Essential Components for Digital LLRF on FPGA

Qiang Du, Lawrence Doolittle

January 25, 2023

Lawrence Berkeley National Lab

Table of contents

Signal Generation

Arbitrary Waveforms

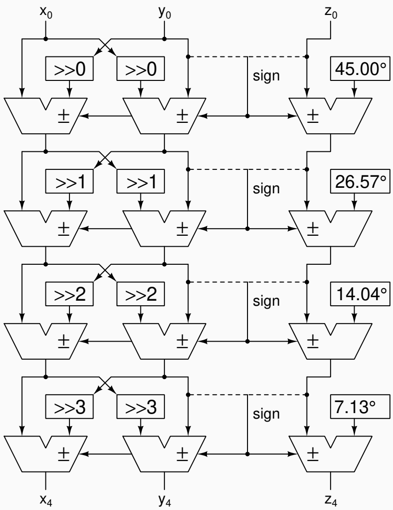
Basic Approach
A memory block is used to store and then continuously output a sequence of samples.

- Frequency quantization depends on sequence length.
- Any waveform can be produced.
- Limitations on square, sawtooth, and other signals.
- "White" noise.
- Shaped excitations.

CORDIC: COordinate Rotation DIgital Computer

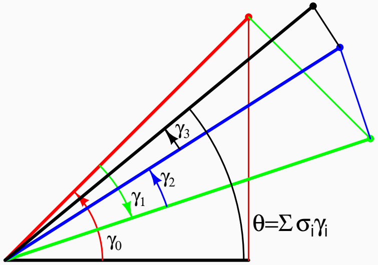
- COordinate Rotation DIgital Computer — first described by Jack E. Volder in 1959.
- Performs iterative coordinate rotations.
- Two basic modes:
  - Rotate an input vector by an arbitrary angle (rotation mode);
  - Rotate an input vector to align with x-axis (vectoring mode).
- Applications:
  - Sine and cosine generation;
  - Cartesian to polar transformation;
  - Arctangent computation;
  - Arcsine, arccosine;
  - Extensions to linear and hyperbolic functions.

CORDIC: COordinate Rotation DIgital Computer



- Elaborate series of shift-and-add/sub
- Each stage applies the matrix

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \begin{pmatrix} 1 & \pm 2^{-i} \\ \mp 2^{-i} & 1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix}$$



CORDIC is extremely FPGA friendly

Based on the observation that you can multiply a vector by the matrix

$$\begin{pmatrix} 1 & \sigma_i 2^{-i} \\ -\sigma_i 2^{-i} & 1 \end{pmatrix} = K_i \begin{pmatrix} \cos\sigma_i\gamma_i & \sin\sigma_i\gamma_i \\ -\sin\sigma_i\gamma_i & \cos\sigma_i\gamma_i \end{pmatrix}, \qquad \sigma_i \in \{-1, +1\}$$

using only shifts and adds; $K_i = 1/\sqrt{1 + 2^{-2i}}$ and $\gamma_i = \tan^{-1} 2^{-i}$.

A series of such rotations ($0 \le i \le N$) can rotate by an arbitrary angle (resolution 2^{-N}). With different processes to choose σ_i it can accomplish $R \to P$ and $P \to R$. The transformation's gain and total angle rotated are

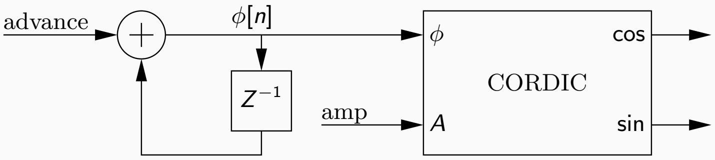
$$G_N = \prod_{i=0}^{N} K_i, \qquad \theta = \sum_{i=0}^{N} \sigma_i\gamma_i$$

- G_N is independent of σ_i and quickly approaches 1.64676... as $N \to \infty$.
- Hardware implementations need a pre-computed table of γ_i values.
- More stages (increasing N) give more accuracy, use more gates, and add delay.

Direct Digital Synthesis Topology

- A phase accumulator followed by a wave shape generator.
- Accumulator advance per clock cycle is adjustable:
  - Changes the frequency;
  - Advance can be modulated as well.
- Wave shape generator — memory or CORDIC.
- With a 30-bit accumulator (MSB=π) frequency quantization is $f_s/10^9$.
- Efficient accumulators (Bresenham's line algorithm).
- With small adjustments to the phase accumulator, a DDS also has the capability to make chirps and lock to a phase reference.

CORDIC in DDS



- Run in rotation mode.
- New phase angle every clock sample.
- Get sine and cosine every clock sample.

DDS: Binary and non-binary fraction phase steps

Binary fraction phase steps:

Common in many NCOs

$$f_{\mathrm{DDS}} = \frac{FTW}{2^N} f_{\mathrm{S}}$$

where FTW is a N-bit "Frequency Tuning Word".

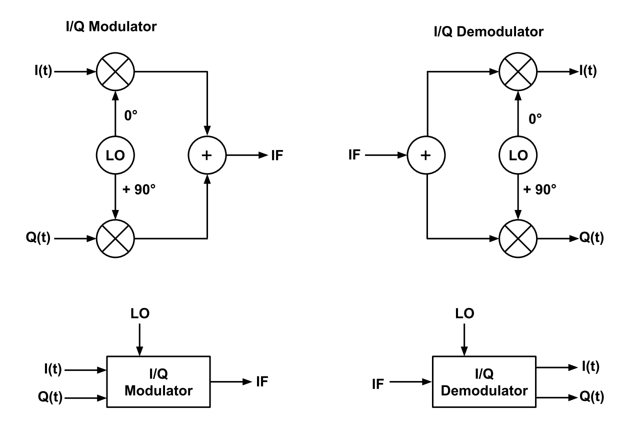
Non-binary fraction phase steps:

- Mathematically equivalent to Bresenham line algorithm.
- LSB rolls over using a "modulo" register.

- Lab 6: Build your own non-binary-fraction phase DDS
- Lab 11: Feedback control
- Lab 12: Feedback Loop Network Analyzer

Frequency Conversion

Heterodyne frequency mixing, I&Q Representation



IQ modulation and demodulation block diagram. Wikipedia

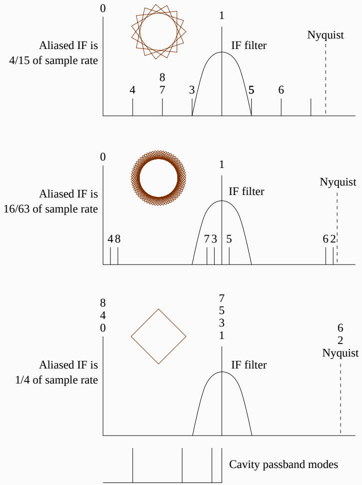
I&Q Representation

$$x(t) = I(t)\cos(\omega t) + Q(t)\sin(\omega t)$$

- Narrow band technique.
- As you move away from ω, signals are further from quadrature.

Non-I&Q Sampling [1]

- IQ sampling, with a coherent sampling period of 4 samples, is unusually sensitive to differential nonlinearity(DNL).
- Small amounts of DNL generate high harmonics of an input carrier.
- All odd harmonics of the signal alias to the same frequency as the carrier itself. Non-IQ sampling moves harmonics away.
- Balancing IF filter latency and linearity is the key for designing.
- Lab 8: ADC Characterization shows harmonics and Non-IQ sampling



Digital Down-Conversion (DDC) using Non-IQ sampling

With $\frac{f_{\text{IF}}}{f_{\text{S}}} = \theta$:

$$\begin{pmatrix} y_n \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} \cos(n\theta) & \sin(n\theta) \\ \cos((n+1)\theta) & \sin((n+1)\theta) \end{pmatrix} \begin{pmatrix} I \\ Q \end{pmatrix}$$

$$\begin{pmatrix} I \\ Q \end{pmatrix} = \frac{1}{\sin\theta} \begin{pmatrix} \sin((n+1)\theta) & -\sin(n\theta) \\ -\cos((n+1)\theta) & \cos(n\theta) \end{pmatrix} \begin{pmatrix} y_n \\ y_{n+1} \end{pmatrix}$$

Digital Down-Conversion (DDC) using Non-IQ sampling

Non-IQ sampling avoids aliasing for high precision digitization. [1]

Examples:

Advanced Light Source-U:

$$f_{\text{MO}} = 500.394\,\text{MHz}$$
$$f_{\text{LO}} = f_{\text{MO}}\frac{11}{12} = 458.695\,\text{MHz}$$
$$f_{\text{IF}} = f_{\text{MO}}\frac{1}{12} = 41.699\,\text{MHz}$$
$$f_{\text{S}} = f_{\text{LO}}\frac{1}{4} = 114.673\,\text{MHz}$$

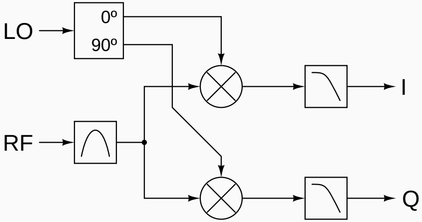
Brazilian Light Source:

$$f_{\text{MO}} = 500\,\text{MHz}$$
$$f_{\text{LO}} = f_{\text{MO}}\frac{23}{24} = 479.17\,\text{MHz}$$
$$f_{\text{IF}} = f_{\text{MO}}\frac{1}{24} = 20.83\,\text{MHz}$$
$$f_{\text{S}} = f_{\text{LO}}\frac{1}{4} = 119.79\,\text{MHz}$$

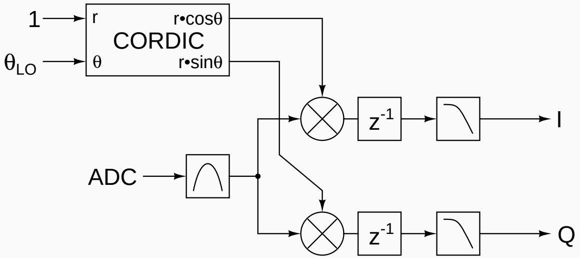
This class LLRF firmware:

$$f_{\text{MO}} = 480\,\text{MHz}$$
$$f_{\text{LO}} = f_{\text{MO}}\frac{23}{24} = 460\,\text{MHz}$$
$$f_{\text{IF}} = f_{\text{MO}}\frac{1}{24} = 20\,\text{MHz}$$
$$f_{\text{S}} = f_{\text{LO}}\frac{1}{4} = 115\,\text{MHz}$$

Digital Down-Conversion (DDC) using Non-IQ sampling

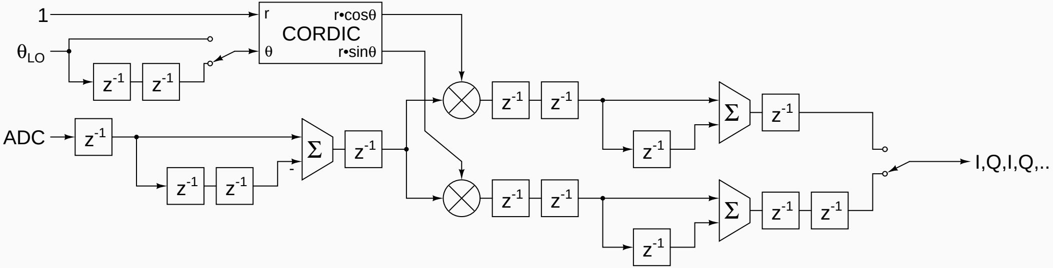


(a) Analog IQ Down-Conversion



(b) Digital Non-IQ DDC

Digital Down-Conversion (DDC) using Non-IQ sampling

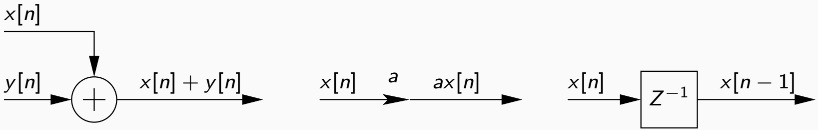


Non-IQ DDS with digital filters and serialization

Lab 10: Digital Down Conversion simulation

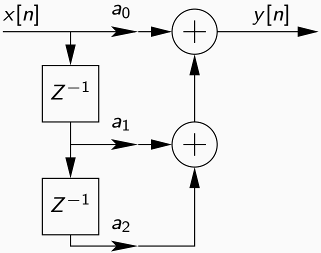
Digital Filtering

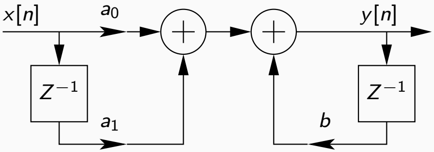
Digital Filtering Basics



Summation | Scaling by a coefficient | One sample delay

Legend

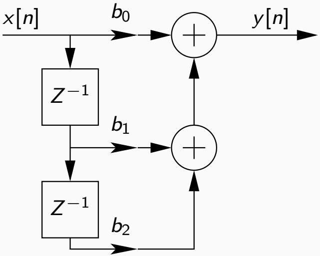




Two Classes of Filters

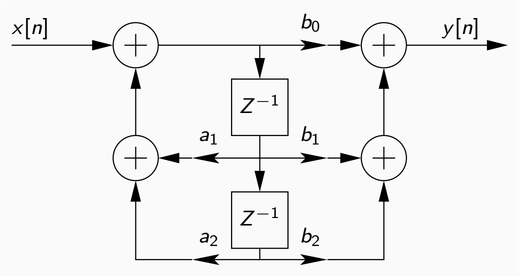
- All linear time-invariant digital filters can be split into two classes:
  - Finite Impulse Response (FIR): filter output depends only on a finite number of past input samples;
  - Infinite Impulse Response (IIR): filter has internal memory, output theoretically persists to infinity.
- Internal memory — feedback.
- Feedback can be unstable — IIR filter designer has to worry about stability.
- FIR filters are unconditionally stable.

FIR Filter



- Response of an FIR: $y[n] = \sum_{i=0}^{N-1} b_i x[N-1-i]$
- Each term in the sum is called "tap".
- N-tap filter requires N multiplies and N adds.
- Z-transform of FIR response: $H(z) = \sum_{i=0}^{N-1} b_i z^{-i}$

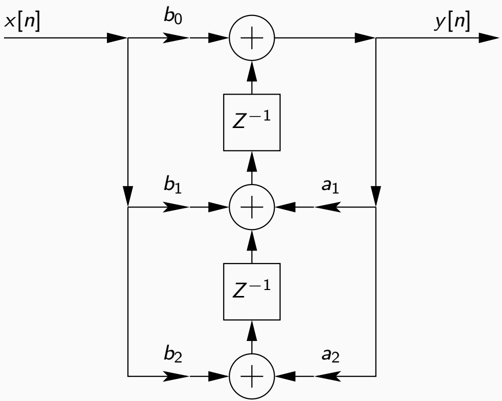
IIR Filter: Biquad Structure



- Direct Form II realization
- Second-order transfer function
- `https://en.wikipedia.org/wiki/Digital_biquad_filter`

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} = \frac{b_0 z^2 + b_1 z + b_2}{z^2 + a_1 z + a_2}$$

IIR Filter: Transposed Direct Form II realization



$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} = \frac{b_0 z^2 + b_1 z + b_2}{z^2 + a_1 z + a_2}$$

IIR Filter Stability

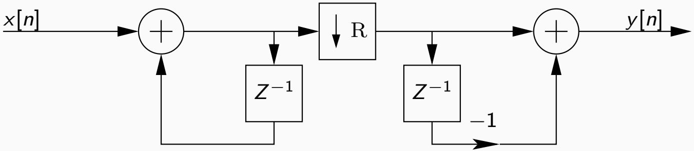
- Z-domain transfer function is stable if the poles (roots of the denominator polynomial) are within a unit circle.
- $|p| < 1$
- Critically stable for $|p| = 1$.
- Integrator is critically stable: $y_n = y_{n-1} + x_n$.

Good Filters

- Structures for efficient filter implementation:
  - Resource usage — no multiplies;
  - Resource usage — many zero coefficients;
  - Resource usage — symmetric structures;
  - Improving quantization effects.
- A few examples
- Cascaded Integrator Comb (CIC)
- Half-band filters
- Lattice structures

CIC filter: (Cascaded-Integrator-Comb)

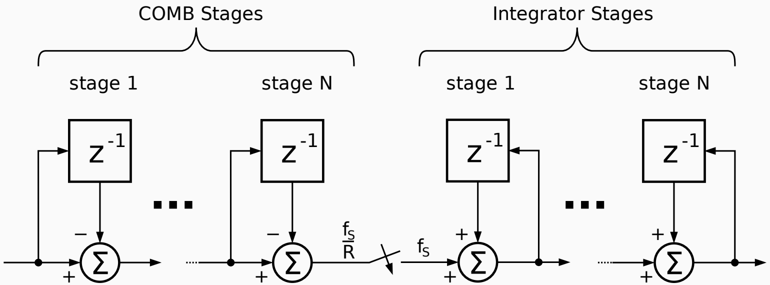
Invented by Eugene B. Hogenauer in 1981, CIC filters are a class of FIR filters used in multi-rate digital signal processing. Combination of N Comb stages and N Integrator stages, for interpolator or decimator.



Sampling rate reduced by R.

Lab 9: CIC filter simulation

CIC filter: Concatenating N stages



CIC interpolator by factor R, with N stages [2]

CIC filter: Characteristics

Integrator:

$$y[n] = y[n-1] + x[n]$$
$$H_I(z) = \frac{1}{1 - z^{-1}}$$

Differentiator (Comb):

$$y[n] = x[n] - x[n - RM]$$
$$H_C(z) = 1 - z^{-RM}$$

$$H(z) = H_I^N(z) H_C^N(z) = \frac{(1 - z^{-RM})^N}{(1 - z^{-1})^N} = \left(\sum_{k=0}^{RM-1} z^{-k} \right)^N$$

where N is number of stage, M is number of samples per stage, R is decimation or interpolation ratio.

Characteristics:

- Linear phase response
- Only uses delay, addition and subtraction, no need for multiplication

CIC filter

"This equation shows that even though a CIC has integrators in it, which by themselves have an infinite impulse response, a CIC filter is equivalent to N FIR filters, each having a rectangular impulse response."

- Matthew P. Donadio

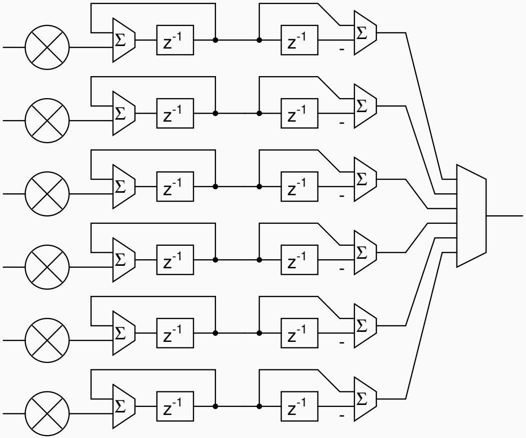
"Since all of the coefficients of these FIR filters are unity, and therefore symmetric, a CIC filter also has a linear phase response and constant group delay."

- Matthew P. Donadio

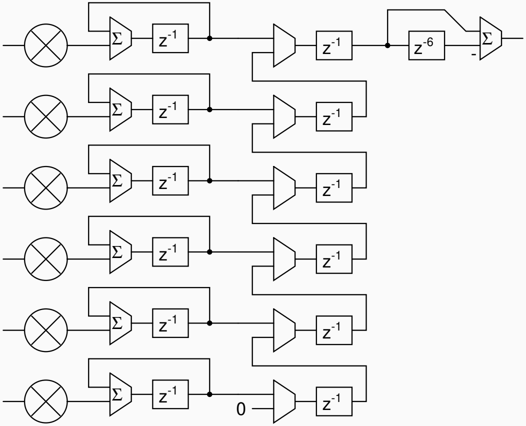
If the rate-change step is made programmable, the filter pass-band behavior changes too. This property makes CIC filters a great match to signal processing at the entrance to waveform memory, sometimes followed by one or more half-band filters.

Designers of CIC filters have to pay attention to system gain (RM^N) and bit growth.

Multi-channel CIC Decimator: FPGA Resource optimization [3]



∼4 cells per bit per channel, multiplexer a challenge to route for speed



∼2 cells per bit per channel, routing and timing easy and scalable

Feedback Controller

PID controller

proportional example

```
wire signed [KW-1:0] Kp;
wire signed [EW-1:0] error;
reg signed [KW+EW-1:0] prop=0;
reg signed [KW+EW-1:0] prop_out=0;
always @(posedge clk) begin
    if (reset_all) begin
        prop <= 0;
        prop_out <= 0;
    end else begin
        prop <= error * Kp;
        prop_out <= prop;
    end
end
```

integral example

```
wire signed [KW-1:0] Ki;
wire signed [EW-1:0] error;
reg signed [KW+EW-1:0] intg=0;
reg signed [KW+EW-1:0] intg_out=0;
always @ (posedge clk) begin
    if (reset_all) begin
        intg <= 0;
        intg_out <= 0;
    end else begin
        intg <= error * Ki;
        intg_out <= intg + intg_out;
        end
end
```

Acknowledgement

A part of the course material is from the RF and Digital Signal Processing course, Dmitry Teytelman, Dimtel, Inc. and Dan Van Winkle, SLAC, USPAS, June 2009.

Lawrence Doolittle, Hengjie Ma, and Mark Stuart Champion.
Digital Low–Level RF Control using Non–IQ Sampling.
In *Proceedings of LINAC 2006, Knoxville, Tennessee USA*, pages 568–570. Citeseer, 2006.

E. Hogenauer.
An economical class of digital filters for decimation and interpolation.
IEEE Transactions on Acoustics, Speech, and Signal Processing, 29(2):155–162, 1981.

Lawrence Doolittle.
DSP Tutorial III: Avoiding Resource Overutilization in FPGAs.
In *Proceedings of Low-Level RF workshop, Lake Tahoe, CA USA*, 2013.