OpenCPI
Polar to Rectangular CORDIC Component Data Sheet

OpenCPI Release: v2.3.4

Revision History

Revision	Description of Change	Date
v1.4		10/2018
v1.5		4/2019
v1.6	Convert Worker to Version 2 HDL API	11/2019
v1.7	Table of Worker Configurations and Resource Utilization Table removed	5/2020

Summary - Polar to Rectangular CORDIC

Name	pr_cordic
Worker Type	Application
OpenCPI Release	v2.3.4
Last Update	11/2019
Component Library	ocpi.assets.dsp_comps
Workers	pr_cordic.hdl
Tested Platforms	alst4, E310(PL), isim, Matchstiq-Z1(PL), ml605, modelsim, xsim, ZedBoard(PL)

Functionality

The Polar to Rectangular CORDIC component functions to perform a polar to rectangular conversion using a Coordinate Rotation Digital Computer (CORDIC) algorithm.

Worker Implementation Details

pr_cordic.hdl

Traditionally, CORDIC algorithms are represented with three inputs, x_0, y_0, and z_0 and their respective three outputs x_N, y_N, and z_N. For this polar to rectangular algorithm, we drive the *magnitude* to x_0 and the *phase* to z_0. This process is summarized below in Equation 1.

$$x_0 = R, y_0 = 0, z_0 = \theta \tag{1}$$

Both the *magnitude* and *phase* are normalized prior to the CORDIC processes. The *magnitude* is normalized from -1 to $+1$ and the *phase* is normalized from $-\pi$ to $+\pi$. Due to the iterative nature of CORDIC algorithms, there is a gain factor A_n that needs to be handled within the system. This system accounts for the gain in Equation 2 internally by scaling the *magnitude* by a Gain Correction Factor K_c as seen in Equation 3.

$$A_n = \prod_n \sqrt{1 + 2^{-2i}} \tag{2}$$

$$K_c = \frac{1}{A_n} = \prod_{i=0}^{N} cos(atan(2^{-i})), n = STAGES - 1 \tag{3}$$

Additionally, the *phase* input is prescaled internally by S_c in Equation 4 to simplify the arithmetic. Once S_c is applied to θ the input range will have a maximum negative value corresponding to $-\pi$ and a maximum positive value corresponding to $+\pi - \delta$, where δ is defined in Equation 5.

$$S_c = 2^{(DATA\_WIDTH-1)}/\pi \tag{4}$$

$$\delta = 2\pi/2^{(DATA\_WIDTH)} \tag{5}$$

This worker implementation takes the inputs from Equation 1 and scales them by the gain adjustment and phase prescaled to yield Equation 6.

$$x_0 = K_c R, y_0 = 0, z_0 = S_c/\theta \tag{6}$$

Lastly, the expected outputs are found in Equation 7.

$$x_N = Rcos(\theta), y_N = Rsin(\theta), z_N = 0 \tag{7}$$

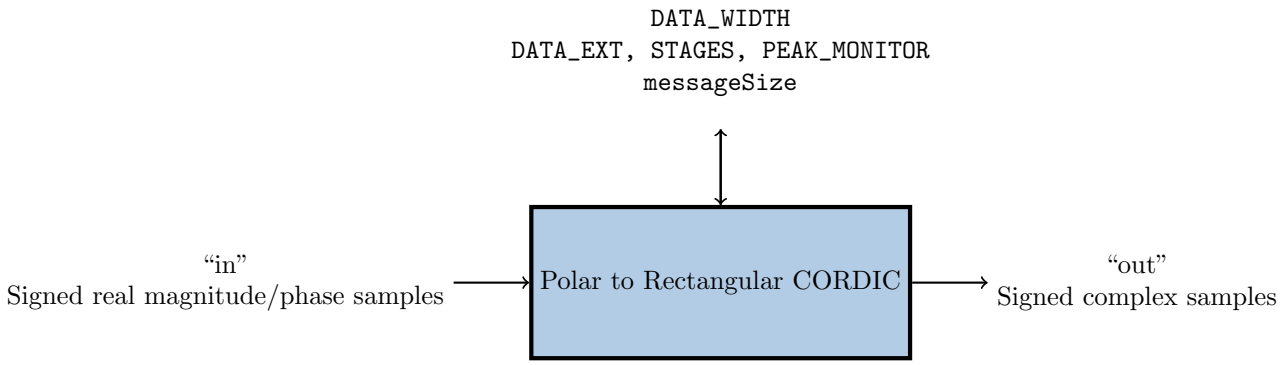
Theory

CORDIC algorithms are iterative algorithms used to calculate various trigonometric functions. This CORDIC algorithm specifically performs the conversion from the polar coordinate system to rectangular coordinate system. In polar coordinates, inputs to this component are the *radius* (or *magnitude*) and the *angle* θ, and expected outputs in the rectangular coordinate system are x and y. This process is summarized below in Equation 8.

$$R, \theta \rightarrow x, y \tag{8}$$

Block Diagrams

Top level



State Machine

None

Source Dependencies

pr_cordic.hdl

- projects/assets/components/dsp_comps/pr_cordic.hdl/pr_cordic.vhd
- projects/assets/hdl/primitives/dsp_prims/dsp_prims_pkg.vhd
  projects/assets/hdl/primitives/dsp_prims/cordic/src/cordic_pr.vhd
  projects/assets/hdl/primitives/dsp_prims/cordic/src/cordic.vhd
  projects/assets/hdl/primitives/dsp_prims/cordic/src/cordic_stage.vhd
- projects/assets/hdl/primitives/misc_prims/misc_prims_pkg.vhd
  projects/assets/hdl/primitives/misc_prims/round_conv/src/round_conv.vhd
- projects/assets/hdl/primitives/util_prims/util_prims_pkg.vhd
  projects/assets/hdl/primitives/util_prims/pd/src/peakDetect.vhd

Component Spec Properties

Name	Type	SequenceLength	ArrayDimensions	Accessibility	Valid Range	Default	Usage
DATA_WIDTH	UChar	-	-	-	-	-	Real input and complex output data width
DATA_EXT	UChar	-	-	-	-	-	CORDIC requirement: Number of growth bits implemented by CORDIC
STAGES	UChar	-	-	-	-	-	Number of CORDIC stages implemented
messageSize	UShort	-	-	Writable	8192	8192	Number of bytes in output message (Not implemented by Version 2)

Worker Properties

pr_cordic.hdl

Type	Name	Type	SequenceLength	ArrayDimensions	Accessibility	Valid Range	Default	Usage
SpecProperty	DATA_WIDTH	-	-	-	Parameter	1-16	16	Real input and complex output data width
SpecProperty	DATA_EXT	-	-	-	Parameter	1-16	6	CORDIC requirement: # of extension bits
SpecProperty	STAGES	-	-	-	Parameter	16-32	18	Number of CORDIC stages implemented
Property	PEAK_MONITOR	Bool	-	-	Parameter	Standard	true	Enable/Disable build-time inclusion of peak monitoring
Property	peak	Short	-	-	Volatile	Standard	-	Peak value of FM Discriminator output

Component Ports

Name	Producer	Protocol	Optional	Advanced	Usage
in	false	iqstream_protocol	false	-	Signed complex (Magnitude/Phase) samples
out	true	iqstream_protocol	false	-	Signed complex (I/Q) samples

Worker Interfaces

pr_cordic.hdl

Type	Name	DataWidth	Advanced	Usage
StreamInterface	in	32	-	Signed magnitude samples (31:16), Signed phase samples (15:0)
StreamInterface	out	32	InsertEOM=1	Signed complex samples

Control Timing and Signals

The CORDIC Polar-to-Rectangular HDL worker uses the clock from the Control Plane and standard Control Plane signals. This worker has a start-up transient delay of `STAGES`+2 valid samples. This means that once the input is ready and valid and the output is ready, it takes that many valid input samples before the first output sample is made available. The delays for this worker are itemized be as follows:

- 0 : pr_cordic.vhd
- 1 : pr_cordic.vhd/cordic_pr.vhd
- STAGES : pr_cordic.vhd/cordic_pr.vhd/cordic.vhd/cordic_stage.vhd
- 1 : cordic_pr.vhd/round_conv.vhd

Latency
`STAGES`+2 clock cycles

Worker Configuration Parameters

pr_cordic.hdl

Performance and Resource Utilization

pr_cordic.hdl

Test and Verification

A single test case is implemented to validate the CORDIC Polar-to-Rectangular component. An input file is generated with a constant magnitude of 29760 in bits (31:16) (driven into CORDIC input x_0) and multiple cycles of ramp data centered about zero representing a constant phase in bits (15:0) (driven into CORDIC input z_0). The amount of input data was chosen such that, at least two messages were output. The expected outputs are a cosine from I (the CORDIC x_N output) with a magnitude equal to that of x_0 and a sine from Q (the CORDIC y_N output) also with a magnitude equal to that of x_0. Input magnitude and phase plots may be viewed in Figures 1 and 2 below.

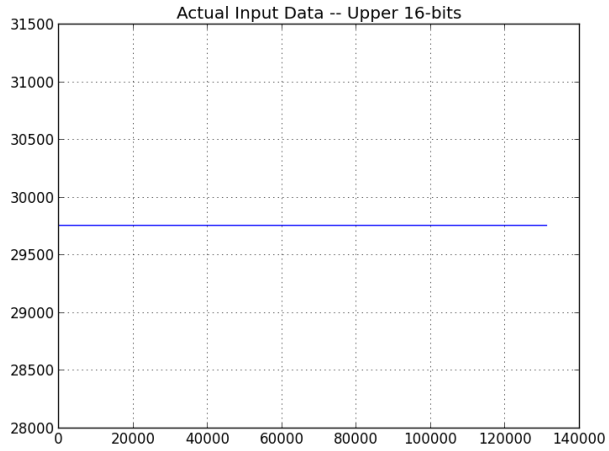


Figure 1: Time Domain Magnitude Input

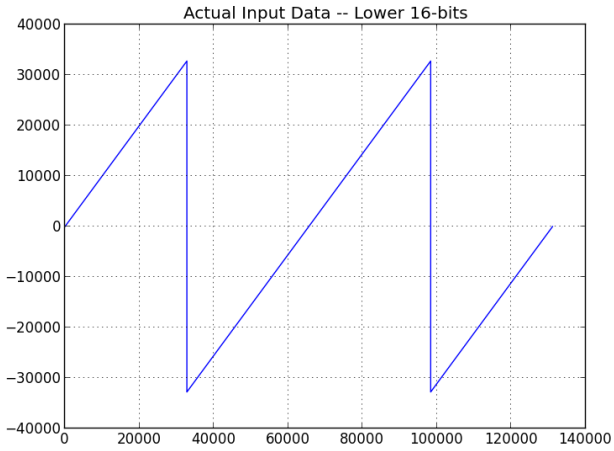


Figure 2: Time Domain Phase Input

Verification of output data is performed by creating cycles of a complex sinusoid in python to match the expected number of output data, and then comparing the worker output sample-by-sample with these expected results. A difference of ±1.0 is tolerated to allow for differences in fixed-point implementation and hardware rounding. Figures 3 and 4 depict the output of the Polar to Rectangular CORDIC.

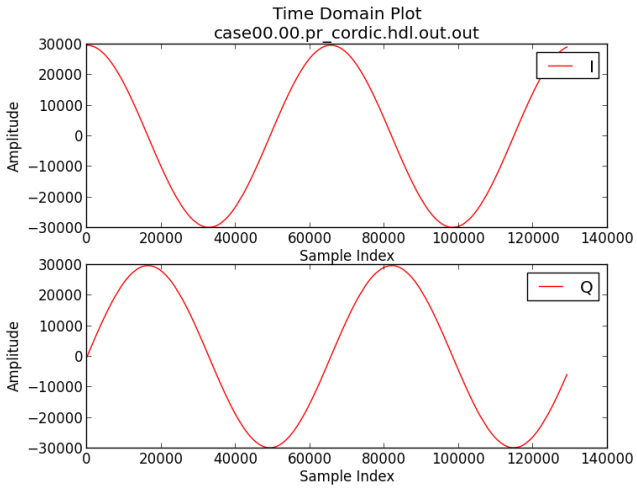


Figure 3: Time Domain Rectangular Output

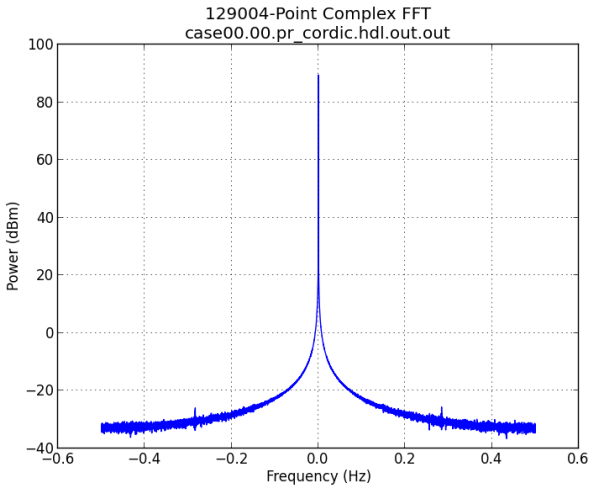


Figure 4: Frequency Domain Rectangular Output