Adv. Radio Sci., 7, 133–137, 2009
www.adv-radio-sci.net/7/133/2009/

Advances in Radio Science

Resolving longitudinal amplitude and phase information of two continuous data streams for high-speed and real-time processing

A. Guntoro and M. Glesner

Institute of Microelectronic Systems, Department of Electrical Engineering and Information Technology, Technische Universität Darmstadt, Germany

Abstract. Although there is an increase of performance in DSPs, due to its nature of execution a DSP could not perform high-speed data processing on a continuous data stream. In this paper we discuss the hardware implementation of the amplitude and phase detector and the validation block on a FPGA. Contrary to the software implementation which can only process data stream as high as 1.5 MHz, the hardware approach is 225 times faster and introduces much less latency.

1 Introduction

Digital signal processing which replaces the analog control system in our heavy ion acceleration environment is depicted in Fig. 1. Two analog input signals which correspond to the ion states at a specific time are received from the cavity sensors and go to the A/D block (with 14-bit resolution at 28.5 MHz sampling rate). Longitudinal amplitude and phase information of both signals are extracted by the detector and go to the validation block. The validation block outputs the longitudinal amplitude and phase differences of both signals. Special phase-corrector filter tunes up the phase of the signal. The corrected signal is used at the end by the controller to control the DDS (Direct Digital Synthesis). In this paper we focus on the amplitude and phase detector and the validation block which should be able to process two high-speed continuous data streams.

Correspondence to: A. Guntoro
(guntoro@mes.tu-darmstadt.de)

2 Longitudinal amplitude and phase

Given a discrete signal $f[n]$, the longitudinal amplitude information of the signal is defined as Iverson (2004); Hind et al. (2003):

$$\chi_X = \sqrt{x^2 + y^2}$$

and the corresponding longitudinal phase information is

$$\phi_X = arctan\frac{y}{x}$$

with $x{=}Q_1{-}Q_2$ and $y{=}I_1{-}I_2$ where I_1, I_2, Q_1, and Q_2 are the in-phase and quadrature components of the signal.

To resolve the magnitude information, one *square root*, two *square*, and one *addition* operations are involved. Accordingly one *division* operation and one *arctan* function are involved to compute the phase.

The magnitude and phase difference of two signals are easily computed with:

$$\chi_\Delta = \chi_1 - \chi_2 \qquad \phi_\Delta = \phi_1 - \chi_2$$

Our original implementation of the amplitude and phase detector and the validation block which are performed on DSP C6713 is only capable to receive two input data streams as high as 1.5 MHz on each channel Klingbeil (2004). Thus the signals are decimated by 19 which introduces friction and information loss.

2.1 CORDIC

The CORDIC (COordinate Rotation DIgital Computer) algorithm is a method to compute elementary functions by means of planar rotation and vectoring. The basic of CORDIC principle utilizes only shift and add/sub operations which are simple and cheap from the resource allocation point of view Andraka (1998). The coordinate system of the input

Published by Copernicus Publications on behalf of the URSI Landesausschuss in der Bundesrepublik Deutschland e.V.

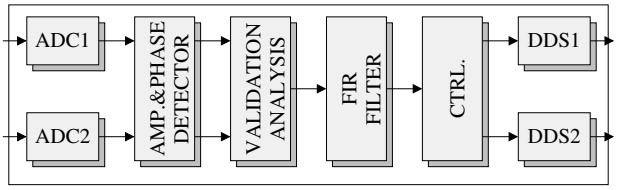


Fig. 1. Digital signal processing chain.

parameter is rotated through constant angles until the angle is reduced to zero.

Planar rotation for a vector A of (X_j, Y_j) can be defined in matrix form as:

$$\begin{bmatrix} X_j \\ Y_j \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \end{bmatrix}$$

To calculate the rotation, trigonometric functions (cos and sin) are involved. This rotation angle θ in fact can be executed in steps by means of iterative process. With a slight rearrangement, Eq. (1) defines this process which performs the rotation stepwise.

$$\begin{bmatrix} X_{n+1} \\ Y_{n+1} \end{bmatrix} = \cos\theta_n \begin{bmatrix} 1 & -\tan\theta_n \\ \tan\theta_n & 1 \end{bmatrix} \begin{bmatrix} X_n \\ Y_n \end{bmatrix} \quad (1)$$

By selecting the angle steps θ_n such that the tangent of a step ($\tan\theta_n$) is a power of 2, the multiplication operation can be eliminated and substituted by shift operation. Equation (2) gives the angle parameter for each step.

$$\theta_n = arctan\left(\frac{1}{2^n}\right) \quad (2)$$

Their sums must equal to the rotation angle θ:

$$\sum_{n=0}^{\infty} S_n\theta_n = \theta$$

where S_n={−1; +1} in respect to addition or substraction is determined later. Rewriting Eq. (1), we have:

$$\begin{bmatrix} X_{n+1} \\ Y_{n+1} \end{bmatrix} = \cos\theta_n \begin{bmatrix} 1 & -S_n 2^{-n} \\ S_n 2^{-n} & 1 \end{bmatrix} \begin{bmatrix} X_n \\ Y_n \end{bmatrix}$$

Except for the common coefficient $\cos\theta_n$ which can be treated as a constant K and computed at the end, multiplications are replaced with shift operations.

Residue Z which defines then angle difference between the expected rotation and the iterative rotations is:

$$Z_{n+1} = \theta - \sum_{i=0}^{n} \theta_i = \theta - \sum_{i=0}^{n} arctan\frac{1}{2^i}$$

The rotation parameter S_n is determined by:

$$S_n = \begin{cases} -1 & \text{if } Z_n < 0 \\ +1 & \text{if } Z_n \geq 0 \end{cases}$$

Practically, the iteration does not go up to infinity. In the hardware implementation, for each iteration the resolution of the result is increased by one bit (16 iterations deliver 16 bits result). Thus, it is not necesarry to compute the *arctan* on the fly. Look-up table of 16 entries is enough to store the precomputed *arctan* values in this case.

Driving Z to zero, the CORDIC performs:

$$\begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix} = \begin{bmatrix} P(X_i\cos Z_i - Y_i\sin Z_i) \\ P(Y_i\cos Z_i + X_i\sin Z_i) \\ 0 \end{bmatrix}$$

With initial values X_i=1/P, Y_i=0, and Z_i=θ we will have at the end of the iteration:

$$\begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix}$$

To suit our demand, we need to drive Y to zero instead of Z. The obtained result is:

$$\begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix} = \begin{bmatrix} P\sqrt{X_i^2 + Y_i^2} \\ 0 \\ Z_i + arctan\dfrac{Y_i}{X_i} \end{bmatrix}$$

Again with X_i=X, Y_i=Y, and Z_i=0 we obtain

$$\begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix} = \begin{bmatrix} P\sqrt{X^2 + Y^2} \\ 0 \\ arctan\dfrac{Y_i}{X_i} \end{bmatrix} \quad (3)$$

Neglecting the amplification factor P, X_j and Z_j correspond to the magnitude and phase information we need.

3 Hardware Realization

The critical constraint is to process two continuous data streams at a minimum of 28.5 MHz each without introducing any decimation. We exploit a pipeline structure within our design to guarantee this real-time demand. Figure 2 concisely depicts the tasks for each pipeline stage.

3.1 2-to-1 controller

This stage receives two input data streams. It collects four samples from each input and feeds them to the subtraction block. To save resources, two input data streams are processed alternately (i.e. the first clock cycle for the first input, the next clock cycle for the second input and so on).

3.2 Substraction

This stage handles the extraction of incoming quadrature and in-phase signals into corresponding X and Y which will be fed to the CORDIC core. These variables represent directly the vector of which the core operates.

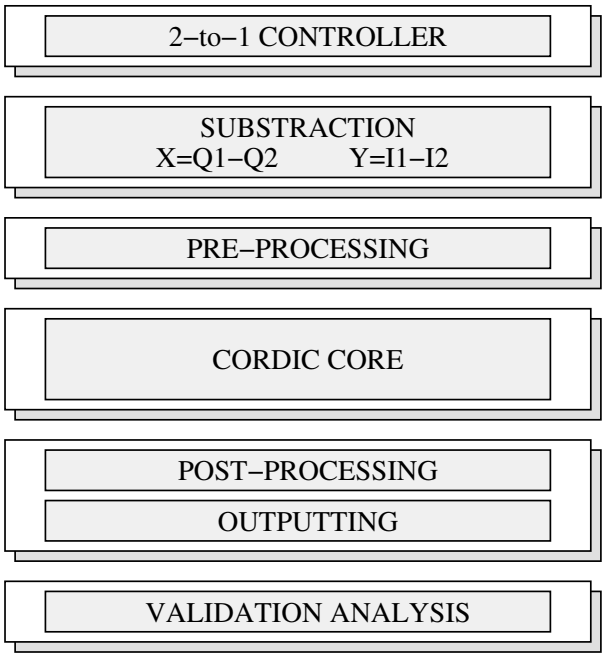


Fig. 2. Pipeline stages of the amplitude and phase detector and validation analysis block.

3.3 Preprocessing

The CORDIC vectoring is only capable of processing the signals arranged on quadrant I and IV. Since the combination of inputs X and Y may also lay on quadrant II or III, preprocessing and rearrangement of the inputs are necessary. The quadrant preprocessing conversion is formulated as

$$f'(x,y) = \begin{cases} f(x,y) & x \geq 0 \\ f(-x,-y)+\pi & x < 0, y \geq 0 \\ f(-x,-y)-\pi & x < 0, y < 0 \end{cases}$$

which tells that quadrant II is mapped to quadrant IV, thus the result will be corrected by adding π, and quadrant III is mapped to quadrant I and will be corrected by subtracting π from the result.

3.4 CORDIC core

The core stage expands the CORDIC iteration into micro pipelines to fulfill our high-speed requirement. 16 pipeline stages are needed to provide 16-bits resolution at the output. To have a full resolution swing, 16-bits two's complement binary format is used to represent the value between $-\pi$ and $+\pi$ for the angle component. LUT-based precalculated arctan 2^{-n} is used to store 16 arctan entries. Similar to the angle representation, two's complement is also used to represent the magnitude component. This is done because the iterative process inside of the core employs also addition and subtraction in estimating the magnitude.

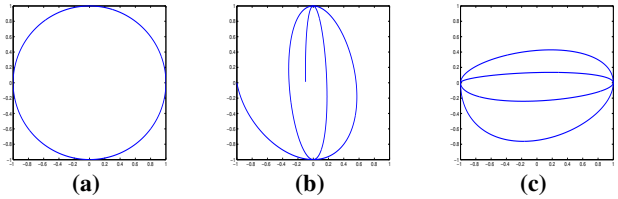


Fig. 3. Different test input sources.

Table 1. Maximum error, absolute mean, and standard deviation of the resulting error for the magnitude component (normalized to 1) for each input source.

Source	Max Error	$\lvert\bar{x}\rvert$	σ
Const. Mag.	0.000338	0.000111	0.000069
Decaying X	0.000316	0.000118	0.000072
Decaying Y	0.000359	0.000120	0.000072

3.5 Postprocessing and outputting

The Postprocessing stage corrects the output by adding or subtracting the result with π based on the information collected at the preprocessing stage. Additionally, the amplification factor which is introduced by CORDIC as defined in Eq. (3) might also be eliminated at this stage if it is demanded. Although it depends on the application, while most applications demand no unity gain correction for the magnitude output.

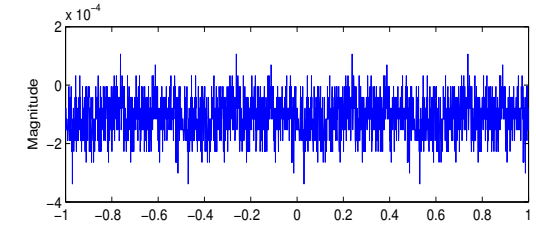
3.6 Validation analysis

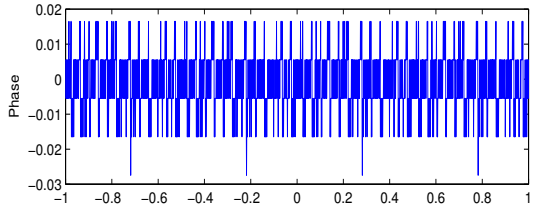
This stage calculates the difference between two longitudinal magnitudes and two longitudinal phases.

4 Performance

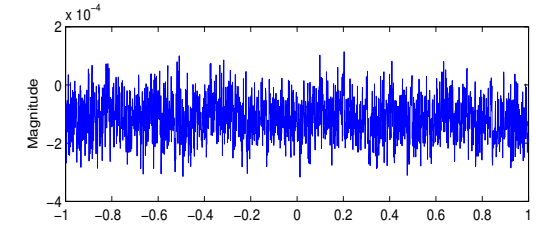
Three different input sources are utilized to analyze the performance of the CORDIC hardware. Figure 3 depicts these sources. Three cases are considered here: (a) a linear phase and constant magnitude input source, (b) a signal with decaying X component and (c) a signal with decaying Y component.

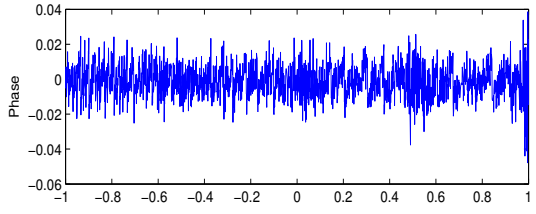
Figure 4 maps the errors (the difference between the expected values and the computed version) of magnitude and phase result of each input source. Here it can be seen that the hardware computation with 16-bit resolution produces the same results with some minor errors compared to the expected results. Tables 1 and 2 summarize the statistical parameters of the error for both magnitude and phase components.



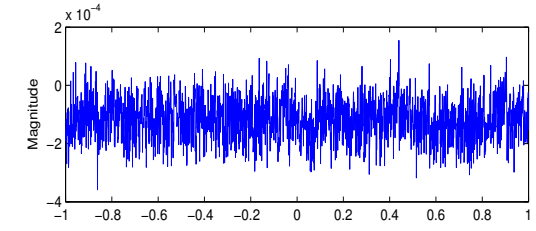


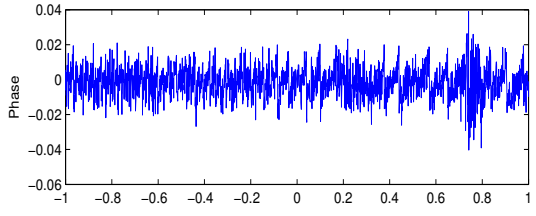
(a) Constant Magnitude





(b) Decaying X Component





(c) Decaying Y Component

Fig. 4. The difference between the expected and computed values of every given input source.

Table 2. Maximum error, absolute mean, and standard deviation (all in degree) of the resulting error for the phase component for each input source.

Source	Max Error	$\lvert\bar{x}\rvert$	σ
Const. Mag.	0.027466	0.007682	0.008817
Decaying X	0.047844	0.007344	0.009223
Decaying Y	0.040306	0.007084	0.008829

Table 3. Resource utilization on Xilinx XC2V2000-4.

	Usage/Cap.	Percentage
Slices	478/10752	4%
Flip-Flops	788/21504	3%
4 input LUTs	792/21504	3%
Max. Frequency	168.89 MHz	

Here we can see that by using 16-bit data representation to perform the computation, the introduced errors are tolerable. For the magnitude component, the maximum error is in order of 3×10^{-3}; and for the phase component, the maximum phase error does not exceed 0.05°.

5 Synthesis

The amplitude and phase detector and validation block are written in VHDL. The code has been synthesized using Xilinx ISE. As the target device, we use Xilinx XC2V2000-4 which is employed in our DSP chain. Table 3 details the resource usage of the implemented blocks on the target device. It is clear that the implemented block consumes only a small amount of resources and has high operation speed. Due to the pipeline structure, the amplitude and phase detector and the validation block can receive two input data streams as high as 337.78 MHz each (the core operated on 4 samples per clock cycle). Additionally, the computational latency determined by the number of pipeline stages (1+1+1+16+2+1=22) is very low (i.e. 130 ns).

6 Conclusions

Contrary to the DSP implementation which is only capable to process data streams as high as 1.5 MHz, the dedicated hardware block is 225 times faster and introduces only a very small amount of computational latency. Thus, the decimation by 19 which was taken on our original implementation is no longer required.

References

Andraka, R.: A survey of CORDIC algorithms for FPGA based computers, FPGA 98 Monterey CA USA, 1998.

Hind, M., Rajan, V., and Sweeney, P.: Phase shift detection: a problem classification, in: IBM Research Report, 2003.

Iverson, J.: Digital Control Technology Enhances Power System Reliability and Performance, in: Technical Information from Cummins Power Generation, Cummins Power Generation, 2004.

Klingbeil, H.: A Fast DSP-based Phase-detector for Closed-loop RF Control in Synchrotrons, IEEE Transaction on Instrumentation and Measurement, 2004.