The Use of CORDIC in Software Defined Radios: A Tutorial

Javier Valls, Trini Sansaloni, Asun Pérez-Pascual, Vicente Torres, and Vicenç Almenar,
Universidad Politécnica de Valencia

ABSTRACT

CORDIC is a versatile algorithm widely used for VLSI implementation of digital signal processing applications. This article presents a tutorial of how to use CORDIC to implement different communication subsystems that can be found in a software defined radio. Specifically, it shows how to use CORDIC to implement direct digital synthesizers, AM, PM, and FM analog modulators and ASK, PSK and FSK modulators, up-/downconverters of in-phase and quadrature signals, full mixers for complex signals, and phase detection for synchronizers. The article also shows some tricks to efficiently implement the algorithm.

INTRODUCTION

The Coordinate Rotation Digital Computer (CORDIC) was introduced in 1959 by Volder [1]. It is an easy-to-implement and versatile algorithm widely used for digital signal processing applications. It computes iteratively the rotation of a two-dimensional vector using only add and shift operations. CORDIC has been traditionally used for hardware implementations. In [2] several algorithms which admit efficient implementation using CORDIC were reviewed: linear transforms, digital filtering, and matrix based DSP computing algorithms. It was shown that CORDIC-based architectures are a very appealing alternative to conventional multiply-and-add hardware. However, CORDIC may be also applied to implement different communication subsystems found in a digital radio: direct digital synthesizers; amplitude modulation (AM), phase modulation (PM), and frequency modulation (FM) analog modulators; amplitude shift keying (ASK), phase shift keying (PSK), and frequency shift keying (FSK) modulators, up-/downconverters of in-phase and quadrature signals, full mixers for complex signals, and phase detection for synchronizers.

This article shows how to use CORDIC in software defined radios (SDR). First, CORDIC is introduced as a computational resource that is able to rotate a vector by an angle, and convert from Cartesian to polar coordinates; also shown is a generic scheme suitable for implementing the different tasks required in communication systems. Second, several applications are commented on: direct digital synthesis; frequency, phase, and amplitude modulation; up-/downconversion; and frequency and phase synchronization for quadrature amplitude modulation (QAM) and orthogonal frequency-division multiplexing (OFDM) systems. Third, the CORDIC algorithm is exposed, and some implementation issues to improve its performance are included: how to enhance the spurious free dynamic range (SFDR) in direct digital synthesizers (DDS); how to avoid the $\pi/2$ multiplier of the phase accumulator; and how to simplify the computation of the scaling factor.

A FUNCTIONAL LOOK AT CORDIC

For an easy understanding of how to use the CORDIC algorithm in the implementation of digital intermediate frequency (IF) communications systems, CORDIC is presented in this section only as a computational resource with three inputs (X_0, Y_0, and Z_0) and three outputs (X_N, Y_N, and Z_N) that allows performing the following operations (illustrated in Fig. 1):

- Rotation of a vector (I,Q) by an angle θ when it is operating in rotation mode (RM); the rotated output vector is multiplied by a constant value K
- Cartesian-to-polar conversion, when it is operating in vectoring mode (VM); the modulus of the vector is also scaled by K

A generic scheme that shows how to use RM CORDIC to implement different digital communication tasks is illustrated in Fig. 2. The scheme is composed of an RM CORDIC where signals I and Q are connected into X_o and Y_o inputs, and the phase term θ connected into Z_o input is $\theta = (\Sigma[f_c + f_m]) + \phi_m) \cdot \pi$. This phase term is composed of the accumulation, at a sample period of T_s, of two frequency terms, f_c and f_m, and a phase term, ϕ_m. The additions involved in its computation are signed modulo-1 (limited to the interval $[-1,1]$), and the frequency and phase terms f_c, f_m and ϕ_m are normalized to 1. The CORDIC Z_o input needs a phase input that takes values in the interval $[-\pi, \pi]$, so a multiplication by π is required to extend the interval of the normalized term θ to the interval required by CORDIC.

CORDIC IN SDR COMMUNICATIONS SYSTEMS

This section shows how CORDIC algorithm is used to perform several tasks needed in SDR systems.

0163-6804/06/$20.00 © 2006 IEEE

This includes direct digital synthesis; FM, PM, and AM; ASK, PSK, and FSK; up-/downconversion of in-phase and quadrature signals; and frequency and phase synchronization subsystems.

Direct Digital Synthesis

Direct digital synthesis is a method to generate waveforms directly in the digital domain. In our case study of communications systems the target waveforms are the sine and cosine ones. A DDS is composed of a phase accumulator and a phase-to-amplitude converter [3], as shown in Fig. 3a. In a conventional DDS based on lookup tables (LUTs) the phase accumulator is an integer N-bit accumulator (an unsigned modulo-2^N accumulator), whose output directly addresses the LUT where the amplitude values of sine or cosine waves are stored. The maximum value of the accumulator (2^N–1) represents the phase 2π of the sine or cosine wave. The accumulator generates a ramp signal when it is incremented by a fixed value, due to its unsigned modulo-2^N property; hence, a periodic waveform is obtained at the output of the phase-to-amplitude converter (Fig. 3b).

The CORDIC algorithm configured in RM can behave as a quadrature phase-to-amplitude converter that directly generates sine and cosine waveforms [4]. The main advantage of using CORDIC-based DDS with respect to LUT-based methods is that it can achieve both high phase resolution and high precision with lower hardware cost [5]. A difference between both methods is that the phase accumulator generates an integer value that addresses an LUT in the LUT-based method, while it generates an angle in CORDIC-based DDS. Thus, in the last case a ramp signal in the interval $[-\pi,\pi,]$ must be obtained by the accumulator, as shown in Fig. 3c. This accumulator is easily implemented with an N-bit adder. A two's complement fractional numeric format (only one integer bit) is considered; hence, a ramp in the interval [–1,1] is generated, and a multiplier by π is introduced to achieve the desired range.

To generate sine and cosine waveforms of a digital frequency f_c with the scheme based on CORDIC of Fig. 2, the parameters f_m, ϕ_m, and Q must be zero and I = 1/K. The oscillation frequency is controlled by giving a fixed value to f_c. In such a case CORDIC generates directly the cosine and sine waveforms ($s_i(n) = cos(f_c \cdot \pi \cdot n)$ and $s_q(n) = sin(f_c \cdot \pi \cdot n)$) through X_N and Y_N outputs, respectively.

The maximum synthesized frequency is obtained by taking the value $f_c = 1$ (which is equivalent to analog frequency of $f_s/2$); and the minimum frequency is achieved with $f_c = 2^{-(N-1)}$, where N is the word-length of the accumulator.

Frequency, Phase and Amplitude Modulators

The CORDIC scheme of Fig. 2 can be used to directly generate in the digital domain at IF the analog modulations AM, PM, and FM, or the binary modulations ASK, PSK, and FSK.

Considering $m(n)$ as the modulator signal, AM can be implemented by choosing in Fig. 2 a carrier IF f_c, using the input X_o as modulator signal $I = m(n)/K$, and leaving to zero f_m, ϕ_m, and Q. In such a case the AM signal ($s(n) = m(n) \cdot cos(f_c \cdot \pi \cdot n)$) is generated through X_N CORDIC output.

If PM is desired, the terms f_m and Q of Fig. 2 are zeroed, the input X_o is fixed to $I = 1/K$, and the phase modulator signal is $f_m = m(n)$. Then the PM signal ($s(n) = cos(f_c \cdot \pi \cdot n + m(n) \cdot \pi)$) is obtained with a carrier frequency f_c, through the X_N output.

An FM signal can be generated with the scheme of Fig. 2 if the frequency modulator signal is $f_m = m(n)$, the carrier frequency is a fixed value f_c, the terms ϕ_m, and Q are zero and the X_0 input is $I = 1/K$. The FM signal ($s(n) = cos(f_c \cdot \pi \cdot n + (\Sigma m(n)) \cdot \pi)$) is also obtained by the output X_N.

In an AM, PM, or FM modulator, it is required to up-sample the base-band modulator signal $m(n)$ up to the sampling rate (f_s) of the CORDIC processor and DAC. As this signal usually is a narrow band one, a Cascade-Integrator-Comb filter (CIC) [6] is a very low hardware cost solution to perform this task.

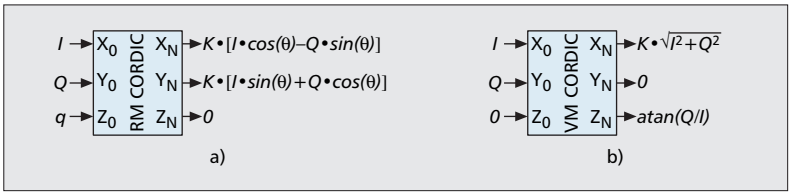

Figure 1. *a) Rotation mode CORDIC; b) vectoring mode CORDIC.*

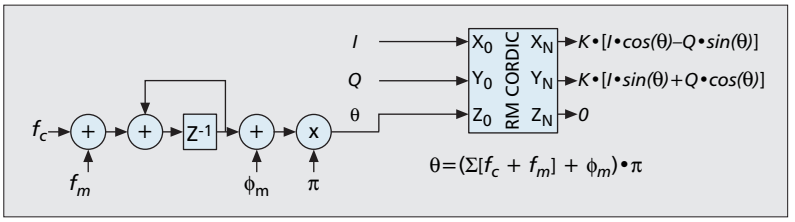

Figure 2. *Generic scheme to use CORDIC in RM.*

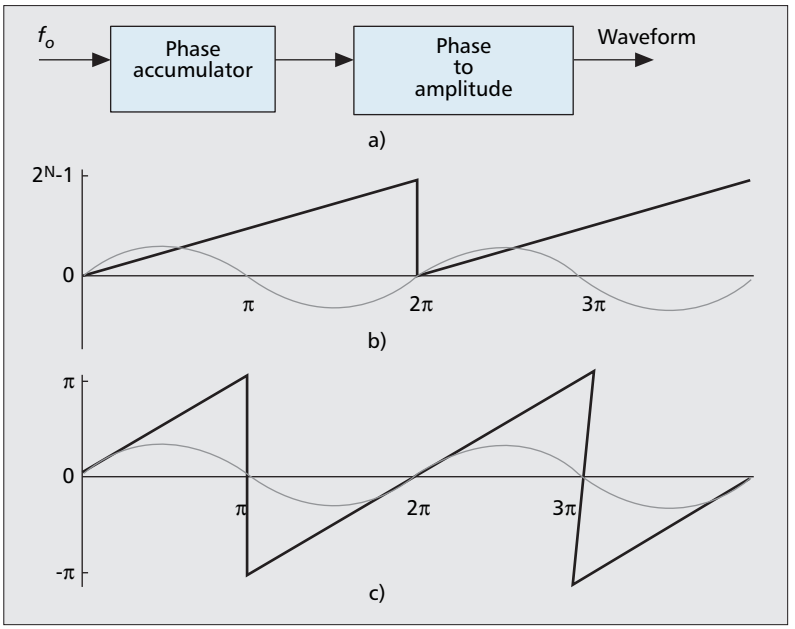

Figure 3. *a) DDS block diagram; b) waveforms of the LUT-based method; c) waveforms of the CORDIC-based method.*

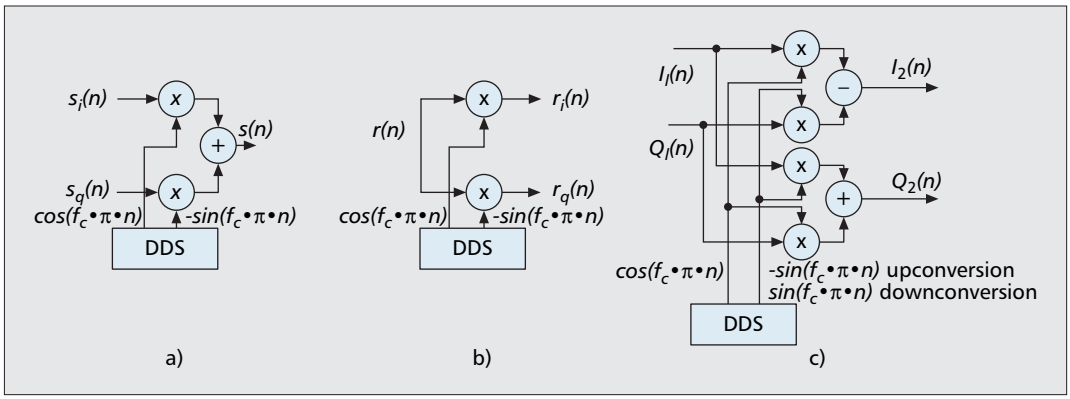

Figure 4. *a) Half mixer upconverter; b) half mixer downconverter; c) complex mixer up-/downconverter.*

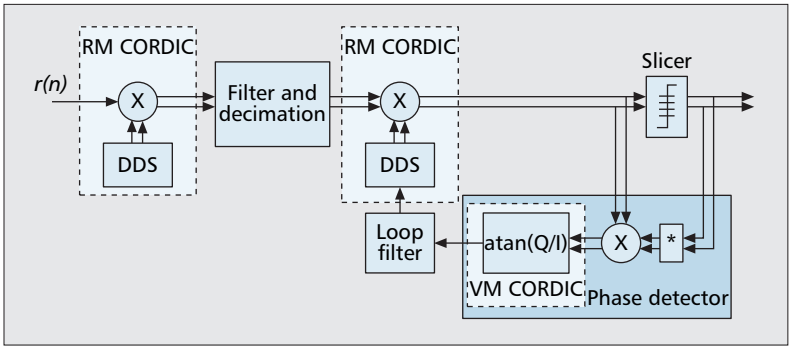

Figure 5. *Scheme of a digital IF QAM receiver.*

Digital modulations ASK, PSK, and FSK can also be generated with the scheme of Fig. 2 as AM, PM, and FM are, respectively, if the modulator signal $m(n)$ is the digital symbol stream. In such a case, the baseband modulator signal does not need to be upsampled, but needs to keep a constant value during the symbol time.

Up-/Downconversion: Digital Mixers in Quadrature Modulations

Digital up-/downconversion to/from an IF is a typical solution in SDR systems. In QAMa bitstream is grouped in symbols, and symbols are divided into their in-phase and quadrature components that are pulse shaped and interpolated up to the mixer rate. In an SDR receiver, the sampled received signal is mixed, and the oversampled in-phase and quadrature branches are decimated and filtered by the matched filter.

The upconversion mixer, shown in Fig. 4a, can be implemented with the CORDIC RM scheme of Fig. 2. Parameters f_m and ϕ_m must be zero, and the baseband in-phase and quadrature signals $s_i(n)$ and $s_q(n)$ are connected to X_0 and Y_0 from the output X_N, a scaled-by-K version of the mixed signal $(s(n) = K \cdot [s_i(n) \cdot cos(f_c \cdot \pi \cdot n) - s_q(n) \cdot sin(f_c \cdot \pi \cdot n)])$ is obtained.

In the receiver side, after sampling, the IF signal is mixed. This operation, depicted in Fig. 4b, can again be implemented using a CORDIC RM: the received signal $r(n)$ must be connected to Y_0, and X_0 must be zeroed. The scaled-by-K version of the in-phase and quadrature oversampled received signals $(r_i(n) = K \cdot r(n) \cdot cos(f_c \cdot \pi \cdot n)$ and $r_q(n) = -K \cdot r(n) \cdot sin(f_c \cdot \pi \cdot n))$ are obtained from Y_N and X_N, respectively.

Digital downconversion based on the Hilbert transform, and carrier and frequency synchronization are situations where complex mixing (Fig. 4c) of quadrature signals is required. This mixing can be seen as a multiplication of a complex signal $(I + jQ)$ by a complex exponential $e^{j\omega}$: if this frequency is positive (like in up-conversion) the in-phase signal I_1 is connected to X_0, and the quadrature signal Q_1 to Y_0, and their respective in-phase and quadrature outputs $(I_2 = K \cdot [I_1 \cdot cos(f_c \cdot \pi \cdot n) - Q_1 \cdot sin(f_c \cdot \pi \cdot n)]$ and $Q_2 = K \cdot [I_1 \cdot sin(f_c \cdot \pi \cdot n) + Q_1 \cdot cos(f_c \cdot \pi \cdot n)])$ are obtained from X_N and Y_N, respectively.

When the frequency is negative (as in downconversion) the in-phase signal I_1 is connected into Y_0 and the quadrature Q_1 into X_0, being the in-phase and quadrature output signals $(I_2 = K \cdot [Q_1 \cdot sin(f_c \cdot \pi \cdot n) + I_1 \cdot cos(f_c \cdot \pi \cdot n)]$ and $Q_2 = K([Q_1 \cdot cos(f_c \cdot \pi \cdot n) - I_1 \cdot sin(f_c \cdot \pi \cdot n)])$ obtained from Y_N and X_N, respectively.

The main advantage of using CORDIC as a digital mixer with respect to the conventional scheme of Fig. 4 (LUT-based DDS and multipliers) is that the multipliers and the ROM to store the sine and cosine waveforms are avoided. The extreme case is given by the complex mixer in which four multipliers and the LUT-based DDS is replaced by a single RM CORDIC.

Frequency and Phase Synchronization

First, we deal with single-carrier modulations. In Fig. 5 a block diagram of a digital IF receiver for QAM signals is shown. This scheme allows frequency and phase synchronization to be performed in baseband at a lower sampling frequency; thus, less hardware resources are needed. In downconversion it is usual to employ free-running oscillators (i.e., an RM CORDIC); this solution usually leaves some carrier frequency offset (CFO). The CFO correction is done by a quadrature complex mixer that again can be implemented using an RM CORDIC [7]. Finally, in order to drive the complex mixer it is necessary to estimate the CFO, which can be done by means of a VM CORDIC at the output of the phase detector [8]. Then the estimated CFO feeds the loop filter that controls the complex mixer.

Next, let us see how CORDIC can be used for frequency synchronization in IEEE 802.11a wire-

less LAN (WLAN)-OFDM receivers. In this standard data is transmitted in bursts, each preceded by a preamble, and receivers can use this preamble for frame, time, and frequency synchronization [9]. After downconversion to baseband, a maximum residual CFO of 200 kHz is allowed by the standard. This CFO must be estimated by the receiver, for example, if an autocorrelation scheme is employed to synchronize the receiver: the position of the maximum indicates where each OFDM symbol begins, and its phase is proportional to the CFO (a VM CORDIC can be used to calculate the phase). Then an RM CORDIC can be used to correct the CFO from the OFDM signal (f_c in the scheme of Fig. 2 would be the estimated CFO). Once the CFO is corrected, the receiver can perform a fast Fourier transform (FFT) of each OFDM symbol to obtain the transmitted data; to avoid errors caused by a residual CFO from an incorrect estimation, each OFDM symbol has four embedded pilot tones to calculate the angle deviation [9]. This deviation can be estimated again with a VM CORDIC and used in two ways: to rotate the 42 data pilots from the current OFDM symbol, or to improve CFO correction using the estimated phase as ϕ_m (Fig. 2) in the pre-FFT RM CORDIC [10].

The CORDIC Algorithm

CORDIC computes a pseudo-rotation of a two-dimensional vector instead of a perfect rotation. This means that the original vector is rotated by an angle θ, and its magnitude is enlarged by a constant factor K. The CORDIC algorithm iteratively computes the pseudo-rotation by an angle θ with the following iterations:

$$\begin{aligned} X_{i+1} &= [X_i - d_i \cdot 2^{-i} \cdot Y_i] \\ Y_{i+1} &= [Y_i + d_i \cdot 2^{-i} \cdot X_i] \\ Z_{i+1} &= Z_i - d_i \cdot \alpha_i. \end{aligned} \quad (1)$$

Instead of directly performing a rotation by the angle θ, CORDIC performs several microrotations by the angles $\alpha_i = \pm atan(2^{-i})$. This means that the rotation angle θ is broken down into a set of predefined angles α_i, so after a number of iterations the angle θ is approximated by $\Sigma(d_i \cdot \alpha_i)$, where d_i belongs to the set {–1,1}. CORDIC Eq. 1 admit two operating modes, the rotation mode (RM) and the vectoring mode (VM) that compute the equations indicated in Fig. 1, which depend on how the directions of the microrotations (d_i) are chosen: $d_i = sign(Z_i)$ for RM and $d_i = -sign(Y_i)$ for VM. A detailed explanation of how CORDIC iterations are obtained can be found in [11].

Implementation of CORDIC

The CORDIC algorithm is easily implemented in both software and hardware. Figure 6a shows an example of CORDIC RM pseudo-code for software implementation, and Fig. 6b an iteration stage for hardware implementation. To satisfy a P-bit precision CORDIC operation, $P + 1$ iterations are needed. Furthermore, the length of the data path to compute the X and Y variables has to be $N = P + 2 + \log_2(P)$ bits, and for the computation of Z only a precision of $P + 1$ bits is needed in the operations.

In this section three implementation issues that lead to enhanced performance via improving the SFDR of the generated waveform, or by reducing the computational load or hardware cost, are exposed.

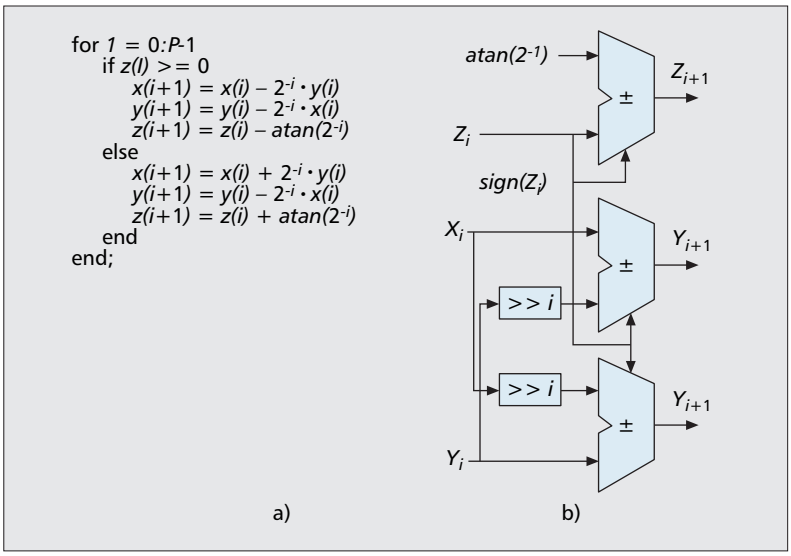

Figure 6. *a) RM CORDIC pseudo code; b) hardware implementation of a CORDIC iteration.*

Enhancing the SFDR in DDS

The convergence range of the CORDIC algorithm implemented by the iterations indicated in Eq. 1 is limited to the addition of all predefined angles: $\pm\Sigma\alpha_i \approx \pm 1.7433$. In order to use CORDIC as a phase-to-amplitude converter in a DDS the convergence range must be extended to $\pm\pi$, as shown in Fig. 3c. To accomplish this task two approaches can be considered. The first consists of adding at the beginning an extra iteration by $\pm\pi/2$, which is formulated as

$$\begin{aligned} X_1 &= -d_0 \cdot Y_0, \\ Y_1 &= d_0 \cdot X_0, \\ Z_1 &= Z_0 \cdot d_0 \cdot \pi/2. \end{aligned} \quad (2)$$

The second approach consists of using the following trigonometric identities:

$$\begin{aligned} cos(Z - \pi) &= -cos(Z), \\ sin(Z - \pi) &= -sin(Z), \end{aligned} \quad (3)$$

which allow computing an out-of-range angle by means of its complementary one. In such a case two extra two's complementers are required after conventional CORDIC operation together with a simple logic to detect an angle higher than $\pm\pi/2$.

Although both approaches lead to the same aim and exhibit similar computational load, the second achieves better SFDR than the first, due to better CORDIC precision within convergence range of ±1.7433. A DDS with 8-bit precision achieves 60 dB of SFDR if the first method is used and 63 dB if the second is applied. This enhancement is even higher if the conventional LUT-based quarter-wave symmetry method is utilized. A 12-bit-precision CORDIC-based DDS with quarter-wave symmetry achieves 84 dB of SFDR; using the second approach, it is 96 dB.

Avoiding Multiplication by π/2 or π

Besides the method to convert phase to amplitude, a difference between LUT-based and

A DDS with 8-bit precision achieves 60 dB of SFDR if the first method is used, and 63 dB if second method is applied. This enhancement is even higher if the conventional LUT-based quarter-wave symmetry method is utilized.

CORDIC-based DDS is that in the first, the phase accumulator is an unsigned modulus-2^N accumulator (N-bit word length) whose output directly addresses the LUT where the amplitude values are stored; in the second, the phase accumulator is a two's complement modulus-π accumulator whose output directly generates the phases (in radians) that have to be converted to amplitudes by CORDIC. However, the two's complement modulus-π accumulator is implemented as a two's complement modulus-1 accumulator followed by a multiplication by π [5] or by $\pi/2$, depending on if the first (Eq. 2) or second (Eq. 3) approach to extend the convergence range is used. For clarity it is supposed that the second method is used. Then, in Eq. 1 each Z_i is obtained as

$$Z_i = Z'_i \cdot \pi/2, \quad (4)$$

where Z'_i is the output of the two's complement modulus-1 accumulator. By replacing Eq. 4 in Eq. 1, the equation to compute angles remains

$$Z'_{i+1} = Z'_i - d_i\alpha'_i,$$

where $\alpha'_i = \alpha_i \cdot 2/\pi$. Therefore, now it is only necessary to change the set of predefined angles to compute CORDIC, thus avoiding multiplication by $2/\pi$. An additional advantage of this method is that for precision lower than 15 bits, the error bound of CORDIC is reduced from that of the direct codification of angles.

Simplifying the Correction Factor Multiplier

As previously stated, each iteration in CORDIC is not a perfect rotation since the length of the vector grows by a factor $K_i = (1 + 2^{-2i})^{1/2}$ [12]. Therefore, after N iterations its magnitude has changed by a factor $K = \Pi(1 + 2^{-2i})^{1/2}$ that converges to 1.6467. Therefore, in order to obtain a constant vector length, the result has to be scaled by $1/K$. CORDIC outputs can be scaled avoiding a full multiplication if the following approximation is used: $1/K = 0.60725$ ($^{1/2}((1 + 1/4) \approx (1 - 1/32)((1 + 1/256) \cdot (1 - 1/1024)$, which approximates the scaling factor with a precision of 16 bits. Then the full multiplier can be replaced by only four additions.

Conclusions

A tutorial on how to use the CORDIC algorithm to implement different blocks found in communications systems has been presented. Specifically, we have shown how to use CORDIC to implement direct digital synthesizers; AM, PM, and FM analog modulators, and ASK, PSK, and PSK modulators; up/down converters of in-phase and quadrature signals; full mixers for complex signals; and phase detection for synchronizers. We have also provided some tricks for efficiently implementing the algorithm.

Acknowledgments

This research was supported by FEDER, the Spanish Ministerio de Educación y Ciencia, under grant no. TEC2005-08406-C03-01, and Generatitat Valenciana, under Grant no. GV06/114.

References

[1] J. E. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Trans. Elect. Comp.*, vol. EC-8, no. 3, 1959, pp. 330–34.
[2] Y. Hu, "CORDIC-based VLSI Architectures for Digital Signal Processing," *IEEE Sig. Proc. Mag.*, July 1992, pp. 16–35.
[3] L. Cordness, "Direct Digital Synthesis: A Tool for Periodic Wave Generation (Part 1)," *IEEE Sig. Proc. Mag.*, vol. 21, July 2004, pp. 50–54.
[4] J. Vankka, "Methods of Mapping from Phase to Sine Amplitude in Direct Digital Synthesis," *IEEE Trans. Ultrason. Ferroelect. Freq. Control*, vol. 44, no. 2, Mar. 1997, pp. 526–34.
[5] F. Cardells-Tormo and J. Valls, "Area-Optimized Implementation of Quadrature Direct Digital Frequency Synthesizer on LUT-based FPGAs," *IEEE Trans. Circuits and Sys. II: Analog and Digital Sig. Proc.*, vol. 50, no. 3, Mar. 2003, pp. 135–38.
[6] E. Hogenauer, "An Economical Class of Digital Filters For Decimation and Interpolation," *IEEE Trans. Acoustics, Speech and Sig. Proc.*, vol. ASSP-29, Apr. 1981, pp. 155–62.
[7] F. Cardells *et al.*, "Efficient FPGA-based QPSK Demodulation Loops: Application to the DVB standard," *LNCS*, vol. 2438, Springer-Verlag, 2002, pp. 102–11.
[8] C. Dick, F. Harris, and M. Rice, "FPGA Implementation of Carrier Synchronization for QAM Receivers," *J. VLSI Sig. Proc.*, vol. 36, 2004, pp. 57–71.
[9] J. Heiskala and J. Terry, *OFDM Wireless LANs: A Theoretical and Practical Guide*, SAM, 2001.
[10] M. Engels, *Wireless OFDM Systems: How to Make Them Work?*, Kluwer, 2002.
[11] B. Parhami, *Computer Arithmetic: Algorithmic and Hardware Designs*, Oxford Univ. Press, 2000.

Additional Reading

[1] H. Myer, M. Moeneclaey, and S. Fechtel, *Digital Communication Receivers*, Wiley, 1998.

Biographies

Javier Valls (jvalls@eln.upv.es) received his telecommunication engineering degree from the Universidad Politecnica de Cataluña, Spain, and his Ph.D. degree in telecommunication engineering from the Universidad Politecnica de Valencia, Spain, in 1993 and 1999, respectively. He has been an associate professor in the Department of Electronics at Universidad Politecnica de Valencia since 1996. His current research interests include the design of FPGA-based systems, computer arithmetic, VLSI signal processing, and digital communications.

Trini Sansaloni received her telecommunication engineering and Ph.D. (telecommunication engineering) degrees from the Universidad Politecnica de Valencia in 1994 and 2001, respectively. She is an associate professor in the Department of Electronics at Universidad Politecnica de Valencia. Her current research interests include the design of FPGA-based systems and VLSI signal processing.

Asun Pérez-Pascual received her telecommunication engineering and Ph.D. (telecommunication engineering) degrees from the Universidad Politecnica de Valencia, Spain, in 1997 and 2002, respectively. She has been an associate professor in the Department of Electronics at Universidad Politecnica de Valencia since 2002. Her current research interests include the design of FPGA-based systems, computer arithmetic, VLSI signal processing, and digital communications.

Vicente Torres received his telecommunication engineering and Ph.D. (telecommunication engineering) degrees from the Universidad Politecnica de Valencia in 1994 and 2001, respectively. He has been an associate professor in the Department of Electronics at Universidad Politecnica de Valencia since 1995. His current research interests include the design of FPGA-based systems, with a focus on digital communications.

Vicenç Almenar received his telecommunication engineer and Ph.D. degrees from the Universidad Politecnica de Valencia (UPV) in 1993 and 1999, respectively. In 2000 he spent five months at the Centre for Communications Systems Research (CCSR), University of Surrey, United Kingdom, where he was involved in research on digital signal processing for digital communications. He is currently an associate professor in the Department of Communications, UPV. His current research interests include OFDM, MIMO, signal processing, and simulation of digital communications systems.