International Journal of
Applied Mathematics,
Electronics and Computers
IJAMEC
ISSN: 2147-8228 http://ijamec.atscience.org
Advanced Technology and Science
Original Research Paper

Direct Digital Frequency Synthesizer Designs in MATLAB

Yunus Emre ACAR *[1], Ercan YALDIZ [2]

Accepted 3rd September 2016

Abstract: *This study presents the structure of the Direct Digital Frequency Synthesizers (DDFSs) which have several advantages compared to conventional synthesizers such as high frequency, fast switching speed and low power dissipations. In order to lessen the physical area and power dissipation, ROM compression techniques are applied in designs. Bipartite Table Method (BTM) and Multipartite Table Method (MTM) are utilized in this study because of the fact that they provide high compression rates. By using MTM, the compression rates of 157.54:1, 726.71:1 and 3463.29:1 are obtained at 58.40 dB, 75.30 dB and 84.66 dB SFDR levels, respectively.*

Keywords: *DDFS, CORDIC, MTM, BTM.*

1. Introduction

Direct Digital Frequency Synthesis is one of the most popular techniques to synthesize frequency for the systems requiring specific frequencies, fast switching, low power dissipation and small silicon area. There are several methods using table based or iterative approaches to implement Direct Digital Frequency Synthesizers (DDFSs). In addition to natural advantages of the approaches, ROM compression techniques are applied to reduce the ROM size. Furthermore, the approaches are enhanced with some modifications offering trade-off between Spurious Free Dynamic Range (SFDR), switching speed and the used silicon area.

Direct Digital Frequency Synthesizers (DDFSs) are commonly used in several areas such as defense industry, satellite systems, radars, test and measuring equipments, etc. As distinct from analog or indirect synthesizers, DDFSs provide high frequency resolution, fast switching speed, continuous phase switching, small physical area and low power dissipation [1]. For the last 10 years, works on the frequency synthesizers are focused on minimizing the used area and power dissipation while keeping the spectral purity above an acceptable level [2], [3] and [4]. High frequency synthesizers are also studied with some recently offered approaches [5], [6] and [7].

A DDFS consists of three sub blocks which are phase accumulator, phase to amplitude converter and the digital to analog converter. Figure 1 illustrates the principle parts of a DDFS.

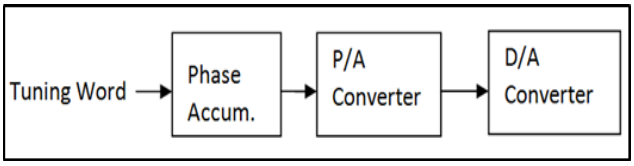
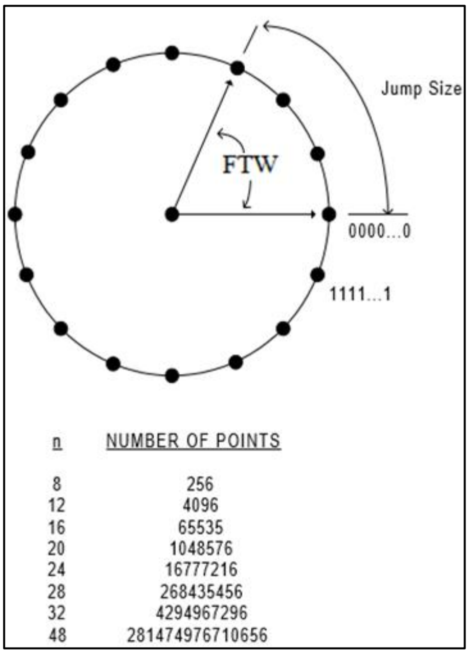


Figure 1. Principal parts of DDFS

1.1. Phase Accumulator

Phase accumulator is used in DDFS for an adjustable frequency output. It is controlled by an N bit Frequency Tuning Word (FTW). Several frequencies can be obtained by using this digital control word with one clock source.

Phase accumulator works as an N bit counter and a digital phase wheel is created. Each of the 2^Npoints on this wheel correspond the amplitude value of the related phase. The increment of the counter is determined by the FTW. Figure 2 illustrates the digital phase wheel idea.



n	NUMBER OF POINTS
8	256
12	4096
16	65535
20	1048576
24	16777216
28	268435456
32	4294967296
48	281474976710656

Figure 2. Digital Phase Wheel [1]

The frequency of the output signal is

$$f_{out} = FTW \times \frac{f_{clk}}{2^N} \qquad (1.1)$$

1.2. Phase to Amplitude Converter (PAC)

There are two main approaches for phase to amplitude conversion in a DDFS. The corresponding amplitude value of the related phase can be obtained by using Look Up Tables (LUTs) or iterative calculations. While table based methods allow operating at higher frequencies, iterative methods provides better spectral purity. Many studies have done about both methods for the last decade [2], [9], [11] and [12]. The works about table based methods have generally aimed to reduce the ROM size and to

[1] *Department of Electrical and Electronics, Technology Faculty, Selçuk University, Campus, 42031, Konya/Turkey*
* *Corresponding Author: Email: yacar@selcuk.edu.tr*

Note: This paper has been presented at the 3rd International Conference on Advanced Technology & Sciences (ICAT'16) held in Konya (Turkey), September 01-03, 2016.

increase the Spurious Free Dynamic Range (SFDR) which shows the spectral purity of the generated sinusoid. On the other hand, COordniate Rotational Digital Computer (CORDIC) based iterative methods have been proposed to achieve better SFDR levels.

1.3. Digital to Analog Converter (DAC)

DACs are used to convert the digital data taken from the PACs to analog signals. DAC part is one of the most important parts of the DDFS and directly related to its performance.

The resolution and sampling frequency of DAC are able to determine the limits of DDFS. While delta-sigma and R-2R type DACs are beneficial when high resolution is required, current steering type DACs have sampling rates up to GSPSs with comparatively lower resolution. The lower resolution comes with higher quantizing errors and this cause a decrement in SFDR but oversampling may lessen the decrement a little. Addition to all these, the AC and DC characteristics of the DAC are also important and need to be considered at the implementation stage [8].

2. Methods

2.1. CORDIC Based DDFSs

CORDIC is a structure that was proposed by Volder to calculate basic trigonometric functions in 1959 [10]. In CORDIC based algorithms, two-dimension vector rotation idea is used. The vector rotation idea is shown in Figure 3, and the related trigonometric equations are

$$x_o = x_i\, cos\theta - y_i\, sin\theta \tag{2.1}$$
$$y_o = x_i\, sin\theta + y_i cos\theta \tag{2.2}$$

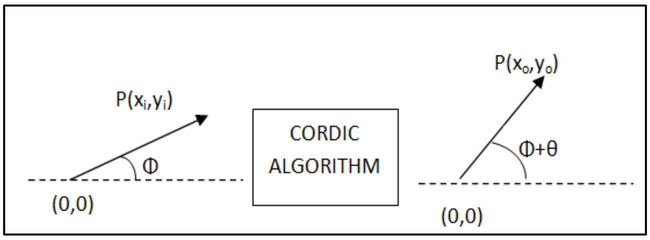


Figure 3. CORDIC Algorithm

In CORDIC based DDFS designs, Eq. (2.1) and (2.2) are reformed as Eq. (2.3) where G is the gain constant.

$$\begin{bmatrix} x_o \\ y_o \end{bmatrix} = G \begin{bmatrix} 1 & -tan\,\theta \\ tan\,\theta & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \tag{2.3}$$

As mentioned earlier, CORDIC is applied as an iterative method to calculate some trigonometric functions. Angle (θ) of rotation is completed after T sub rotations and evaluated as in Eq. (2.4) where δ_t is the direction of the rotation.

$$\theta = \delta_0\, \theta_0 + \delta_1 \theta_1 + \cdots + \delta_t \theta_t \quad \delta_k \,\epsilon\, \{-1{,}0{,}1\} \tag{2.4}$$

$\tan\theta$ is chosen as multiple of 2^{-1} so that (2.3) is easily applied digitally. The equation can be rearranged as

$$x_{i+1} = x_i - y_i\, \delta_i\, 2^{-i} \tag{2.5}$$
$$y_{i+1} = y_i - x_i\, \delta_i\, 2^{-i} \tag{2.6}$$

The gain constant is generally applied as the initial end point. P(G, 0) is used instead of P(0,0) as the initial end point of the vector where $G = \prod_T \frac{\mathbf{1}}{\sqrt{\mathbf{1+2^{-2i}}}}$ [11].

2.2. Table Based DDFSs

Table based methods originally have a simpler idea. ROMs/RAMs are used as LUTs. Phase accumulator is utilized as the address counter of the ROMs.

The ROM size determines the resolution of the DDFS. The bigger the ROM size is, the better the resolution is. This was the bottleneck of the traditional table based methods. The bigger ROM size causes much more power dissipation and needs more physical area. ROM compression techniques have been enhanced to overcome this problem. Quadrant compression, Sunderland and Hutchinson methods, Bipartite Table Method (BTM) and Multipartite Table Method (MTM) are some of the most common ROM compression techniques.

Quadrant compression technique uses the symmetric structure of the sinusoid. Instead of a LUT that stores the sine values between$0 - 2\pi$, a LUT that stores the sine values between $0 - \pi/2$ is used. The most significant two bits of the P bit phase word determine the quadrant of the phase wheel and the rest include the phase information. The block diagram of the technique is given in Figure 4.

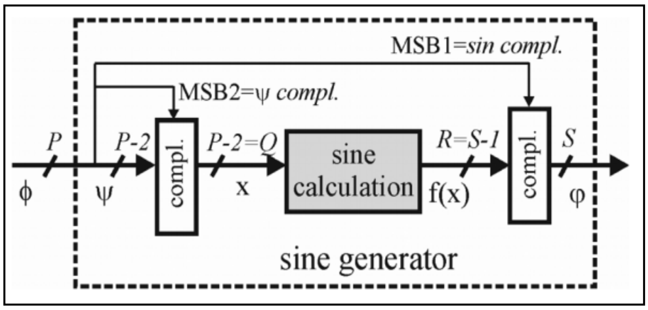


Figure 4. Quadrant Compression Technique [9]

BTM was proposed by Dinechin and Tisserand in 2005 [4]. The technique depends on partial piecewise linear approach. The x axis is divided into 2^a equal pieces and f(x) is interpolated separately for each. The equations of the slope and approximated f(x) are

$$m_i = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \tag{2.7}$$
$$f_{app}(x) = y_i + m_i\,(x - x_i) \tag{2.8}$$

As distinct from the piecewise linear approach, x axis is divided into 2^b larger intervals (b < a) and same slope value used for the adjacent pieces in larger intervals. Thus, the ROM size is efficiently decreased. Figure 5 shows the piecewise linear approach used in BTM.

In BTM, there are two tables to store the values required for the interpolation. The y_i values are stored in Table of Initial Values (TIV) with α bits and Table of Offsets (TO) stores information for m_i and $(x - x_i)$ part of Eq. (2.8) with β and γ bits separately The ROM is addressed by $2^\alpha + 2^{\beta+\gamma}$ bits [9].

The linearity of $m_i \times (x - x_i)$ provides opportunity to make the ROM size smaller. By using this linearity property (2.10), it can be beneficial to use more than one TO for some SFDR levels. When the number of TOs is more than one, the technique is called Multipartite Table Method (MTM).

$$m_i \times (x - x_i) = m_i x_1 + m_i x_2 + \cdots + m_i x_k \tag{2.9}$$

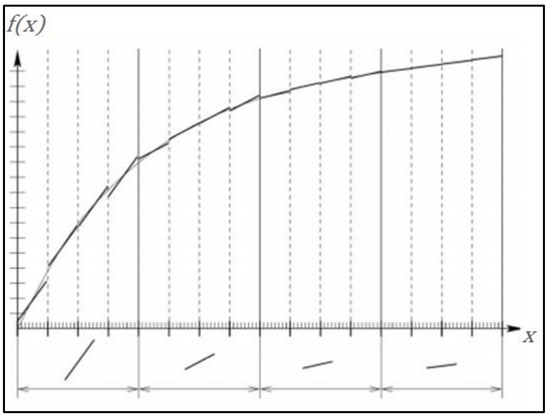


Figure 5. Bipartite Table Method (BTM) [4]

The study of De Caro and his friends shows that the number of TOs is not proportional to SFDR. The study also indicates that SFDR of the DDFS strongly depends on x, y, z numbers. From the point of this view, it is obvious that an optimization is essential to get best results. If the number of TOs is more than two then the optimization requires more complex algorithms and calculations. Moreover, the higher number of TOs does not ensure smaller physical area in compression with two or three TOs for the SFDR levels less than 90 dBc [9]. Thus, this paper is focused on the table based methods with one or two TOs.

3. MATLAB Results

There are three main objectives while designing a DDFS. These are maximizing the SFDR, minimizing the ROM size and increasing the maximum operating frequency as much as possible. In this paper, the SFDR and ROM size considerations are investigated.

3.1.BTM Design

As mentioned earlier, BTM uses two different tables to store the sine amplitude information. While one of them is for initial values, the other one is for the offset values. The size of TIV is calculated as

$$TIV_{size} = R \times 2^{\alpha} \tag{3.1}$$

R is the amplitude resolution and α is the number of bits that represents TIV values. TO values are represented with $\beta + \gamma$ bits and the size is calculated as

$$TO_{size} = (R - \alpha) \times 2^{\beta+\gamma-1} \tag{3.2}$$

The sum of $\mathrm{TO_{size}}$ and $\mathrm{TIV_{size}}$ gives the total ROM size. The compression ratio is evaluated as

$$compression\ rate = \frac{R \times 2^{(\alpha+\beta+\gamma)}}{TIV_{size}+TO_{size}} \tag{3.3}$$

The P phase information taken from the accumulator includes α and γ bits. The ROM size depends on these bits with some β bits in α bits. Figure 6 shows the decomposition of the phase word.

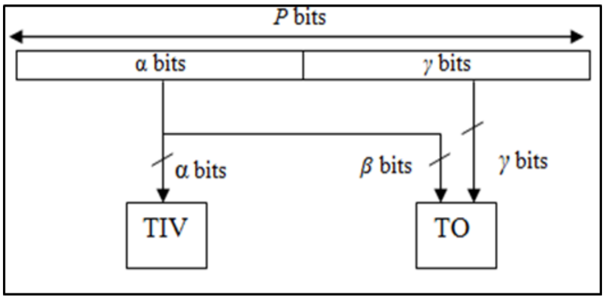


Figure 6. Phase Word Decomposition in BTM Design

Table 3.1.BTM Results in MATLAB

α	$\beta + \gamma$	Resolution (bits)	SFDR (dB)	ROM size (bits)
5	2	8	56.22	352
5	3	8	58.14	448
5	4	8	56.25	640
5	2	10	70.45	960
5	3	10	69.93	1600
5	4	10	71.80	2880
5	2	12	78.76	3968
5	3	12	82.68	7552
5	4	12	80.88	14720

The BTM design results are given in Table 3.1 and allow two significant deductions. Firstly, it is obvious that increasing the length of the phase word does not improve the SFDR and it makes the ROM size bigger. So, determining the best decomposition of the phase word has a great importance. Secondly, increasing the amplitude resolution provides better SFDR but causes a notable increment in the ROM size.

3.2.MTM Design

MTM is originally based on BTM and requires smaller ROM size comparing to it with a negligible decrease in SFDR. It is generally preferred when higher SFDR is required because BTM provides the same SFDR with a very high ROM size. In this part of the paper, some MTM design results are investigated and are compared with BTM.

The TIV size calculation in MTM is similar to BTM. Using more than one TO makes the difference in total ROM size. The phase word decomposition is given in Figure 7 where $\theta_i = \beta_i + \gamma_i$.

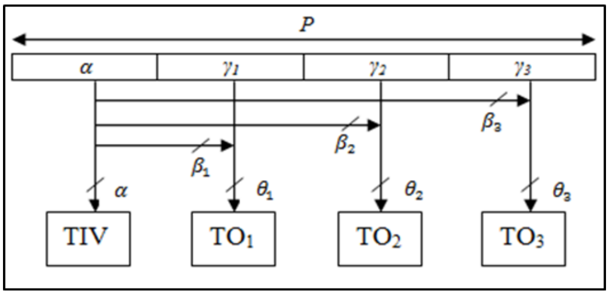


Figure 7. Phase Word Decomposition in MTM Design

Total ROM size in MTM is calculated as

$$Total_{size} = TIV_{size} + \sum_i^n TO_{i_{size}} \tag{3.4}$$

where n is the number of TOs and (3.5) is used to evaluate $TO_{i_{size}}$.

$$TO_i = \left[\left(R - \sum_{k=0}^{i} \gamma_k\right) \times 2^{\beta_i+\gamma_{i-1}-1}\right] \tag{3.5}$$

The phase word length and the amplitude resolution determine SFDR and ROM size. Some design results are given in Table 3.2.

As clearly seen from the table it is possible to reach better SFDR by just rearranging the decomposition of the phase word. The results indicate that minimizing α value allows making total ROM size smaller. It is also clear that increasing P and R values causes a remarkable rise in SFDR.

While closer α, βi and βi+1 values provide the best compression, it is more complicated to offer any condition for the best SFDR. Although there are some algorithms proposed to find out the best decomposition of the phase word for a specific SFDR level, there is not any multi-objective algorithm to optimize both the SFDR and the ROM size. In this paper, any algorithm is not used and the results given in tables show the effect of the parameters.

Table 3.2. 2 TOs MTM Desıgn Results ın MATLAB

	α	γ_1	β_1	γ_2	β_2	SFDR (dB)	ROM size (bits)
P=10 R=8	5	3	2	2	2	57.09	304
	4	4	2	2	2	55.96	256
	4	3	2	3	2	55.32	208
	4	3	3	3	2	59.26	272
	4	3	4	3	2	57.02	400
	4	3	2	3	4	57.09	256
	3	3	2	4	4	55.64	400
P=13 R=11	7	3	3	3	3	77.18	1568
	7	4	3	2	3	80.34	1664
	6	4	3	3	3	81.98	1056
	5	5	3	3	3	75.20	1152
	5	4	3	4	2	78.25	864
	5	4	5	4	2	73.20	1952
	4	4	5	5	2	69.04	2160

Table 3.3 shows some design results with different P and R values at SFDR levels around 57, 75 and 84 dB. It demonstrates that increasing the number of TOs is more beneficial at higher SFDR levels. Although the table shows that increasing the number of TOs in all SFDR levels gives better results in ROM size, it is not practical. De Caro and his friends show that the more TO is the better ROM compression ratio but not the smaller required area for DDFS. Table III also indicates that it is not possible to get a remarkable increase in SFDR without increasing P and R values.

Table 3.3. MATLAB Desıgn Results For Dıfferent Number of TOs

	Techniques	SFDR (dB)	ROM size (bits)	Comp. Ratio

P=10
R=8
	BTM	57.32	1040	31.51:1

	MTM (2 TOs)	57.09	304	107.79:1
	MTM (3 TOs)	58.40	208	157.54:1

P=13
R=11
	BTM	75.31	3008	119.83:1

	MTM (2 TOs)	75.20	928	388.41:1
	MTM (3 TOs)	75.30	496	726.71:1

P=16
R=13
	BTM	84.24	16384	208:1

	MTM (2 TOs)	84.50	1984	1717.68:1
	MTM (3 TOs)	84.66	984	3463.29:1

4. Conclusions

The DDFS structure has been investigated in this study. The phase accumulator, phase to amplitude converter and digital to analog converter blocks of the structure has been mentioned in section I. Later on, CORDIC, BTM and MTM based DDFS have been explained in section II. In section III, table based methods have been focused and some MATLAB design results have been interpreted. BTM designs are revealed with two different R values. 59.26 dB SFDR level is achieved with R=8 and 272 bits ROM size. Increasing the resolution to 11 bits, 81.98 dB SFDR level is reached with 1056 bits. 157.54:1, 726.71:1 and 3463.29:1 compression rates are obtained at 58.40 dB, 75.30 dB and 84.66 dB SFDR levels respectively by increasing the number of TOs.

The main problem of the table based methods is that they need much more physical area. When advanced ROM compression techniques such as MTM are applied to minimize the area, a complex optimization algorithm is required at higher SFDR levels. There are some beneficial optimization algorithms to minimize the ROM size at some target SFDR levels but multi-objective optimization techniques may be considered as a future work to optimize the sampling frequency, ROM size and SFDR.

References

[1] Kester W. A Technical Tutorial on Digital Signal Synthesis, Analog Devices, Tech. tut. , 1999.

[2] Strollo A. G. M., De Caro D. and Petra N. A 630 MHz, 76 mW Direct Digital Frequency Synthesizer Using Enhanced ROM Compression Technique, *Ieee Journal Of Solid-State Circuits*, vol.42, Feb. 2007, pp. 350–360.

[3] Jafari H., Ayatollahi A., and Mirzakuchaki S. A low power, high SFDR, ROM-less direct digital frequency synthesizer, *IEEE Conference on Electron Devices and Solid-State Circuits*, 2005, pp. 829-832.

[4] Dinechin F. and Tisserand A. Multipartite Table Methods, *IEEE Transactions On Computers*, vol.54, Mar. 2005, pp. 319-330.

[5] Yuan L., Zhang Q. and Shi Y. A 2 GHz Direct Digital Frequency Synthesizer based on multi-channel structure, *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015, pp.3064 – 3067.

[6] Chen Y. H. and Chau Y. A. A Direct Digital Frequency Synthesizer Based on a New Form of Polynomial Approximations, *IEEE Transactions on Consumer Electronics*, vol.56, May. 2010, pp. 436-440.

[7] Asok K. S. and Sahoo K. P. Digital Hardware Optimization for 1.5-GHz High-Speed DDFS, *Electronics, Circuits and Systems (ICECS), 2014 21st IEEE International Conference*, 2014, pp. 746 – 749.

[8] Kılıç A. Implementation of A Digital Signal Synthesizer With High Spurious Free Dynamic Range, M. Eng. thesis, METU Natural And Applied Sciences, Turkey, 2006.

[9] De Caro D., Petra N., and Strollo A. G. M. Reducing Lookup-Table Size in Direct Digital Frequency Synthesizers Using Optimized Multipartite Table Method, *IEEE Transactions On Circuits And Systems—I*, vol.55, Aug. 2008, pp. 2116-2127.

[10] Volder J. E. The CORDIC trigonometric computing technique, *IRE Transactions on Electronic Computers*, vol.8 , Sept. 1959, pp. 330–334.

[11] Menakadevi T. and Madheswaran M. Direct Digital Synthesizer using Pipelined CORDIC Algorithm for Software Defined Radio, *International Journal of Science and Technology*, vol. 2, June 2012, pp.372-378.

[12] Shuqin W., Yiding H., Kaihong Z. and Zongguang Y. A 200 MHz low-power direct digital frequency synthesizer based on mixed structure of angle rotation, IEEE 8th International Conference on ASIC, 2009, pp. 1177-1179.