LabVIEW -
VI pre spracovanie signálov

(použité informačné zdroje National Instruments)

doc. Ing. Ján Šaliga, PhD.

KEMT FEI TU Košice

Knižnice pre spracovanie signálov

- **Signal processing VIs** – typické funkcie pre generovanie, spracovanie a analýzu signálov
- **Mathematics VIs** – rôzne zložité matematické funkcie, ktoré je možné využiť ako doplnok a rozšírenie funkcií z Signal processing VI
- Využitie polymorfizmu z **Numeric Functions** (jednoduché matematické funkcie a konverzie komplexných čísel) a VI z **Waveform VIs and Functions** (najmä výber a vloženie poľa dát do clustera, AČ prevod = digitalizácia analógového signálu)
- Využitie VI z **Array Vis** (výber skupiny vzoriek, spájanie, mazanie, ...) a **Cluster and Variant Vis** (rozdelenie, spojenie náhrada položky v zložitých štuktúrach = clusteroch)
- Rôzne toolkity, napr.
  - SignalExpress™ Express VIs
  - Signal Processing Toolset VIs
  - Sound and Vibration VIs
  - ...

Prehľad funkcií v triede Signal Processing

Filters Vis implementácia IIR, FIR, and nelineárnych filtrov.

Point By Point Vis spracovanie signálov po bodoch (nie v dávkach ako inde).

Signal Generation Vis generovanie jednorozmerných polí vzoriek analógových a číslicových signálov

Signal Operation Vis úprava signálov

Spectral Analysis Vis spektrálna analýza (pozor na fyzikálny význam a jednotky výsledných hodnôt).

Waveform Generation Vis generovanie signálov v odlišnom formáte (vaweform – špeciálny cluster s doplnkovými informáciami o časových súvislostiach, napr. perióda vzorkovania.

Waveform Conditioning Vis filtrácia a použitie oknových funkcií na waveform.

Waveform Measurements Vis zistenie parametrov v časovej aj frekvenčnej oblasti pre waveform (DC, RMS, Tone Frequency/Amplitude/ Phase, Harmonic Distortion, SINAD, and Averaged FFT Measurements).

Windows Vis aplikácia oknových funkcií.

Waveform Generation Vis - deterministické signály

- **Basic Function Generator** Creates an output waveform based on signal type.
- **Basic Multitone with Amplitudes** Generates a waveform that is the sum of integer cycle sine tones.
- **Basic Multitone** Generates a waveform that is the sum of integer cycle sine tones.
- **Formula Waveform** Creates an output waveform using a formula string to specify the time function to be used.
- **Multitone Generator** Generates a waveform that is the sum of integer cycle sine tones.
- **Sawtooth Waveform** Generates a waveform containing a sawtooth wave.
- **Simulate Arbitrary Signal** Simulates a signal that you define.
- **Simulate Signal** Simulates a sine wave, square wave, triangle wave, sawtooth wave, or noise signal.
- **Sine Waveform** Generates a waveform containing a sine wave.
- **Square Waveform** Generates a waveform containing a square wave.
- **Tones and Noise Waveform** Generates a waveform composed of a sum of sine tones, noise, and DC offset.
- **Triangle Waveform** Generates a waveform containing a triangle wave.

Waveform Generation Vis - šum

- **Bernoulli Noise Waveform** Generates a pseudorandom pattern of ones and zeroes. Each element of signal out is computed by flipping a coin weighted by one probability.
- **Binomial Noise Waveform** Generates a binomially-distributed pseudorandom pattern whose values are the number of occurrences of an event given the probability of that event occurring and the number of trials.
- **Gamma Noise Waveform** Generates a pseudorandom pattern of values which are the waiting times to the order number event of a unit mean Poisson process.
- **Gaussian White Noise** Waveform Generates a Gaussian distributed pseudorandom pattern whose statistical profile is (0,s), where s is the absolute value of the specified standard deviation.
- **Inverse f Noise Waveform** Generates a continuous noise waveform with a power spectral density that is inversely proportional to frequency over a specified frequency range.
- **MLS Sequence Waveform** Generates a maximum length sequence of ones and zeroes using a modulo-2 primitive polynomial of order polynomial order.
- **Uniform White Noise Waveform** Generates a uniformly distributed pseudorandom pattern whose values are in the range [–*a*:*a*], where *a* is the absolute value of amplitude.
- **Periodic Random Noise Waveform** Generates a waveform containing periodic random noise (PRN).
- **Poisson Noise Waveform** Generates a pseudorandom sequence of values which are the number of discrete events occurring in the interval specified by mean of a unit rate Poisson process.

Signal Generation Vis - deterministické signály

Sine Pattern Generates an array containing a sinusoidal pattern.

Sine Wave Generates an array containing a sine wave.Square WaveGenerates an array containing a square wave.

Impulse Pattern Generates an array containing an impulse pattern.

Pulse Pattern Generates an array containing a pulse pattern.

Pulse Train Generates an array that concatenates a series of pulses according to the Prototype Pulse. This VI constructs the Pulse Train output by the specified interpolation method.

Ramp Pattern Generates an array containing a ramp pattern.

Sawtooth Wave Generates an array containing a sawtooth wave.

Triangle Pattern Generates an array that contains a triangle pattern.

Triangle Wave Generates an array containing a triangle wave.

Sinc Pattern Generates an array containing a sinc pattern.

Periodic Sinc Pattern Generates an array containing a periodic sinc pattern.

Tones and Noise Generates an array composed of a sum of sine tones, noise, and DC offset.

Signal Generator by Duration Generates a signal with a shape given by the signal type.

Arbitrary Wave Generates an array containing an arbitrary wave.

Chirp Pattern Generates an array containing a chirp pattern.

Signal Generation Vis - šum

Bernoulli NoiseGenerates a pseudorandom pattern of ones and zeroes. LabVIEW computes each element of bernoulli noise as if flipping a coin weighted by ones probability.

Binary MLSGenerates a maximum length sequence of ones and zeroes using a modulo-2 primitive polynomial of order polynomial order.

Binomial NoiseGenerates a binomially-distributed, pseudorandom pattern whose values are the number of occurrences of an event, given the probability of that event occurring and the number of trials.

Gamma NoiseGenerates a pseudorandom pattern of values that are the waiting times to the order number event of a unit mean Poisson process.

Gaussian Modulated Sine PatternGenerates an array that contains a Gaussian-modulated sinusoidal pattern.

Gaussian MonopulseGenerates an array that contains a Gaussian monopulse.

Gaussian White NoiseGenerates a Gaussian-distributed, pseudorandom pattern whose statistical profile is (μ,sigma) = (0,*s*), where *s* is standard deviation.

Poisson NoiseGenerates a pseudorandom sequence of values that are the number of discrete events occurring in a given interval, specified by mean, of a unit rate Poisson process.

Periodic Random Noise Generates an array containing periodic random noise (PRN).

Uniform White Noise Generates a uniformly distributed, pseudorandom pattern whose values are in the range [-*a*:*a*], where *a* is the absolute value of amplitude.

Signal Operation Vis – I.

AC & DC Estimator Estimates the AC and DC levels of the input Signal.

AutoCorrelation Computes the autocorrelation of the input sequence X. The data type you wire to the X input determines the polymorphic instance to use.

Convolution and Correlation Performs convolution, deconvolution, or correlation on the input signals.

Convolution Computes the convolution of the input sequences X and Y. The data type you wire to the X input determines the polymorphic instance to use.

Deconvolution Computes the deconvolution of the input sequences X * Y and Y.

CrossCorrelation Computes the cross correlation of the input sequences X and Y. The data type you wire to the X input determines the polymorphic instance to use.

Normalize Normalizes the input vector or matrix using its statistical profile (μ,s), where μ is the mean and s is the standard deviation, to obtain a Normalized Vector or Normalized Matrix whose statistical profile is (0,1).

Quick Scale Determines the maximum absolute value of the input X and then scales X using this value.

Scale Removes the offset of an input signal X and then scales the result so that the output sequence is in the range [–1:1].

Scaling and Mapping Changes the amplitude of a signal by scaling or mapping the signal.

Y[i]=Clip{X[i]} Clips the elements of Input Array to within the bounds specified by upper limit and lower limit.

Unit Vector Finds the norm of the Input Vector and obtains its corresponding Unit Vector by normalizing the original Input Vector with its norm.

Unwrap Phase Unwraps the Phase array by eliminating discontinuities whose absolute values exceed pi.

Peak Detector Finds the location, amplitude, and second derivative of peaks or valleys in the input signal.

Threshold Peak Detector Analyzes the input sequence X for valid peaks and keeps a count of the number of peaks encountered and a record of Indices, which locates the points that exceed the threshold in a valid peak.

Signal Operation Vis – II.

Decimate (continuous) Continuously decimates the input sequence X by the decimating factor and the averaging Boolean control. The data type you wire to the X input determines the polymorphic instance to use.

Decimate (single shot) Decimates the input sequence X by the decimating factor and the averaging Boolean control. The data type you wire to the X input determines the polymorphic instance to use.

Rational Resample Resamples the input signal X by interpolating X, passing the interpolated signal through an FIR filter, and decimating the filtered signal. The data type you wire to the X input determines the polymorphic instance to use.

Resample (constant to constant) Resamples input signal X according to delay and dt using an FIR filter implementation. The data type you wire to the X input determines the polymorphic instance to use.

Resample (constant to variable) Resamples input signal X according to Time using an FIR filter implementation. The data type you wire to the X input determines the polymorphic instance to use.

Upsample Inserts zeros in input sequence X by upsampling factor. The data type you wire to the X input determines the polymorphic instance to use.

Zero Padder Resizes the input sequence Input Array to the next higher valid power of 2, sets the new trailing elements of the sequence to zero, and leaves the first n elements unchanged, where n is the number of samples in the input sequence. The data type you wire to the Input Array input determines the polymorphic instance to use.

Riffle Riffles the elements in the input array X. The data type you wire to the X input determines the polymorphic instance to use.

Y[i]=X[i-n] Shifts the elements in the Input Array by the specified number of shifts: n.

Waveform Conditioning VIs

Align and Resample Performs an alignment of signals by changing the start time or performs a resampling of signals by changing the time delta. This Express VI returns the adjusted signals.

Align Waveforms (continuous) Performs element-wise alignment of waveforms and returns the aligned waveforms. The data types you wire to the waveform inputs determine the polymorphic instance to use.

Align Waveforms (single shot) Performs element-wise alignment of two waveforms and returns the aligned waveforms. The data types you wire to the waveform inputs determine the polymorphic instance to use.

Continuous Convolution (FIR) Convolves single or multiple waveforms and one or more kernels with state, allowing subsequent calls to be processed in a continuous manner. If you are convolving multiple waveforms, the VI maintains separate convolution states for each waveform.

Digital FIR Filter Filters signals in either single or multiple waveforms. If you are filtering multiple waveforms, the VI maintains separate filter states for each waveform. The data types you wire to the signal in and FIR filter specifications inputs determine the polymorphic instance to use.

Digital IIR Filter Filters signals in either single or multiple waveforms. If you are filtering multiple waveforms, the VI maintains separate filter states for each waveform. The data types you wire to the signal in and IIR filter specifications inputs determine the polymorphic instance to use.

Filter Processes signals through filters and windows.

Resample Waveforms (continuous) Resamples an input waveform according to the user-defined values for t0 and dt. The data type you wire to the waveform in input determines the polymorphic instance to use.

Resample Waveforms (single shot) Resamples input waveforms or data according to the user-defined t0 and dt values. The data type you wire to the waveform or data input determines the polymorphic instance to use.

Scaled Window Applies a scaled window to the time-domain signal and outputs window constants for further analysis. The data type you wire to the signal in input determines the polymorphic instance to use.

Trigger and Gate Uses triggering to extract a segment out of a signal. The trigger conditions can be based on a start or stop trigger threshold or can be static. When a trigger condition is static, the trigger occurs immediately and this Express VI returns a predefined number of samples.

Waveform Measurements Vis – I.

FFT Power Spectral Density Computes the averaged power spectral density of **time signal**. The data type you wire to the **time signal** input determines the polymorphic instance to use.

FFT Power Spectrum Computes the averaged auto power spectrum of **time signal**. The data type you wire to the **time signal** input determines the polymorphic instance to use.

FFT Spectrum (Mag-Phase) Computes the averaged FFT spectrum of **time signal**. FFT results are returned as **magnitude** and **phase**. The data type you wire to the **time signal** input determines the polymorphic instance to use.

FFT Spectrum (Real-Im) Computes the averaged FFT spectrum of **time signal**. FFT results are returned as real and imaginary parts. The data type you wire to the **time signal** input determines the polymorphic instance to use.

Frequency Cross Spectrum (Mag-Phase) Computes the averaged cross power spectrum of the input signals. Results are returned as **magnitude** and **phase**.

Cross Spectrum (Real-Im) Computes the averaged cross power spectrum of the input signals. Results are returned as real and imaginary parts.

Amplitude and Level Measurements Performs voltage measurements on a signal.

Averaged DC-RMS Calculates the DC and/or the RMS values of an input waveform or array of waveforms. The data type you wire to the **signal in** input determines the polymorphic instance to use.

Basic Averaged DC-RMS Takes a waveform or an array of waveforms in, applies a window to the signal, and averages the DC and RMS values calculated from the windowed signal with the previous DC and RMS values according to the **averaging type** input. The data type you wire to the **signal in** input determines the polymorphic instance to use.

Cycle Average and RMS Returns the average and RMS levels of a selected cycle of a periodic waveform or an array of periodic waveforms. The data type you wire to the **signal in** input determines the polymorphic instance to use.

Waveform Measurements Vis – II.

Amplitude and Levels Returns the **amplitude**, **high state level**, and **low state level** of a waveform or an array of waveforms. The data type you wire to the **signal in** input determines the polymorphic instance to use.

Pulse MeasurementsAccepts a periodic waveform or an array of periodic waveforms and returns the **period**, **pulse duration** (pulse width), **duty cycle** (duty factor), and **pulse center** of a selected pulse and period. The data type you wire to the **signal in** input determines the polymorphic instance to use.

Extract Multiple Tone InformationReturns the frequency, amplitude, and phase for each signal tone whose amplitude exceeds a specified **threshold**. The data type you wire to the **time signal in** input determines the polymorphic instance to use. If you wire an array of waveforms to **time signal in**, LabVIEW selects the Extract Multiple Tone Information N Chan instance by default.

Extract Single Tone InformationTakes a signal in, finds the single tone with the highest amplitude or searches a specified frequency range, and returns the single tone frequency, amplitude, and phase. The data type you wire to the **time signal in** input determines the polymorphic instance to use.

Tone MeasurementsFinds the single tone with the highest amplitude or searches a specified frequency range to find the single tone with the highest amplitude. You also can find the frequency and phase for a single tone.

Response Function (Mag-Phase)Computes the frequency response and the coherence based on the input signals. Results are returned as **magnitude**, **phase**, and **coherence**.

Frequency Response Function (Real-Im)Computes the frequency response and the coherence based on the input signals. Results are returned as **real part**, **imaginary part**, and **coherence**.

Waveform Measurements Vis – III.

Harmonic Distortion AnalyzerTakes a signal in and performs a full harmonic analysis, including measuring the fundamental frequency tone and harmonics, and returning the fundamental frequency, all harmonic amplitude levels, and the total harmonic distortion (THD). The data type you wire to the **signal in** input determines the polymorphic instance to use.

SINAD AnalyzerTakes a signal in and performs a full Signal in Noise and Distortion (SINAD) analysis, including measuring the fundamental frequency tone and returning the fundamental frequency and SINAD level in dB. The data type you wire to the **signal in** input determines the polymorphic instance to use.

Spectral MeasurementsPerforms FFT-based spectral measurements, such as the averaged magnitude spectrum, power spectrum, and phase spectrum on a signal.

Distortion MeasurementsPerforms distortion measurements on a signal, such as tone analysis, total harmonic distortion (THD), and signal in noise and distortion (SINAD).

Timing and Transition MeasurementsPerforms timing and transition measurements, such as frequency, period, or duty cycle, on a signal, usually a pulse.

Dual Channel Spectral MeasurementMeasures the frequency response of the input signals and the coherence based on the current and previous input signals. This Express VI returns results such as **Magnitude**, **Phase**, **Coherence**, **Real**, and **Imaginary**.

Transition MeasurementsAccepts an input signal of a single waveform or an array of waveforms and measures the transition duration (rise or fall time), slew rate, preshoot, and overshoot of a selected positive or negative transition in each waveform. The data type you wire to the **signal in** input determines the polymorphic instance to use.

Subpalette:

Waveform Monitoring VIsUse the Waveform Monitoring VIs to analyze the waveforms for trigger points, to search for peaks, and to perform limit mask testing.

Spectral Analysis Vis I.

Amplitude and Phase Spectrum Computes the single-sided, scaled (korigované) amplitude spectrum of a real-valued time-domain signal and returns the amplitude spectrum as magnitude and phase.

Auto Power Spectrum Computes the single-sided, scaled, auto power spectrum of a time-domain signal.

Buneman Frequency Estimator Estimates the frequency of a given sine wave with an unknown wavelength.

Cross Power SpectrumComputes the single-sided, scaled, cross power spectrum of two real-time signals.

Cross Power Computes the cross power spectrum, **Sxy**, of the input signals **X** and **Y**. The data type you wire to the **X** input determines the polymorphic instance to use.

Power & Frequency Estimate Computes the estimated power and frequency around a peak in the power spectrum of a time-domain signal.

Power Spectrum Computes the **Power Spectrum**, Sxx, of the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Spectral Analysis Vis – II.

Spectrum Unit Conversion Converts either the power, amplitude, or gain (amplitude ratio) spectrum to alternate formats including Log (decibel and dbm) and spectral density.

STFT Spectrograms Computes the signal energy distribution in the joint time-frequency domain, using the Short-Time Fourier Transform (STFT) algorithm.

Unevenly Sampled Signal Spectrum Calculates the power spectrum of a signal that is unevenly spaced in time.

WVD Spectrogram Computes the signal energy distribution in the joint time-frequency domain using the Wigner-Ville distribution algorithm.

Transforms VIs - I.

FFT Computes the Fast Fourier Transform (FFT) of the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Inverse FFT Computes the inverse Discrete Fourier Transform (DFT) of the input sequence **FFT {X}**. You must manually select the polymorphic instance you want to use.

Inverse FHT Computes the inverse fast Hartley transform of the input sequence **X**.

Laplace Transform Real Computes the real Laplace transform of the input sequence **X**.

Walsh Hadamard Inverse Computes the inverse of the real Walsh Hadamard transform of the input sequence **X**.

Walsh Hadamard Computes the real Walsh Hadamard transform of the input sequence **X**.

Wavelet Transform Daubechies4 Inverse Computes the inverse of the wavelet transform based on the Daubechies4 function of the input sequence **X**.

Wavelet Transform Daubechies4 Computes the wavelet transform based on the Daubechies4 function of the input sequence **X**.

Transforms VIs - II.

Chirp Z TransformComputes the Chirp-Z Transform of the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

DCTComputes the Discrete Cosine Transform (DCT) of the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

DSTComputes the Discrete Sine Transform (DST) of the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Fast Hilbert TransformComputes the fast Hilbert transform of the input sequence **X**.

FHTComputes the fast Hartley transform (FHT) of the input sequence **X**.

Inverse Chirp Z TransformComputes the inverse Chirp-Z Transform of the input sequence **Chirp-Z {X}**

Inverse DCTComputes the inverse Discrete Cosine Transform (DCT) of the input sequence **DCT {X}**. The data type you wire to the **DCT {X}** input determines the polymorphic instance to use.

Inverse DSTComputes the inverse Discrete Sine Transform (DST) of the input sequence **DST {X}**. The data type you wire to the **DST {X}** input determines the polymorphic instance to use.

Inverse Fast Hilbert TransformComputes the inverse fast Hilbert transform of the input sequence **X** using Fourier identities.

Filters VIs – I. (FIR)

FIR Windowed Filter Filters the input data sequence, **X**, using the set of windowed FIR filter coefficients specified by the **sampling freq: fs**, **low cutoff freq: fl**, **high cutoff freq: fh**, and number of **taps**. The data type you wire to the **X** input determines the polymorphic instance to use.

Equi-Ripple BandPass Generates a bandpass FIR filter with equi-ripple characteristics using the Parks-McClellan algorithm and the **higher pass freq**, **lower pass freq**, **# of taps**, **lower stop freq**, **higher stop freq**, and **sampling freq: fs**. The Equi-Ripple BandPass VI then applies a linear-phase, bandpass filter to the input sequence **X** using the Convolution VI to obtain **Filtered X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Equi-Ripple BandStop Generates a bandstop FIR digital filter with equi-ripple characteristics using the Parks-McClellan algorithm and **higher pass freq**, **lower pass freq**, **# of taps**, **lower stop freq**, **higher stop freq**, and **sampling freq: fs**. The Equi-Ripple BandStop VI then applies a linear-phase, bandstop filter to the input sequence **X** using the Convolution VI to obtain **Filtered X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Equi-Ripple HighPass Generates a highpass FIR filter with equi-ripple characteristics using the Parks-McClellan algorithm and the **# of taps**, **stop freq**, **high freq**, and **sampling freq: fs**. The Equi-Ripple HighPass VI then applies a linear-phase, highpass filter to the input sequence **X** using the Convolution VI to obtain **Filtered X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Equi-Ripple LowPass Generates a lowpass FIR filter with equi-ripple characteristics using the Parks-McClellan algorithm and the **# of taps**, **pass freq**, **stop freq**, and **sampling freq: fs**. The Equi-Ripple LowPass VI then applies a linear-phase, lowpass filter to the input sequence **X** using the Convolution VI to obtain **Filtered X**. The data type you wire to the **X** input determines the polymorphic instance to use.

FIR Filter with I.C. Filters the input sequence **X** using the direct-form FIR filter specified by **FIR Coefficients**. You can use this VI to process blocks of continuous data. The data type you wire to the **X** input determines the polymorphic instance to use.

Filters VIs – II.

Bessel FilterGenerates a digital Bessel filter by calling the Bessel Coefficients VI. The data type you wire to the **X** input determines the polymorphic instance to use.Butterworth FilterGenerates a digital Butterworth filter by calling the Butterworth Coefficients VI. The data type you wire to the **X** input determines the polymorphic instance to use.

Chebyshev FilterGenerates a digital, Chebyshev filter by calling the Chebyshev Coefficients VI. The data type you wire to the **X** input determines the polymorphic instance to use.

Elliptic FilterGenerates a digital, elliptic filter by calling the Elliptic Coefficients VI. The data type you wire to the **X** input determines the polymorphic instance to use.

Inverse Chebyshev FilterGenerates a digital, Chebyshev II filter by calling the Inv Chebyshev Coefficients VI. The data type you wire to the **X** input determines the polymorphic instance to use.

Inverse f FilterDesigns and implements an IIR filter whose magnitude-squared response is inversely proportional to frequency over a specified frequency range. This inverse-f filter is typically used to colorize spectrally flat, or white, noise. The data type you wire to the **X** input determines the polymorphic instance to use.

Median FilterApplies a median filter of rank to the input sequence **X**, where rank is **right rank** if **right rank** is greater than zero, or **left rank** if **right rank** is less than zero.

Savitzky-Golay FilterFilters the input data sequence **X** using a Savitzky-Golay FIR smoothing filter. The data type you wire to the **X** input determines the polymorphic instance to use.

Zero Phase FilterApplies a zero phase filter to an input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Subpalette

Advanced FIR Filtering VIsUse the Advanced FIR Filtering VIs to implement advanced FIR filters.

Advanced IIR Filtering VIsUse the Advanced IIR Filtering VIs to implement advanced IIR filters.

Windows Vis – I.

Aplikácia oknových funkcií

Blackman-Harris WindowApplies a three-term, Blackman-Harris window to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use

Blackman-Nuttall WindowApplies a Blackman-Nuttall window to the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Blackman WindowApplies a Blackman window to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Bohman WindowApplies a Bohman window to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Chebyshev WindowApplies an asymmetrical Dolph-Chebyshev window to the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Cosine Tapered WindowApplies a cosine tapered window to the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Exact Blackman WindowApplies an Exact Blackman window to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Exponential WindowApplies an exponential window to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Flat Top WindowApplies a flat top window to the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Force WindowApplies a force window to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Gaussian WindowApplies an asymmetrical Gaussian window to the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Windows Vis – II.

General Cosine WindowApplies a general cosine window to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Hamming Window Applies a Hamming window to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Hanning Window Applies a Hanning window to the input signal X. The data type you wire to the X input determines the polymorphic instance to use.

Kaiser-Bessel Window Applies a Kaiser-Bessel window to the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Modified Bartlett-Hanning Window Applies a modified Bartlett-Hanning window to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Parzen Window Applies a Parzen window to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Scaled Time Domain Window Applies a scaled window to the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Symmetric Window Applies a symmetric window to the input sequence **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Triangle Window Applies a triangular window (Bartlett Window) to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use.

Welch Window Applies a Welch window to the input signal **X**. The data type you wire to the **X** input determines the polymorphic instance to use

Window Properties Computes the coherent gain and equivalent noise bandwidth of a window.

Point By Point VIs

Filters PtByPt Vis Use the Filters PtByPt VIs to implement IIR, FIR, and nonlinear filters.

Fitting PtByPt Vis Use the Fitting PtByPt VIs to fit data to a model and perform interpolation and extrapolation.

Geometry PtByPt Vis Use the Geometry PtByPt VIs to perform geometry operations.

Integral & Differential PtByPt Vis Use the Integral & Differential PtByPt VIs to perform integration and differentiation.

Interpolation PtByPt Vis Use the Interpolation PtByPt VIs to perform interpolation and extrapolation.

Linear Algebra PtByPt Vis Use the Linear Algebra PtByPt VIs to perform matrix- and vector-related computations and analysis.

Other Functions PtByPt Vis Use the Other Functions PtByPt VIs to perform miscellaneous point-by-point operations.

Polynomial PtByPt Vis Use the Polynomial PtByPt VIs to perform calculations and evaluations with polynomials.

Probability & Statistics PtByPt Vis Use the Probability & Statistics PtByPt VIs to perform probability, descriptive statistics, and analysis of variance operations.

Signal Generation PtByPt Vis Use the Signal Generation PtByPt VIs to generate one-dimensional arrays with specific waveform patterns.

Signal Operation PtByPt Vis Use the Signal Operation PtByPt VIs to perform common one- and two-dimensional numerical analysis.

Spectral Analysis PtByPt Vis Use the Spectral Analysis PtByPt VIs to implement common transforms used in mathematics and signal processing.

Transforms PtByPt Vis Use the Transforms PtByPt VIs to compute transforms.

Vybrané operácie s poľami I.

Array Constant – Vytvorí konštantu vo forme poľa

Initialize Array – vytvorí n–rozmerné pole, všetky prvky majú zadanú konštantnú hodnotu.

Array Max & Min - nájde maximum a minimum.

Array Size – vráti rozmery poľa

Array SubsetReturns a portion of **array** starting at **index** and containing **length** elements.

Build Array spojí viaceré polia tak, že na koniec prvého pripojí začiatok druhého atď.

Delete From Array – zmaže časť poľa a vráti samostne zmazanú časť a zvyšok.

Index Array – vráti prvok alebo časť poľa

Decimate 1D Array – rozdelí prvky do niekoľkých polí

Insert Into Array - vloží na zvolené miesto prvok alebo pole.

Replace Array Subset nahradí prvok alebo skupinu prvkov v poli novými hodnotami

Rotate 1D Array – urobí rotáciu prvkov v poli

Split 1D Array – rozdelí pole na dve časti podľa indexu

Vybrané operácie s poľami II.

Interleave 1D ArraysInterleaves corresponding elements from the input arrays into a single output array.

Interpolate 1D ArrayLinearly interpolates a decimal **y value** from an **array of numbers or points** using a **fractional index or x** value.

Matrix To ArrayConverts a matrix of elements to an array of elements of the same data type. The data type you wire to the **Real Matrix** input determines the polymorphic instance to use.

Reshape ArrayChanges the dimensions of an array according to the values of **dimension size 0..m-1**.

Reverse 1D ArrayReverses the order of the elements in **array**.

Search 1D ArraySearches for an **element** in a **1D array** starting at **start index**. Because the search is linear, you need not sort the array before calling this function. LabVIEW stops searching as soon as the element is found.

Sort 1D ArrayReturns a sorted version of **array** with the elements arranged in ascending order.

Threshold 1D ArrayInterpolates points in a 1D array representing a 2D non-descending graph. This function compares **threshold y** to the values in **array of numbers or points** starting at **start index** until it finds a pair of consecutive elements such that **threshold y** is greater than the value of the first element and less than or equal to the value of the second element.

Transpose 2D ArrayRearranges the elements of **2D array** such that **2D array**[i,j] becomes **transposed array**[j,i].

Array To ClusterConverts a 1D array to a cluster of elements of the same type as the array elements. Right-click the function and select Cluster Size from the shortcut menu to set the number of elements in the cluster.

Array To MatrixConverts an array to a matrix of elements of the same type as the array elements. The data type you wire to the **Real 2D Array** input determines the polymorphic instance to use.

Cluster To ArrayConverts a cluster of elements of the same data type to a 1D array of elements of the same data type.

Waveform VIs and Functions

Analog to Digital – prevedie analógový signál na číslicový = digitalizuje

Build Waveform – vytvorí waveform z poľa vzoriek a podmienok zosnímania

Digital to Analog - Converts a digital waveform or digital data to **analog waveform**. You must manually select the polymorphic instance you want to use.

Get Waveform Components Returns the analog waveform you specify. You specify components by clicking on the center of the output terminal and selecting the component you want.

Get Waveform Subset Retrieves a subset of a waveform at a specified time or index. You must manually select the polymorphic instance you want to use.

Get Waveform Time Array Creates an array of waveform time stamps. Each element in the array is the time stamp for each data value in the waveform. The data type you wire to the **waveform in** input determines the polymorphic instance to use.

...