Digital Signal Processing Techniques

Dmitry Teytelman

Dimtel, Inc., San Jose, CA, 95124, USA

June 17, 2009

Outline

Linear and Non-linear DSP

- Yesterday we talked mostly about linear signal processing — filtering.
  - Of course sampling and quantization is non-linear;
  - Sampling rate change (decimation).
- Many more exciting non-linear things to do.

Linear and Non-linear DSP

- Yesterday we talked mostly about linear signal processing — filtering.
  - Of course sampling and quantization is non-linear;
  - Sampling rate change (decimation).
- Many more exciting non-linear things to do.

Linear and Non-linear DSP

- Yesterday we talked mostly about linear signal processing — filtering.
  - Of course sampling and quantization is non-linear;
  - Sampling rate change (decimation).
- Many more exciting non-linear things to do.

Problem Partitioning

- When you do real-time signal processing, problem partitioning is critical.
- DSP complexity scale.
- Push floating point elaborate algorithms to a CPU, not in real time.
- Project the problem into a set of coefficient setpoints, controlling real-time hardware.

Problem Partitioning

- When you do real-time signal processing, problem partitioning is critical.
- DSP complexity scale.
- Push floating point elaborate algorithms to a CPU, not in real time.
- Project the problem into a set of coefficient setpoints, controlling real-time hardware.

Problem Partitioning

- When you do real-time signal processing, problem partitioning is critical.
- DSP complexity scale.
- Push floating point elaborate algorithms to a CPU, not in real time.
- Project the problem into a set of coefficient setpoints, controlling real-time hardware.

Problem Partitioning

- When you do real-time signal processing, problem partitioning is critical.
- DSP complexity scale.
- Push floating point elaborate algorithms to a CPU, not in real time.
- Project the problem into a set of coefficient setpoints, controlling real-time hardware.

Non-linear Methods

- Signal synthesis.
- Downconversion.
- Upconversion.
- Sampling rate changes: decimation and upsampling.
- Quadratic functions.
- Trigonometric functions.

Non-linear Methods

- Signal synthesis.
- Downconversion.
- Upconversion.
- Sampling rate changes: decimation and upsampling.
- Quadratic functions.
- Trigonometric functions.

Non-linear Methods

- Signal synthesis.
- Downconversion.
- Upconversion.
- Sampling rate changes: decimation and upsampling.
- Quadratic functions.
- Trigonometric functions.

Non-linear Methods

- Signal synthesis.
- Downconversion.
- Upconversion.
- Sampling rate changes: decimation and upsampling.
- Quadratic functions.
- Trigonometric functions.

Non-linear Methods

- Signal synthesis.
- Downconversion.
- Upconversion.
- Sampling rate changes: decimation and upsampling.
- Quadratic functions.
- Trigonometric functions.

Non-linear Methods

- Signal synthesis.
- Downconversion.
- Upconversion.
- Sampling rate changes: decimation and upsampling.
- Quadratic functions.
- Trigonometric functions.

Outline

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.

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.

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.

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.

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.

LLRF4 Example

- Memory block of 2048 samples.
- Short — causes coarse frequency quantization on sinewaves.
- Still provides a flexible tool.

LLRF4 Example

- Memory block of 2048 samples.
- Short — causes coarse frequency quantization on sinewaves.
- Still provides a flexible tool.

LLRF4 Example

- Memory block of 2048 samples.
- Short — causes coarse frequency quantization on sinewaves.
- Still provides a flexible tool.

Outline

A Great Tool

- 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.

A Great Tool

- 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.

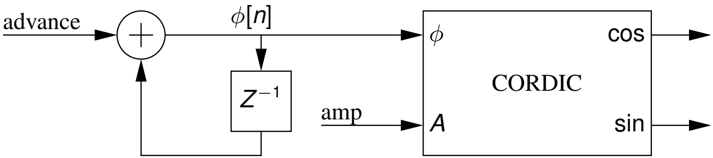
A Great Tool

- 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.

A Great Tool

- 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 in DDS



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

Outline

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).

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).

Outline

I&Q Definition

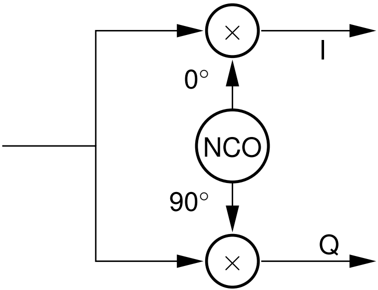
I&Q Representation

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

- Narrowband technique.
- As you move away from ω, signals are further from quadrature.
- Obvious transition to polar coordinates.
  - Analog domain processing historically favored polar coordinates.
  - Involves amplitude and phase detectors.
  - Phase shifters, VGAs, variable attenuators.

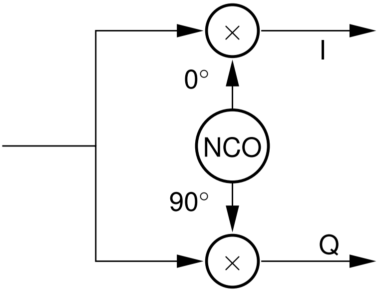
Outline

Downconversion



- Project the input signal into quadrature components.
- Is something missing on this block diagram?

Downconversion



- Project the input signal into quadrature components.
- Is something missing on this block diagram?

Optimized Demodulation

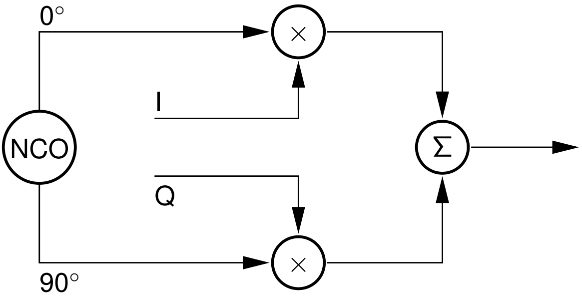
- Idea from Larry Doolittle.
- Carrier signal with θ phase advance per sample.
- $y_n = I \cos n\theta + Q \sin n\theta$
- $y_{n+1} = I \cos(n+1)\theta + Q \sin(n+1)\theta$
- Rewrite as

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

To reconstruct I and Q, take the these two samples and multiply by the inverse matrix:

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

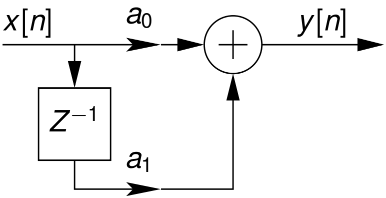
Upconversion



- Start from I and Q (baseband or IF).
- Frequency translation.
- Phase and frequency shifting, modulations.

Outline

Gain and Phase Block

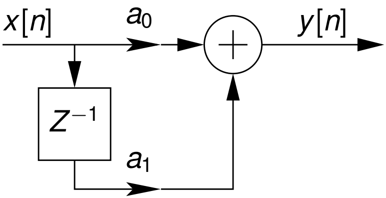


- Two-tap FIR filter.
- θ is the IF phase advance per sampling period.
- Works well near $\theta = \pi/2$.

Coefficients

$$\begin{bmatrix} 1 & \cos\theta \\ 0 & \sin\theta \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = G \begin{bmatrix} \cos\phi \\ \sin\phi \end{bmatrix}$$

Gain and Phase Block

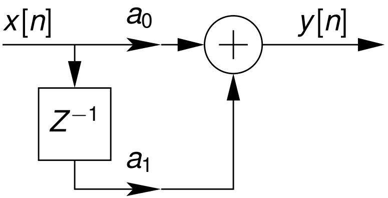


- Two-tap FIR filter.
- θ is the IF phase advance per sampling period.
- Works well near $\theta = \pi/2$.

Coefficients

$$\begin{bmatrix} 1 & \cos\theta \\ 0 & \sin\theta \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = G \begin{bmatrix} \cos\phi \\ \sin\phi \end{bmatrix}$$

Gain and Phase Block



- Two-tap FIR filter.
- θ is the IF phase advance per sampling period.
- Works well near $\theta = \pi/2$.

Coefficients

$$\begin{bmatrix} 1 & \cos\theta \\ 0 & \sin\theta \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = G \begin{bmatrix} \cos\phi \\ \sin\phi \end{bmatrix}$$