	4.97×10^{-4}	86	1.28×10^{-3}	71	6.29×10^{-3}	53
8	6.30×10^{-5}	**106**	4.67×10^{-4}	81	1.59×10^{-3}	65
16	7.92×10^{-6}	123	1.35×10^{-4}	93	4.01×10^{-4}	78
32	9.97×10^{-7}	142	3.61×10^{-4}	105	1.00×10^{-4}	90

When it comes to the performance priority, SFDR is well recognized to be more important than the maximum error such that the parabolic interpolation method is apparently better than the other two methods. We then explore the error distribution of different segmentations of the parabolic interpolation method by MATLAB simulations. Fig. 3 demonstrates the error distribution of 4 different segmentation scenarios. Notably, although 32-segment partition has the minimal error distribution, 8-segment parition with over 100 dBc SFDR is much more easy to be realized on silicon. Thus, the parabolic interpolation with 8-segment partition is chosen to be realized in this investigation.

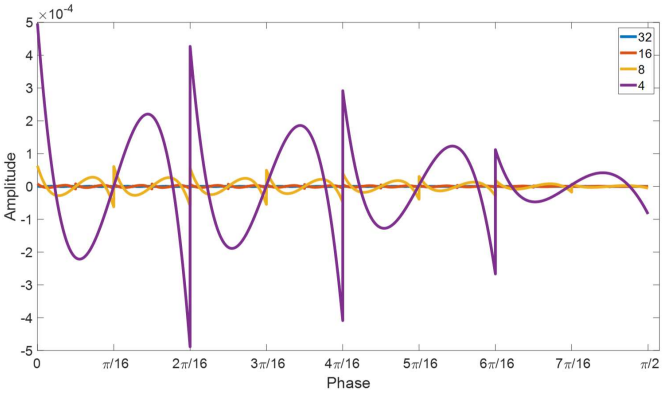

Fig. 3. Error distribution of different segmentations

1) Parabolic polynomial interpolation derivation: A quadrant of sinusoid is partitioned into i segments, where every segment is approximated by the 2nd-order parabolic equation, and each segment has m sampling points, as shown in Eqn. (1).

$$y_{ij}(x_{ij}) = (a_i x_{ij} + b_i)\, x_{ij} + c_i, \quad i = 1 \sim 8, \quad j = 1 \sim m \quad (1)$$

According to the calculation method of [5], the least square method is used to attain the coefficients, we differentiate Eq. (1) to get Eqn. (2) such that the optimal "a_i" and "b_i" will be found.

$$y'_{ij}(x) = 2a_i x_{ij} + b_i, \quad i = 1 \sim 8, \quad j = 1 \sim m \qquad (2)$$

2) Selection of pipelining stages: Pipelining is a well-known method to enhance clock rate and throughput. However, how many stages of pipelining is needed in the 8-segment DDFS design using 2nd-order parabolic polynomial, namely Eqn. (1), is the next issue to be resovled. Although higher order of pipelining is feasible, the increase of clock rate becomes very obscure and the hardware cost turns into quite significant. Therefore, a 4-stage pipeline structure is selected to carry out the proposed DDFS design as shown in Fig. 4

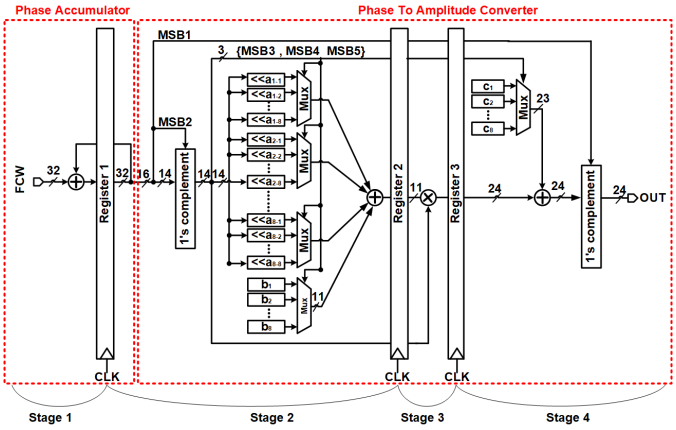

Fig. 4. Diagram of 4-stage pipeline DDFS

3) SFDR simulation by MATLAB: The above 4-stage DDFS design is firstly simulated by MATLAB using the toolbox therewith to attain the sinusoidal outcome as shown in Fig. 5, where the SFDR is also found to be 74 dBc.

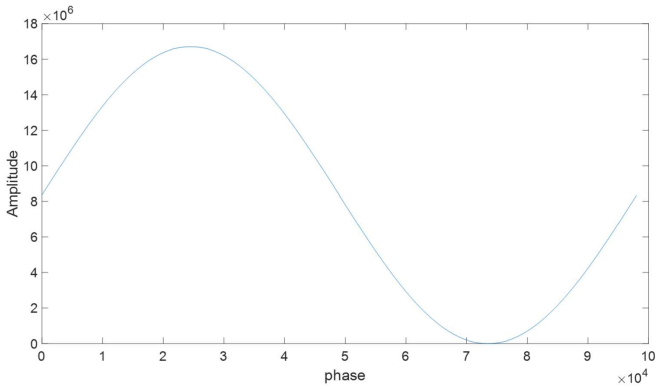

Fig. 5. Synthesized sine wave genearted by MATLAB

III. Verification by FPGA Emulation

1) FPGA emulation and experiment: The proposed 4-stage pipeline DDFS using parabolic equations is coded using Verilog and downloaded to Altera DE2 FPGA platform, which is product no.: EP2C35672C6, where a 10-bit DAC (digital to analog converter, ADV7123) is included. The experimental site is shown as Fig. 6, where LeCroy WaveRunner 610Zi is used as an OSC to display the measurement results.

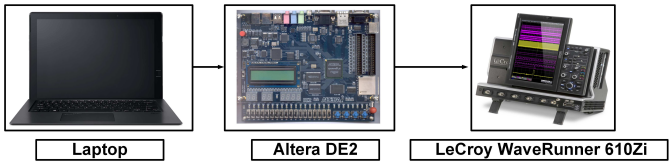

Fig. 6. FPGA experimental setup

Fig. 7 is the measurement display on the OSC, where the power performance is displayed by dBm. If it is converted into dBc, the readings must be doubled by the basics definitions in Eqn. (3) and (4). Hence, the dBc is normalized to be 68, since the reading of dBm is 34. The experiment has been carried out 33 times to attain a meaningful statistic result. The overall histogram is shown in Fig. 8, where the average is 68.4242 dBc, and all of the measured readings are distributed within $\leq 3\sigma$, where σ = 1.1659 dBc, which justify the credibility and repeatability of the measurement.

$$\mathrm{dBm} = 10 \cdot \log(\frac{\mathrm{P}}{1\mathrm{mW}}) \tag{3}$$

$$2 \cdot \mathrm{dBm} = \mathrm{dBc} \rightarrow 34\ \mathrm{dBm} = 68\ \mathrm{dBc} \tag{4}$$

The reason why the measured SFDR lower than that derived by MATLAB is resulted from the imperfection of DAC, which is the 10-bit DAC (ADV7123) inlcuded in the FPGA. The INL (integral nonlinearity) disclosed in ADV7123 is given in Fig. 9, which is as high as 0.75 LSB.

2) Performance comparison: Referring to Table II, the proposed design is compared with several recent DDFS reports. Our design is the only one not implemented on a single silicon. By contrast, it is the only one measured by FPGA emulations. However, the SFDR attained by our design outperforms the rest of DDFS designs. This comparision proves that the effectiveness of the proposed design method.

TABLE II
SFDR Performance Comparison

	[2]	[4]	[12]	[13]	this work
year	2013	2016	2017	2017	2018
process μm CMOS	0.13	0.18	0.065	0.18	FPGA
FCW (bits)	32	17	24	16	32
O/P resolution (bits)	12	10	10	16	24
clock rate (MHz)	650	100	2000	1000	100
verification	Meas.	Meas.	Meas.	Simu.	Meas.
SFDR (dBc)	60	52.47	56.5	49.1	68.4242

IV. Conclusion

According to the measurement by FPGA emulation and the comparison with recent DDFS designs realized by expensive CMOS processes, our design demonstrates the superior DDFS performance provided that the DAC has a subpar INL. This outcome justifies that the proposed factorized equations and pipeline structure are very efffective in the implementation of high-SFDR DDFS designs.

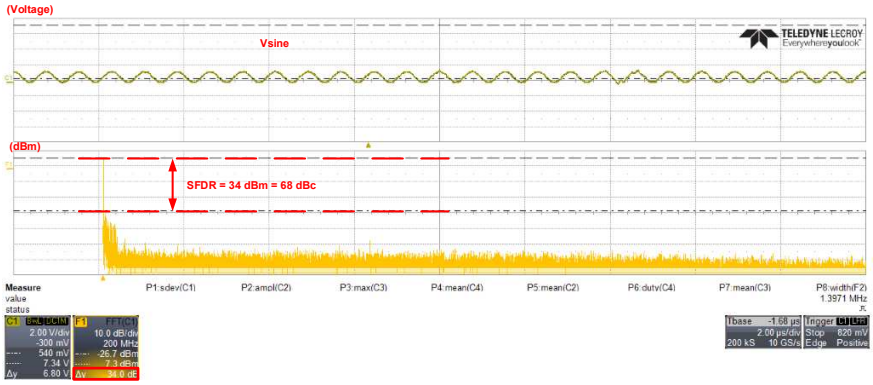

Fig. 7. SFDR measurement on FPGA platform by spectrum analysis

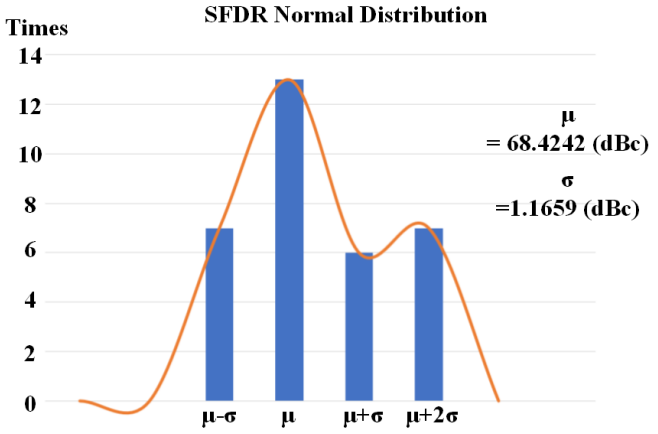

Fig. 8. Histogram of SFDR measurement

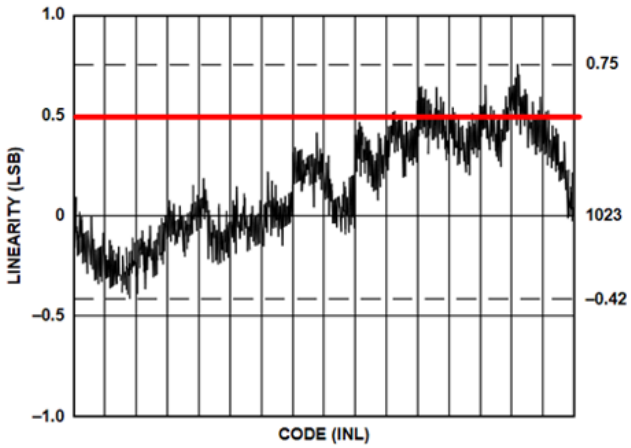

Fig. 9. DAC INL

ACKNOWLEDGEMENT

This investigation is partially supported by Ministry of Science and Technology, Taiwan, under grant MOST 107-2218-E-110-004-. The authors would like to express their deepest gratefulness to Chip Implementation Center of National Applied Research Laboratories, Taiwan, for their thoughtful chip fabrication service and EDA tool support.

REFERENCES

[1] J. Tierney, C. Rader, and B. Gold, "A digital frequency synthesizer," *IEEE Trans. on Audio and Electroacoustics*, vol. 19, no. 1, pp. 48-57, Mar. 1971.

[2] J. Cali, X. Geng, F. Zhao, M. Pukish, F. Dai, and A. Aklian, "A 650 MHz DDFS for stretch processing radar in 130nm BiCMOS process," in *Proc. European Microwave Integrated Circuit Conference (EMICC)*, pp. 33-36, Oct. 2013.

[3] M. Padash, S. Toofan, and M. Yargholi, "A 9-bit, 1-giga samples per second sine and cosine direct digital frequency synthesizer," in *Proc. Iranian Conf. on Electrical Engineering (ICEE)*, pp. 438-442, May 2014.

[4] P. R. B. de Carvalho, J. A. A. Palacio, and W. Van Noije, "Area optimized CORDIC-based numerically controlled oscillator for electrical bio-impedance spectroscopy," in*Proc. IEEE Int. Freq. Control Symposium (IFCS)*, pp. 1-6, May 2016.

[5] C.-C. Wang, C.-H. Hsu, C.-C. Lee, and J.-M. Huang, "A ROM-less DDFS based on a parabolic polynomial interpolation method with an offset," *Journal of Signal Processing Systems*, vol. 61, pp.1-9, May 2010.

[6] C.-H. Hsu, Y.-C. Chen, and C.-C. Wang, "ROM-less DDFS using non-equal division parabolic polynomial interpolation method," in *Proc. Int. Symp. on Integrated Circuits (ISIC)*, pp. 59-62, Dec. 2011.

[7] X. Geng, F. F. Dai, J. D. Irwin, and R. C. Jaeger, "An 11-Bit 8.6 GHz direct digital synthesizer MMIC with 10-Bit segmented sine-weighted DAC," *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 45, no. 2, pp. 300-313, Feb. 2010.

[8] C.-C. Wang, J.-M. Huang, Y.-L. Tseng, W.-J. Lin, and R. Hu, "Phase-adjustable pipelining ROM-less direct digital frequency synthesizer with a 41.66-MHz output frequency," *IEEE Trans. Circuits Syst.- II Exp. Briefs*, vol. 53, no. 10, pp. 1143-1147, Oct. 2006.

[9] B. Pontikakis, H.-T. Bui, F-R. Boyer, Y. Savaria, "Precise free-running period synthesizer (FRPS) with process and temperature compensation," *IEEE MidWest Symposium on Circuits and Systems 2007 (MWSCAS '07),* pp. 1118-1121, Aug. 2007.

[10] P. Sotiriadis, "Timing and spectral properties of the Flying Adder frequency synthesizers," *IEEE International Frequency Control Symposium 2009,* pp. 788-792, July 2009.

[11] Y. Song, and B. Kim, "A 250 MHz Direct Digital Frequency Synthesizer with Delta-Sigma Noise Shaping," *2003 IEEE Int. Solid-State Circuits Conf. (ISSCC 2003),* vol. 1, pp. 472-509, Feb. 2003.

[12] A. M. Alonso and X. Yuan and M. Miyahara and A. Matsuzawa, "A 2 GS/s 118 mW digital-mapping direct digital frequency synthesizer in 65nm CMOS," *12th European Microwave Integrated Circuits Conference (ISSCC 2003),* pp. 228-231, Oct. 2017.

[13] R. Suryavanshi and S. Sridevi and B. Amrutur, "A comparative study of direct digital frequency synthesizer architectures in 180nm CMOS," *International conference on Microelectronic Devices, Circuits and Systems. (ICMDCS 2017),* pp. 1-5, Aug . 2017.