NVIDIA Performance Primitives (NPP)

Version 9.0

August 18, 2017

Contents

Chapter 1

NVIDIA Performance Primitives

Note: The static NPP libraries depend on a common thread abstraction layer library called cuLIBOS (libculibos.a) that is now distributed as part of the toolkit. Consequently, cuLIBOS must be provided to the linker when the static library is being linked against. To minimize library loading and CUDA runtime startup times it is recommended to use the static library(s) whenever possible. To improve loading and runtime performance when using dynamic libraries, NPP 9.0 has deprecated the full sized nppi library and replaced it with a full set of nppi sub-libraries. Linking to only the sub-libraries that contain functions that your application uses can significantly improve load time and runtime startup performance. Some nppi functions make calls to other nppi and/or npps functions internally so you may need to link to a few extra libraries depending on what function calls your application makes. The nppi sub-libraries are split into sections corresponding to the way that nppi header files are split. This list of sub-libraries is as follows:

```
nppial  arithmetic and logical operation functions in nppi_arithmetic_and_logical_operations.h
nppicc  color conversion and sampling functions in nppi_color_conversion.h
nppicom JPEG compression and decompression functions in nppi_compression_functions.h
nppidei data exchange and initialization functions in nppi_data_exchange_and_initialization.h
nppif   filtering and computer vision functions in nppi_filter_functions.h
nppig   geometry transformation functions found in nppi_geometry_transforms.h
nppim   morphological operation functions found in nppi_morphological_operations.h
nppist  statistics and linear transform in nppi_statistics_functions.h and nppi_linear_transforms.
nppisu  memory support functions in nppi_support_functions.h
nppitc  threshold and compare operation functions in nppi_threshold_and_compare_operations.h
```

For example, on Linux, to compile a small application foo using NPP against the dynamic library, the following command can be used:

```
nvcc foo.c  -lnppi  -o foo
```

Whereas to compile against the static NPP library, the following command has to be used:

```
nvcc foo.c  -lnppi_static -lculibos -o foo
```

It is also possible to use the native host C++ compiler. Depending on the host operating system, some additional libraries like pthread or dl might be needed on the linking line. The following command on Linux is suggested:

```
g++ foo.c  -lnppi_static -lculibos -lcudart_static -lpthread -ldl
-I <cuda-toolkit-path>/include -L <cuda-toolkit-path>/lib64 -o foo
```

NPP is a stateless API, as of NPP 6.5 the ONLY state that NPP remembers between function calls is the current stream ID, i.e. the stream ID that was set in the most recent nppSetStream call and a few bits of device specific information about that stream. The default stream ID is 0. If an application intends to use NPP with multiple streams then it is the responsibility of the application to call nppSetStream whenever it wishes to change stream IDs. Several NPP functions may call other NPP functions internally to complete their functionality. For this reason it is recommended that cudaDeviceSynchronize (or at least cudaStreamSynchronize) be called before making an nppSetStream call to change to a new stream ID. This will insure that any internal function calls that have not yet occurred will be completed using the current stream ID before it changes to a new ID. Calling cudaDeviceSynchronize frequently call kill performance so minimizing the frequency of these calls is critical for good performance. It is not necessary to call cudaDeviceSynchronize for stream management while the same stream ID is used for multiple NPP calls. All NPP functions should be thread safe except for the following functions:

```
nppiDCTQuantFwd8x8LS_JPEG_8u16s_C1R

nppiDCTQuantInv8x8LS_JPEG_16s8u_C1R
```

1.1 What is NPP?

NVIDIA NPP is a library of functions for performing CUDA accelerated processing. The initial set of functionality in the library focuses on imaging and video processing and is widely applicable for developers in these areas. NPP will evolve over time to encompass more of the compute heavy tasks in a variety of problem domains. The NPP library is written to maximize flexibility, while maintaining high performance.

NPP can be used in one of two ways:

- A stand-alone library for adding GPU acceleration to an application with minimal effort. Using this route allows developers to add GPU acceleration to their applications in a matter of hours.
- A cooperative library for interoperating with a developer's GPU code efficiently.

Either route allows developers to harness the massive compute resources of NVIDIA GPUs, while simultaneously reducing development times.

1.2 Documentation

- General API Conventions
- Signal-Processing Specific API Conventions
- Imaging-Processing Specific API Conventions

1.3 Technical Specifications

Supported Platforms:

- Microsoft Windows 7, 8, and 10 (64-bit and 32-bit)
- Microsoft Windows Vista (64-bit and 32-bit)
- Linux (Centos, Ubuntu, and several others) (64-bit and 32-bit)
- Mac OS X (64-bit)
- Android on Arm (32-bit and 64-bit)

1.4 Files

NPP is comprises the following files:

1.4.1 Header Files

- nppdefs.h
- nppcore.h
- nppi::h
- npps::h
- nppversion.h
- npp::h

All those header files are located in the CUDA Toolkit's

```
/include/
```

directory.

1.4.2 Library Files

Starting with Version 5.5 NPP's functionality is now split up into 3 distinct library groups:

- A core library (NPPC) containing basic functionality from the npp.h header files as well as functionality shared by the other two libraries.
- The image processing library NPPI. Any functions from the nppi.h header file (or the various header files named "nppi_xxx.h" are bundled into the NPPI library.
- The signal processing library NPPS. Any function from the npps.h header file (or the various header files named "npps_xxx.h" are bundled into the NPPS library.

On the Windows platform the NPP stub libraries are found in the CUDA Toolkit's library directory:

```
/lib/nppc.lib
```

```
/lib/nppial.lib
```

```
/lib/nppicc.lib
```

```
/lib/nppicom.lib
```

```
/lib/nppidei.lib
```

```
/lib/nppif.lib
```

```
/lib/nppig.lib
```

```
/lib/nppim.lib

/lib/nppist.lib

/lib/nppisu.lib

/lib/nppitc.lib

/lib/npps.lib
```

The matching DLLs are located in the CUDA Toolkit's binary directory. Example

```
/bin/nppial64_90_<build_no>.dll      // Dynamic image-processing library for 64-bit Windows.
```

On Linux and Mac platforms the dynamic libraries are located in the lib directory

```
/lib/libnppc.so.9.0.<build_no>   // NPP dynamic core library for Linux

 /lib/libnpps.9.0.dylib  // NPP dynamic signal processing library for Mac
```

1.5 Supported NVIDIA Hardware

NPP runs on all CUDA capable NVIDIA hardware. For details please see http://www.nvidia.com/object/cuda_learn_products.html

Chapter 2

General API Conventions

2.1 Memory Management

The design of all the NPP functions follows the same guidelines as other NVIDIA CUDA libraries like cuFFT and cuBLAS. That is that all pointer arguments in those APIs are device pointers.

This convention enables the individual developer to make smart choices about memory management that minimize the number of memory transfers. It also allows the user the maximum flexibility regarding which of the various memory transfer mechanisms offered by the CUDA runtime is used, e.g. synchronous or asynchronous memory transfers, zero-copy and pinned memory, etc.

The most basic steps involved in using NPP for processing data is as follows:

1. Transfer input data from the host to device using

   ```
   cudaMemCpy(...)
   ```

2. Process data using one or several NPP functions or custom CUDA kernels
3. Transfer the result data from the device to the host using

   ```
   cudaMemCpy(...)
   ```

2.1.1 Scratch Buffer and Host Pointer

Some primitives of NPP require additional device memory buffers (scratch buffers) for calculations, e.g. signal and image reductions (Sum, Max, Min, MinMax, etc.). In order to give the NPP user maximum control regarding memory allocations and performance, it is the user's responsibility to allocate and delete those temporary buffers. For one this has the benefit that the library will not allocate memory unbeknownst to the user. It also allows developers who invoke the same primitive repeatedly to allocate the scratch only once, improving performance and potential device-memory fragmentation .

Scratch-buffer memory is unstructured and may be passed to the primitive in uninitialized form. This allows for reuse of the same scratch buffers with any primitive require scratch memory, as long as it is sufficiently sized.

The minimum scratch-buffer size for a given primitive (e.g. nppsSum_32f()) can be obtained by a companion function (e.g. nppsSumGetBufferSize_32f()). The buffer size is returned via a host pointer as allocation of the scratch-buffer is performed via CUDA runtime host code.

An example to invoke signal sum primitive and allocate and free the necessary scratch memory:

```
// pSrc, pSum, pDeviceBuffer are all device pointers.
Npp32f * pSrc;
Npp32f * pSum;
Npp8u * pDeviceBuffer;
int nLength = 1024;

// Allocate the device memroy.
cudaMalloc((void **)(&pSrc), sizeof(Npp32f) * nLength);
nppsSet_32f(1.0f, pSrc, nLength);
cudaMalloc((void **)(&pSum), sizeof(Npp32f) * 1);

// Compute the appropriate size of the scratch-memory buffer
int nBufferSize;
nppsSumGetBufferSize_32f(nLength, &nBufferSize);
// Allocate the scratch buffer
cudaMalloc((void **)(&pDeviceBuffer), nBufferSize);

// Call the primitive with the scratch buffer
```

```
nppsSum_32f(pSrc, nLength, pSum, pDeviceBuffer);
Npp32f nSumHost;
cudaMemcpy(&nSumHost, pSum, sizeof(Npp32f) * 1, cudaMemcpyDeviceToHost);
printf("sum = %f\n", nSumHost); // nSumHost = 1024.0f;

// Free the device memory
cudaFree(pSrc);
cudaFree(pDeviceBuffer);
cudaFree(pSum);
```

2.2 Function Naming

Since NPP is a C API and therefore does not allow for function overloading for different data-types the NPP naming convention addresses the need to differentiate between different flavors of the same algorithm or primitive function but for various data types. This disambiguation of different flavors of a primitive is done via a suffix containing data type and other disambiguating information.

In addition to the flavor suffix, all NPP functions are prefixed with by the letters "npp". Primitives belonging to NPP's image-processing module add the letter "i" to the npp prefix, i.e. are prefixed by "nppi". Similarly signal-processing primitives are prefixed with "npps".

The general naming scheme is:

npp<module info><PrimitiveName>_<data-type info>[_<additional flavor info>](<parameter list>)

The data-type information uses the same names as the Basic NPP Data Types. For example the data-type information "8u" would imply that the primitive operates on Npp8u data.

If a primitive consumes different type data from what it produces, both types will be listed in the order of consumed to produced data type.

Details about the "additional flavor information" is provided for each of the NPP modules, since each problem domain uses different flavor information suffixes.

2.3 Integer Result Scaling

NPP signal processing and imaging primitives often operate on integer data. This integer data is usually a fixed point fractional representation of some physical magnitue (e.g. luminance). Because of this fixed-point nature of the representation many numerical operations (e.g. addition or multiplication) tend to produce results exceeding the original fixed-point range if treated as regular integers.

In cases where the results exceed the original range, these functions clamp the result values back to the valid range. E.g. the maximum positive value for a 16-bit unsigned integer is 32767. A multiplication operation of 4 $*$ 10000 = 40000 would exceed this range. The result would be clamped to be 32767.

To avoid the level of lost information due to clamping most integer primitives allow for result scaling. Primitives with result scaling have the "Sfs" suffix in their name and provide a parameter "nScaleFactor" that controls the amount of scaling. Before the results of an operation are clamped to the valid output-data range by multiplying them with $2^{\text{-nScaleFactor}}$.

Example: The primitive nppsSqr_8u_Sfs() computes the square of 8-bit unsigned sample values in a signal (1D array of values). The maximum value of a 8-bit value is 255. The square of $255^2 = 65025$ which would be clamped to 255 if no result scaling is performed. In order to map the maximum value of 255 to 255 in the result, one would specify an integer result scaling factor of 8, i.e. multiply each result with $2^{-8} = \frac{1}{2^8} = \frac{1}{256}$. The final result for a signal value of 255 being squared and scaled would be:

$$255^2 \cdot 2^{-8} = 254.00390625$$

which would be rounded to a final result of 254.

A medium gray value of 128 would result in

$$128^2 * 2^{-8} = 64$$

2.4 Rounding Modes

Many NPP functions require converting floating-point values to integers. The NppRoundMode enum lists NPP's supported rounding modes. Not all primitives in NPP that perform rounding as part of their functionality allow the user to specify the round-mode used. Instead they use NPP's default rounding mode, which is NPP_RND_FINANCIAL.

2.4.1 Rounding Mode Parameter

A subset of NPP functions performing rounding as part of their functionality do allow the user to specify which rounding mode is used through a parameter of the NppRoundMode type.

Chapter 3

Signal-Processing Specific API Conventions

3.1 Signal Data

Signal data is passed to and from NPPS primitives via a pointer to the signal's data type.

The general idea behind this fairly low-level way of passing signal data is ease-of-adoption into existing software projects:

- Passing the data pointer rather than a higher- level signal struct allows for easy adoption by not requiring a specific signal representation (that could include total signal size offset, or other additional information). This avoids awkward packing and unpacking of signal data from the host application to an NPP specific signal representation.

3.1.1 Parameter Names for Signal Data

There are three general cases of image-data passing throughout NPP detailed in the following sections.

Those are signals consumed by the algorithm.

3.1.1.1 Source Signal Pointer

The source signal data is generally passed via a pointer named

```
pSrc
```

The source signal pointer is generally defined constant, enforcing that the primitive does not change any image data pointed to by that pointer. E.g.

```
nppsPrimitive_32s(const Npp32s * pSrc, ...)
```

In case the primitive consumes multiple signals as inputs the source pointers are numbered like this:

```
pSrc1, pScr2, ...
```

3.1.1.2 Destination Signal Pointer

The destination signal data is generally passed via a pointer named

```
pDst
```

In case the primitive consumes multiple signals as inputs the source pointers are numbered like this:

```
pDst1, pDst2, ...
```

3.1.1.3 In-Place Signal Pointer

In the case of in-place processing, source and destination are served by the same pointer and thus pointers to in-place signal data are called:

```
pSrcDst
```

3.1.2 Signal Data Alignment Requirements

NPP requires signal sample data to be naturally aligned, i.e. any pointer

```
NppType * p;
```

to a sample in a signal needs to fulfill:

```
assert(p % sizeof(p) == 0);
```

3.1.3 Signal Data Related Error Codes

All NPPI primitives operating on signal data validate the signal-data pointer for proper alignment and test that the point is not null.

Failed validation results in one of the following error codes being returned and the primitive not being executed:

- NPP_NULL_POINTER_ERROR is returned if the image-data pointer is 0 (NULL).
- NPP_ALIGNMENT_ERROR if the signal-data pointer address is not a multiple of the signal's data-type size.

3.2 Signal Length

The vast majority of NPPS functions take a

```
nLength
```

parameter that tells the primitive how many of the signal's samples starting from the given data pointer are to be processed.

3.2.1 Length Related Error Codes

All NPPS primitives taking a length parameter validate this input.

Failed validation results in the following error code being returned and the primitive not being executed:

- NPP_SIZE_ERROR is returned if the length is negative.

Chapter 4

Imaging-Processing Specific API Conventions

4.1 Function Naming

Image processing related functions use a number of suffixes to indicate various different flavors of a primitive beyond just different data types. The flavor suffix uses the following abbreviations:

- "A" if the image is a 4 channel image this indicates the result alpha channel is not affected by the primitive.
- "Cn" the image consists of n channel packed pixels, where n can be 1, 2, 3 or 4.
- "Pn" the image consists of n separate image planes, where n can be 1, 2, 3 or 4.
- "C" (following the channel information) indicates that the primitive only operates on one of the color channels, the "channel-of-interest". All other output channels are not affected by the primitive.
- "I" indicates that the primitive works "in-place". In this case the image-data pointer is usually named "pSrcDst" to indicate that the image data serves as source and destination at the same time.
- "M" indicates "masked operation". These types of primitives have an additional "mask image" as as input. Each pixel in the destination image corresponds to a pixel in the mask image. Only pixels with a corresponding non-zero mask pixel are being processed.
- "R" indicates the primitive operates only on a rectangular "region-of-interest" or "ROI". All ROI primitives take an additional input parameter of type NppiSize, which specifies the width and height of the rectangular region that the primitive should process. For details on how primitives operate on ROIs see: Region-of-Interest (ROI).
- "Sfs" indicates the result values are processed by fixed scaling and saturation before they're written out.

The suffixes above always appear in alphabetical order. E.g. a 4 channel primitive not affecting the alpha channel with masked operation, in place and with scaling/saturation and ROI would have the postfix: "AC4IMRSfs".

4.2 Image Data

Image data is passed to and from NPPI primitives via a pair of parameters:

1. A pointer to the image's underlying data type.
2. A line step in bytes (also sometimes called line stride).

The general idea behind this fairly low-level way of passing image data is ease-of-adoption into existing software projects:

- Passing a raw pointer to the underlying pixel data type, rather than structured (by color) channel pixel data allows usage of the function in a wide variety of situations avoiding risky type cast or expensive image data copies.
- Passing the data pointer and line step individually rather than a higher- level image struct again allows for easy adoption by not requiring a specific image representation and thus avoiding awkward packing and unpacking of image data from the host application to an NPP specific image representation.

4.2.1 Line Step

The line step (also called "line stride" or "row step") allows lines of oddly sized images to start on well-aligned addresses by adding a number of unused bytes at the ends of the lines. This type of line padding has been common practice in digital image processing for a long time and is not particular to GPU image processing.

The line step is the number of bytes in a line **including the padding.** An other way to interpret this number is to say that it is the number of bytes between the first pixel of successive rows in the image, or generally the number of bytes between two neighboring pixels in any column of pixels.

The general reason for the existence of the line step it is that uniformly aligned rows of pixel enable optimizations of memory-access patterns.

Even though all functions in NPP will work with arbitrarily aligned images, best performance can only be achieved with well aligned image data. Any image data allocated with the NPP image allocators or the 2D memory allocators in the CUDA runtime, is well aligned.

Particularly on older CUDA capable GPUs it is likely that the performance decrease for misaligned data is substantial (orders of magnitude).

All image data passed to NPPI primitives requires a line step to be provided. It is important to keep in mind that this line step is always specified in terms of bytes, not pixels.

4.2.2 Parameter Names for Image Data

There are three general cases of image-data passing throughout NPP detailed in the following sections.

4.2.2.1 Passing Source-Image Data

Those are images consumed by the algorithm.

4.2.2.1.1 Source-Image Pointer

The source image data is generally passed via a pointer named

```
pSrc
```

The source image pointer is generally defined constant, enforcing that the primitive does not change any image data pointed to by that pointer. E.g.

```
nppiPrimitive_32s_C1R(const Npp32s * pSrc, ...)
```

In case the primitive consumes multiple images as inputs the source pointers are numbered like this:

```
pSrc1, pScr2, ...
```

4.2.2.1.2 Source-Planar-Image Pointer Array

The planar source image data is generally passed via an array of pointers named

```
pSrc[]
```

The planar source image pointer array is generally defined a constant array of constant pointers, enforcing that the primitive does not change any image data pointed to by those pointers. E.g.

```
nppiPrimitive_8u_P3R(const Npp8u * const pSrc[3], ...)
```

Each pointer in the array points to a different image plane.

4.2.2.1.3 Source-Planar-Image Pointer

The multiple plane source image data is passed via a set of pointers named

```
pSrc1, pSrc2, ...
```

The planar source image pointer is generally defined as one of a set of constant pointers with each pointer pointing to a different input image plane.

4.2.2.1.4 Source-Image Line Step

The source image line step is the number of bytes between successive rows in the image. The source image line step parameter is

```
nSrcStep
```

or in the case of multiple source images

```
nSrcStep1, nSrcStep2, ...
```

4.2.2.1.5 Source-Planar-Image Line Step Array

The source planar image line step array is an array where each element of the array contains the number of bytes between successive rows for a particular plane in the input image. The source planar image line step array parameter is

```
rSrcStep[]
```

4.2.2.1.6 Source-Planar-Image Line Step

The source planar image line step is the number of bytes between successive rows in a particular plane of the multiplane input image. The source planar image line step parameter is

```
nSrcStep1, nSrcStep2, ...
```

4.2.2.2 Passing Destination-Image Data

Those are images produced by the algorithm.

4.2.2.2.1 Destination-Image Pointer

The destination image data is generally passed via a pointer named

```
pDst
```

In case the primitive generates multiple images as outputs the destination pointers are numbered like this:

```
pDst1, pDst2, ...
```

4.2.2.2.2 Destination-Planar-Image Pointer Array

The planar destination image data pointers are generally passed via an array of pointers named

```
pDst[]
```

Each pointer in the array points to a different image plane.

4.2.2.2.3 Destination-Planar-Image Pointer

The destination planar image data is generally passed via a pointer to each plane of a multiplane output image named

```
pDst1, pDst2, ...
```

4.2.2.2.4 Destination-Image Line Step

The destination image line step parameter is

```
nDstStep
```

or in the case of multiple destination images

```
nDstStep1, nDstStep2, ...
```

4.2.2.2.5 Destination-Planar-Image Line Step Array

The destination planar image line step array is an array where each element of the array contains the number of bytes between successive rows for a particular plane in the output image. The destination planar image line step array parameter is

```
rDstStep[]
```

4.2.2.2.6 Destination-Planar-Image Line Step

The destination planar image line step is the number of bytes between successive rows for a particular plane in a multiplane output image. The destination planar image line step parameter is

```
nDstStep1, nDstStep2, ...
```

4.2.2.3 Passing In-Place Image Data

4.2.2.3.1 In-Place Image Pointer

In the case of in-place processing, source and destination are served by the same pointer and thus pointers to in-place image data are called:

```
pSrcDst
```

4.2.2.3.2 In-Place-Image Line Step

The in-place line step parameter is

```
nSrcDstStep
```

4.2.2.4 Passing Mask-Image Data

Some image processing primitives have variants supporting Masked Operation.

4.2.2.4.1 Mask-Image Pointer

The mask-image data is generally passed via a pointer named

```
pMask
```

4.2.2.4.2 Mask-Image Line Step

The mask-image line step parameter is

```
nMaskStep
```

4.2.2.5 Passing Channel-of-Interest Data

Some image processing primitives support Channel-of-Interest API.

4.2.2.5.1 Channel_of_Interest Number

The channel-of-interest data is generally an integer (either 1, 2, or 3):

```
nCOI
```

4.2.3 Image Data Alignment Requirements

NPP requires pixel data to adhere to certain alignment constraints: For 2 and 4 channel images the following alignment requirement holds: data_pointer % (#channels $*$ sizeof(channel type)) == 0. E.g. a 4 channel image with underlying type Npp8u (8-bit unsigned) would require all pixels to fall on addresses that are multiples of 4 (4 channels $*$ 1 byte size).

As a logical consequence of all pixels being aligned to their natural size the image line steps of 2 and 4 channel images also need to be multiples of the pixel size.

1 and 3 channel images only require that pixel pointers are aligned to the underlying data type, i.e. pData % sizof(data type) == 0. And consequentially line steps are also held to this requirement.

4.2.4 Image Data Related Error Codes

All NPPI primitives operating on image data validate the image-data pointer for proper alignment and test that the point is not null. They also validate the line stride for proper alignment and guard against the step being less or equal to 0. Failed validation results in one of the following error codes being returnd and the primitive not being executed:

- NPP_STEP_ERROR is returned if the data step is 0 or negative.
- NPP_NOT_EVEN_STEP_ERROR is returned if the line step is not a multiple of the pixel size for 2 and 4 channel images.
- NPP_NULL_POINTER_ERROR is returned if the image-data pointer is 0 (NULL).
- NPP_ALIGNMENT_ERROR if the image-data pointer address is not a multiple of the pixel size for 2 and 4 channel images.

4.3 Region-of-Interest (ROI)

In practice processing a rectangular sub-region of an image is often more common than processing complete images. The vast majority of NPP's image-processing primitives allow for processing of such sub regions also referred to as regions-of-interest or ROIs.

All primitives supporting ROI processing are marked by a "R" in their name suffix. In most cases the ROI is passed as a single NppiSize struct, which provides the with and height of the ROI. This raises the question how the primitive knows where in the image this rectangle of (width, height) is located. The "start pixel" of the ROI is implicitly given by the image-data pointer. I.e. instead of explicitly passing a pixel coordinate for the upper-left corner (lowest memory address), the user simply offsets the image-data pointers to point to the first pixel of the ROI.

In practice this means that for an image (pSrc, nSrcStep) and the start-pixel of the ROI being at location (x, y), one would pass

pSrcOffset = pSrc + y ∗ nSrcStep + x ∗ PixelSize;

as the image-data source to the primitive. PixelSize is typically computed as

PixelSize = NumberOfColorChannels ∗ sizeof(PixelDataType).

E.g. for a pimitive like nppiSet_16s_C4R() we would have

- NumberOfColorChannels == 4;
- sizeof(Npp16s) == 2;
- and thus PixelSize = 4 ∗ 2 = 8;

4.3.1 ROI Related Error Codes

All NPPI primitives operating on ROIs of image data validate the ROI size and image's step size. Failed validation results in one of the following error codes being returned and the primitive not being executed:

- NPP_SIZE_ERROR is returned if either the ROI width or ROI height are negative.
- NPP_STEP_ERROR is returned if the ROI width exceeds the image's line step. In mathematical terms $(\text{widthROI} * \text{PixelSize}) > \text{nLinStep}$ indicates an error.

4.4 Masked Operation

Some primitive support masked operation. An "M" in the suffix of those variants indicates masked operation. Primitives supporting masked operation consume an additional input image provided via a Mask-Image Pointer and Mask-Image Line Step. The mask image is interpreted by these primitives as a boolean image. The values of type Npp8u are interpreted as boolean values where a values of 0 indicates false, any non-zero values true.

Unless otherwise indicated the operation is only performed on pixels where its spatially corresponding mask pixel is true (non-zero). E.g. a masked copy operation would only copy those pixels in the ROI that have corresponding non-zero mask pixels.

4.5 Channel-of-Interest API

Some primitives allow restricting operations to a single channel of interest within a multi-channel image. These primitives are suffixed with the letter "C" (after the channel information, e.g. nppiCopy_-8u_C3CR(...). The channel-of-interest is generally selected by offsetting the image-data pointer to point directly to the channel- of-interest rather than the base of the first pixel in the ROI. Some primitives also explicitly specify the selected channel number and pass it via an integer, e.g. nppiMean_StdDev_8u_-C3CR(...).

4.5.1 Select-Channel Source-Image Pointer

This is a pointer to the channel-of-interest within the first pixel of the source image. E.g. if pSrc is the pointer to the first pixel inside the ROI of a three channel image. Using the appropriate select-channel copy primitive one could copy the second channel of this source image into the first channel of a destination image given by pDst by offsetting the pointer by one:

```
nppiCopy_8u_C3CR(pSrc + 1, nSrcStep, pDst, nDstStep, oSizeROI);
```

4.5.2 Select-Channel Source-Image

Some primitives allow the user to select the channel-of-interest by specifying the channle number (nCOI). This approach is typically used in the image statistical functions. For example,

```
nppiMean_StdDev_8u_C3CR(pSrc, nSrcStep, oSizeROI, nCOI, pDeviceBuffer, pMean, pStdDev );
```

The channel-of-interest number can be either 1, 2, or 3.

4.5.3 Select-Channel Destination-Image Pointer

This is a pointer to the channel-of-interest within the first pixel of the destination image. E.g. if pDst is the pointer to the first pixel inside the ROI of a three channel image. Using the appropriate select-channel copy primitive one could copy data into the second channel of this destination image from the first channel of a source image given by pSrc by offseting the destination pointer by one:

```
nppiCopy_8u_C3CR(pSrc, nSrcStep, pDst + 1, nDstStep, oSizeROI);
```

4.6 Source-Image Sampling

A large number of NPP image-processing functions consume at least one source image and produce an output image (e.g. nppiAddC_8u_C1RSfs() or nppiFilterBox_8u_C1R()). All NPP functions falling into this category also operate on ROIs (see Region-of-Interest (ROI)) which for these functions should be considered to describe the destination ROI. In other words the ROI describes a rectangular region in the destination image and all pixels inside of this region are being written by the function in question.

In order to use such functions successfully it is important to understand how the user defined destination ROI affects which pixels in the input image(s) are being read by the algorithms. To simplify the discussion of ROI propagation (i.e. given a destination ROI, what are the ROIs in in the source(s)), it makes sense to distinguish two major cases:

1. Point-Wise Operations: These are primitives like nppiAddC_8u_C1RSfs(). Each output pixel requires exaclty one input pixel to be read.
2. Neighborhood Operations: These are primitives like nppiFilterBox_8u_C1R(), which require a group of pixels from the source image(s) to be read in order to produce a single output.

4.6.1 Point-Wise Operations

As mentioned above, point-wise operations consume a single pixel from the input image (or a single pixel from each input image, if the operation in question has more than one input image) in order to produce a single output pixel.

4.6.2 Neighborhood Operations

In the case of neightborhood operations a number of input pixels (a "neighborhood" of pixels) is read in the input image (or images) in order to compute a single output pixel. All of the functions for image_-filtering_functions and image_morphological_operations are neigborhood operations.

Most of these functions have parameters that affect the size and relative location of the neighborhood: a mask-size structure and an achor-point structure. Both parameters are described in more detail in the next subsections.

4.6.2.1 Mask-Size Parameter

Many NPP neighborhood operations allow the user to specify the size of the neightborhood via a parameter usually named oMaskSize of type NppiSize. In those cases the neighborhood of pixels read from the source(s) is exactly the size of the mask. Assuming the mask is anchored at location (0, 0) (see Anchor-Point Parameter below) and has a size of (w, h), i.e.

```
assert(oMaskSize.w == w);
assert(oMaskSize.h == h);
assert(oAnchor.x == 0);
assert(oAnchor.y == 0);
```

a neighborhood operation would read the following source pixels in order to compute destiation pixel $D_{i,j}$:

$$\begin{array}{llll} S_{i,j} & S_{i,j+1} & \ldots & S_{i,j+w-1} \\ S_{i+1,j} & S_{i+1,j+1} & \ldots & S_{i+1,j+w-1} \\ \vdots & \vdots & \ddots & \vdots \\ S_{i+h-1,j} & S_{i+h-1,j+1} & \ldots & S_{i+h-1,j+w-1} \end{array}$$

4.6.2.2 Anchor-Point Parameter

Many NPP primitives perforing neighborhood operations allow the user to specify the relative location of the neighborhood via a parameter usually named oAnchor of type NppiPoint. Using the anchor a developer can chose the position of the mask (see Mask-Size Parameter) relative to current pixel index.

Using the same example as in Mask-Size Parameter, but this time with an anchor position of (a, b):

```
assert(oMaskSize.w == w);
assert(oMaskSize.h == h);
assert(oAnchor.x == a);
assert(oAnchor.y == b);
```

the following pixels from the source image would be read:

$$\begin{array}{llll} S_{i-a,j-b} & S_{i-a,j-b+1} & \ldots & S_{i-a,j-b+w-1} \\ S_{i-a+1,j-b} & S_{i-a+1,j-b+1} & \ldots & S_{i-a+1,j-b+w-1} \\ \vdots & \vdots & \ddots & \vdots \\ S_{i-a+h-1,j-b} & S_{i-a+h-1,j-b+1} & \ldots & S_{i-a+h-1,j-b+w-1} \end{array}$$

4.6.2.3 Sampling Beyond Image Boundaries

NPP primitives in general and NPP neighborhood operations in particular require that all pixel locations read and written are valid and within the boundaries of the respective images. Sampling outside of the defined image data regions results in undefined behavior and may lead to system instabilty.

This poses a problem in practice: when processing full-size images one cannot choose the destination ROI to be the same size as the source image. Because neigborhood operations read pixels from an enlarged source ROI, the destination ROI must be shrunk so that the expanded source ROI does not exceed the source image's size.

For cases where this "shrinking" of the destination image size is unacceptable, NPP provides a set of border-expanding Copy primitives. E.g. nppiCopyConstBorder_8u_C1R(), nppiCopyReplicateBorder_-8u_C1R() and nppiCopyWrapBorder_8u_C1R(). The user can use these primitives to "expand" the source image's size using one of the three expansion modes. The expanded image can then be safely passed to a neighborhood operation producing a full-size result.

Chapter 5

Module Index

5.1 Modules

Here is a list of all modules:

Chapter 6

Data Structure Index

6.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 7

Module Documentation

7.1 NPP Core

Basic functions for library management, in particular library version and device property query functions.

Functions

- const NppLibraryVersion ∗ nppGetLibVersion (void)

  Get the NPP library version.

- NppGpuComputeCapability nppGetGpuComputeCapability (void)

  What CUDA compute model is supported by the active CUDA device?

- int nppGetGpuNumSMs (void)

  Get the number of Streaming Multiprocessors (SM) on the active CUDA device.

- int nppGetMaxThreadsPerBlock (void)

  Get the maximum number of threads per block on the active CUDA device.

- int nppGetMaxThreadsPerSM (void)

  Get the maximum number of threads per SM for the active GPU.

- int nppGetGpuDeviceProperties (int ∗pMaxThreadsPerSM, int ∗pMaxThreadsPerBlock, int ∗pNumberOfSMs)

  Get the maximum number of threads per SM, maximum threads per block, and number of SMs for the active GPU.

- const char ∗ nppGetGpuName (void)

  Get the name of the active CUDA device.

- cudaStream_t nppGetStream (void)

  Get the NPP CUDA stream.

- unsigned int nppGetStreamNumSMs (void)

  Get the number of SMs on the device associated with the current NPP CUDA stream.

- unsigned int nppGetStreamMaxThreadsPerSM (void)

  Get the maximum number of threads per SM on the device associated with the current NPP CUDA stream.

- void nppSetStream (cudaStream_t hStream)

  Set the NPP CUDA stream.

7.1.1 Detailed Description

Basic functions for library management, in particular library version and device property query functions.

7.1.2 Function Documentation

7.1.2.1 NppGpuComputeCapability nppGetGpuComputeCapability (void)

What CUDA compute model is supported by the active CUDA device?

Before trying to call any NPP functions, the user should make a call this function to ensure that the current machine has a CUDA capable device.

Returns:

An enum value representing if a CUDA capable device was found and what level of compute capabilities it supports.

7.1.2.2 int nppGetGpuDeviceProperties (int ∗ *pMaxThreadsPerSM*, int ∗ *pMaxThreadsPerBlock*, int ∗ *pNumberOfSMs*)

Get the maximum number of threads per SM, maximum threads per block, and number of SMs for the active GPU.

Returns:

cudaSuccess for success, -1 for failure

7.1.2.3 const char∗ nppGetGpuName (void)

Get the name of the active CUDA device.

Returns:

Name string of the active graphics-card/compute device in a system.

7.1.2.4 int nppGetGpuNumSMs (void)

Get the number of Streaming Multiprocessors (SM) on the active CUDA device.

Returns:

Number of SMs of the default CUDA device.

7.1.2.5 const NppLibraryVersion∗ nppGetLibVersion (void)

Get the NPP library version.

Returns:

A struct containing separate values for major and minor revision and build number.

7.1.2.6 int nppGetMaxThreadsPerBlock (void)

Get the maximum number of threads per block on the active CUDA device.

Returns:

Maximum number of threads per block on the active CUDA device.

7.1.2.7 int nppGetMaxThreadsPerSM (void)

Get the maximum number of threads per SM for the active GPU.

Returns:

Maximum number of threads per SM for the active GPU

7.1.2.8 cudaStream_t nppGetStream (void)

Get the NPP CUDA stream.

NPP enables concurrent device tasks via a global stream state varible. The NPP stream by default is set to stream 0, i.e. non-concurrent mode. A user can set the NPP stream to any valid CUDA stream. All CUDA commands issued by NPP (e.g. kernels launched by the NPP library) are then issed to that NPP stream.

7.1.2.9 unsigned int nppGetStreamMaxThreadsPerSM (void)

Get the maximum number of threads per SM on the device associated with the current NPP CUDA stream.

NPP enables concurrent device tasks via a global stream state varible. The NPP stream by default is set to stream 0, i.e. non-concurrent mode. A user can set the NPP stream to any valid CUDA stream. All CUDA commands issued by NPP (e.g. kernels launched by the NPP library) are then issed to that NPP stream. This call avoids a cudaGetDeviceProperties() call.

7.1.2.10 unsigned int nppGetStreamNumSMs (void)

Get the number of SMs on the device associated with the current NPP CUDA stream.

NPP enables concurrent device tasks via a global stream state varible. The NPP stream by default is set to stream 0, i.e. non-concurrent mode. A user can set the NPP stream to any valid CUDA stream. All CUDA commands issued by NPP (e.g. kernels launched by the NPP library) are then issed to that NPP stream. This call avoids a cudaGetDeviceProperties() call.

7.1.2.11 void nppSetStream (cudaStream_t *hStream*)

Set the NPP CUDA stream.

See also:

nppGetStream()

7.2 NPP Type Definitions and Constants

Data Structures

- struct NppLibraryVersion
- struct NppiPoint

  2D Point

- struct NppPointPolar

  2D Polar Point

- struct NppiSize

  2D Size This struct typically represents the size of a a rectangular region in two space.

- struct NppiRect

  2D Rectangle This struct contains position and size information of a rectangle in two space.

- struct NppiHOGConfig

  The NppiHOGConfig structure defines the configuration parameters for the HOG descriptor:.

- struct NppiHaarClassifier_32f
- struct NppiHaarBuffer

Modules

- Basic NPP Data Types

Defines

- #define NPP_MIN_8U (0)

  Minimum 8-bit unsigned integer.

- #define NPP_MAX_8U (255)

  Maximum 8-bit unsigned integer.

- #define NPP_MIN_16U (0)

  Minimum 16-bit unsigned integer.

- #define NPP_MAX_16U (65535)

  Maximum 16-bit unsigned integer.

- #define NPP_MIN_32U (0)

  Minimum 32-bit unsigned integer.

- #define NPP_MAX_32U (4294967295U)

  Maximum 32-bit unsigned integer.

- #define NPP_MIN_64U (0)

  Minimum 64-bit unsigned integer.

- #define NPP_MAX_64U (18446744073709551615ULL)

  Maximum 64-bit unsigned integer.

- #define NPP_MIN_8S (-127 - 1)

  Minimum 8-bit signed integer.

- #define NPP_MAX_8S (127)

  Maximum 8-bit signed integer.

- #define NPP_MIN_16S (-32767 - 1)

  Minimum 16-bit signed integer.

- #define NPP_MAX_16S (32767)

  Maximum 16-bit signed integer.

- #define NPP_MIN_32S (-2147483647 - 1)

  Minimum 32-bit signed integer.

- #define NPP_MAX_32S (2147483647)

  Maximum 32-bit signed integer.

- #define NPP_MAX_64S (9223372036854775807LL)

  Maximum 64-bit signed integer.

- #define NPP_MIN_64S (-9223372036854775807LL - 1)

  Minimum 64-bit signed integer.

- #define NPP_MINABS_32F (1.175494351e-38f)

  Smallest positive 32-bit floating point value.

- #define NPP_MAXABS_32F (3.402823466e+38f)

  Largest positive 32-bit floating point value.

- #define NPP_MINABS_64F (2.2250738585072014e-308)

  Smallest positive 64-bit floating point value.

- #define NPP_MAXABS_64F (1.7976931348623158e+308)

  Largest positive 64-bit floating point value.

- #define NPP_HOG_MAX_CELL_SIZE (16)

  max horizontal/vertical pixel size of cell.

- #define NPP_HOG_MAX_BLOCK_SIZE (64)

  max horizontal/vertical pixel size of block.

- #define NPP_HOG_MAX_BINS_PER_CELL (16)

  max number of histogram bins.

- #define NPP_HOG_MAX_CELLS_PER_DESCRIPTOR (256) *max number of cells in a descriptor window.*

- #define NPP_HOG_MAX_OVERLAPPING_BLOCKS_PER_DESCRIPTOR (256)

  max number of overlapping blocks in a descriptor window.

- #define NPP_HOG_MAX_DESCRIPTOR_LOCATIONS_PER_CALL (128)

  max number of descriptor window locations per function call.

Enumerations

- enum NppiInterpolationMode {
  NPPI_INTER_UNDEFINED = 0,
  NPPI_INTER_NN = 1,
  NPPI_INTER_LINEAR = 2,
  NPPI_INTER_CUBIC = 4,
  NPPI_INTER_CUBIC2P_BSPLINE,
  NPPI_INTER_CUBIC2P_CATMULLROM,
  NPPI_INTER_CUBIC2P_B05C03,
  NPPI_INTER_SUPER = 8,
  NPPI_INTER_LANCZOS = 16,
  NPPI_INTER_LANCZOS3_ADVANCED = 17,
  NPPI_SMOOTH_EDGE = (1 << 31) }

  Filtering methods.

- enum NppiBayerGridPosition {
  NPPI_BAYER_BGGR = 0,
  NPPI_BAYER_RGGB = 1,
  NPPI_BAYER_GBRG = 2,
  NPPI_BAYER_GRBG = 3 }

  Bayer Grid Position Registration.

- enum NppiMaskSize {
  NPP_MASK_SIZE_1_X_3,
  NPP_MASK_SIZE_1_X_5,
  NPP_MASK_SIZE_3_X_1 = 100,
  NPP_MASK_SIZE_5_X_1,
  NPP_MASK_SIZE_3_X_3 = 200,
  NPP_MASK_SIZE_5_X_5,
  NPP_MASK_SIZE_7_X_7 = 400,
  NPP_MASK_SIZE_9_X_9 = 500,
  NPP_MASK_SIZE_11_X_11 = 600,
  NPP_MASK_SIZE_13_X_13 = 700,
  NPP_MASK_SIZE_15_X_15 = 800 }

Fixed filter-kernel sizes.

- enum NppiDifferentialKernel {
  NPP_FILTER_SOBEL,
  NPP_FILTER_SCHARR }

  Differential Filter types.

- enum NppStatus {
  NPP_NOT_SUPPORTED_MODE_ERROR = -9999,
  NPP_INVALID_HOST_POINTER_ERROR = -1032,
  NPP_INVALID_DEVICE_POINTER_ERROR = -1031,
  NPP_LUT_PALETTE_BITSIZE_ERROR = -1030,
  NPP_ZC_MODE_NOT_SUPPORTED_ERROR = -1028,
  NPP_NOT_SUFFICIENT_COMPUTE_CAPABILITY = -1027,
  NPP_TEXTURE_BIND_ERROR = -1024,
  NPP_WRONG_INTERSECTION_ROI_ERROR = -1020,
  NPP_HAAR_CLASSIFIER_PIXEL_MATCH_ERROR = -1006,
  NPP_MEMFREE_ERROR = -1005,
  NPP_MEMSET_ERROR = -1004,
  NPP_MEMCPY_ERROR = -1003,
  NPP_ALIGNMENT_ERROR = -1002,
  NPP_CUDA_KERNEL_EXECUTION_ERROR = -1000,
  NPP_ROUND_MODE_NOT_SUPPORTED_ERROR = -213,
  NPP_QUALITY_INDEX_ERROR = -210,
  NPP_RESIZE_NO_OPERATION_ERROR = -201,
  NPP_OVERFLOW_ERROR = -109,
  NPP_NOT_EVEN_STEP_ERROR = -108,
  NPP_HISTOGRAM_NUMBER_OF_LEVELS_ERROR = -107,
  NPP_LUT_NUMBER_OF_LEVELS_ERROR = -106,
  NPP_CORRUPTED_DATA_ERROR = -61,
  NPP_CHANNEL_ORDER_ERROR = -60,
  NPP_ZERO_MASK_VALUE_ERROR = -59,
  NPP_QUADRANGLE_ERROR = -58,
  NPP_RECTANGLE_ERROR = -57,
  NPP_COEFFICIENT_ERROR = -56,
  NPP_NUMBER_OF_CHANNELS_ERROR = -53,
  NPP_COI_ERROR = -52,
  NPP_DIVISOR_ERROR = -51,
  NPP_CHANNEL_ERROR = -47,
  NPP_STRIDE_ERROR = -37,
  NPP_ANCHOR_ERROR = -34,
  NPP_MASK_SIZE_ERROR = -33,

NPP_RESIZE_FACTOR_ERROR = -23,
NPP_INTERPOLATION_ERROR = -22,
NPP_MIRROR_FLIP_ERROR = -21,
NPP_MOMENT_00_ZERO_ERROR = -20,
NPP_THRESHOLD_NEGATIVE_LEVEL_ERROR = -19,
NPP_THRESHOLD_ERROR = -18,
NPP_CONTEXT_MATCH_ERROR = -17,
NPP_FFT_FLAG_ERROR = -16,
NPP_FFT_ORDER_ERROR = -15,
NPP_STEP_ERROR = -14,
NPP_SCALE_RANGE_ERROR = -13,
NPP_DATA_TYPE_ERROR = -12,
NPP_OUT_OFF_RANGE_ERROR = -11,
NPP_DIVIDE_BY_ZERO_ERROR = -10,
NPP_MEMORY_ALLOCATION_ERR = -9,
NPP_NULL_POINTER_ERROR = -8,
NPP_RANGE_ERROR = -7,
NPP_SIZE_ERROR = -6,
NPP_BAD_ARGUMENT_ERROR = -5,
NPP_NO_MEMORY_ERROR = -4,
NPP_NOT_IMPLEMENTED_ERROR = -3,
NPP_ERROR = -2,
NPP_ERROR_RESERVED = -1,
NPP_NO_ERROR = 0,
NPP_SUCCESS = NPP_NO_ERROR,
NPP_NO_OPERATION_WARNING = 1,
NPP_DIVIDE_BY_ZERO_WARNING = 6,
NPP_AFFINE_QUAD_INCORRECT_WARNING = 28,
NPP_WRONG_INTERSECTION_ROI_WARNING = 29,
NPP_WRONG_INTERSECTION_QUAD_WARNING = 30,
NPP_DOUBLE_SIZE_WARNING = 35,
NPP_MISALIGNED_DST_ROI_WARNING = 10000 }

Error Status Codes.

- enum NppGpuComputeCapability {
  NPP_CUDA_UNKNOWN_VERSION = -1,
  NPP_CUDA_NOT_CAPABLE = 0,
  NPP_CUDA_1_0 = 100,
  NPP_CUDA_1_1 = 110,
  NPP_CUDA_1_2 = 120,
  NPP_CUDA_1_3 = 130,

NPP_CUDA_2_0 = 200,
NPP_CUDA_2_1 = 210,
NPP_CUDA_3_0 = 300,
NPP_CUDA_3_2 = 320,
NPP_CUDA_3_5 = 350,
NPP_CUDA_3_7 = 370,
NPP_CUDA_5_0 = 500,
NPP_CUDA_5_2 = 520,
NPP_CUDA_5_3 = 530,
NPP_CUDA_6_0 = 600,
NPP_CUDA_6_1 = 610,
NPP_CUDA_6_2 = 620,
NPP_CUDA_6_3 = 630,
NPP_CUDA_7_0 = 700 }

- enum NppiAxis {
  NPP_HORIZONTAL_AXIS,
  NPP_VERTICAL_AXIS,
  NPP_BOTH_AXIS }
- enum NppCmpOp {
  NPP_CMP_LESS,
  NPP_CMP_LESS_EQ,
  NPP_CMP_EQ,
  NPP_CMP_GREATER_EQ,
  NPP_CMP_GREATER }
- enum NppRoundMode {
  NPP_RND_NEAR,
  NPP_ROUND_NEAREST_TIES_TO_EVEN = NPP_RND_NEAR,
  NPP_RND_FINANCIAL,
  NPP_ROUND_NEAREST_TIES_AWAY_FROM_ZERO = NPP_RND_FINANCIAL,
  NPP_RND_ZERO,
  NPP_ROUND_TOWARD_ZERO = NPP_RND_ZERO }

  Rounding Modes.
- enum NppiBorderType {
  NPP_BORDER_UNDEFINED = 0,
  NPP_BORDER_NONE = NPP_BORDER_UNDEFINED,
  NPP_BORDER_CONSTANT = 1,
  NPP_BORDER_REPLICATE = 2,
  NPP_BORDER_WRAP = 3,
  NPP_BORDER_MIRROR = 4 }

- enum NppHintAlgorithm {
  NPP_ALG_HINT_NONE,
  NPP_ALG_HINT_FAST,
  NPP_ALG_HINT_ACCURATE }
- enum NppiAlphaOp {
  NPPI_OP_ALPHA_OVER,
  NPPI_OP_ALPHA_IN,
  NPPI_OP_ALPHA_OUT,
  NPPI_OP_ALPHA_ATOP,
  NPPI_OP_ALPHA_XOR,
  NPPI_OP_ALPHA_PLUS,
  NPPI_OP_ALPHA_OVER_PREMUL,
  NPPI_OP_ALPHA_IN_PREMUL,
  NPPI_OP_ALPHA_OUT_PREMUL,
  NPPI_OP_ALPHA_ATOP_PREMUL,
  NPPI_OP_ALPHA_XOR_PREMUL,
  NPPI_OP_ALPHA_PLUS_PREMUL,
  NPPI_OP_ALPHA_PREMUL }
- enum NppsZCType {
  nppZCR,
  nppZCXor,
  nppZCC }
- enum NppiHuffmanTableType {
  nppiDCTable,
  nppiACTable }
- enum NppiNorm {
  nppiNormInf = 0,
  nppiNormL1 = 1,
  nppiNormL2 = 2 }

7.2.1 Define Documentation

7.2.1.1 #define NPP_HOG_MAX_BINS_PER_CELL (16)

max number of histogram bins.

7.2.1.2 #define NPP_HOG_MAX_BLOCK_SIZE (64)

max horizontal/vertical pixel size of block.

7.2.1.3 #define NPP_HOG_MAX_CELL_SIZE (16)

max horizontal/vertical pixel size of cell.

7.2.1.4 #define NPP_HOG_MAX_CELLS_PER_DESCRIPTOR (256)

max number of cells in a descriptor window.

7.2.1.5 #define NPP_HOG_MAX_DESCRIPTOR_LOCATIONS_PER_CALL (128)

max number of descriptor window locations per function call.

7.2.1.6 #define NPP_HOG_MAX_OVERLAPPING_BLOCKS_PER_DESCRIPTOR (256)

max number of overlapping blocks in a descriptor window.

7.2.1.7 #define NPP_MAX_16S (32767)

Maximum 16-bit signed integer.

7.2.1.8 #define NPP_MAX_16U (65535)

Maximum 16-bit unsigned integer.

7.2.1.9 #define NPP_MAX_32S (2147483647)

Maximum 32-bit signed integer.

7.2.1.10 #define NPP_MAX_32U (4294967295U)

Maximum 32-bit unsigned integer.

7.2.1.11 #define NPP_MAX_64S (9223372036854775807LL)

Maximum 64-bit signed integer.

7.2.1.12 #define NPP_MAX_64U (18446744073709551615ULL)

Maximum 64-bit unsigned integer.

7.2.1.13 #define NPP_MAX_8S (127)

Maximum 8-bit signed integer.

7.2.1.14 #define NPP_MAX_8U (255)

Maximum 8-bit unsigned integer.

7.2.1.15 #define NPP_MAXABS_32F (3.402823466e+38f)

Largest positive 32-bit floating point value.

7.2.1.16 #define NPP_MAXABS_64F (1.7976931348623158e+308)

Largest positive 64-bit floating point value.

7.2.1.17 #define NPP_MIN_16S (-32767 - 1)

Minimum 16-bit signed integer.

7.2.1.18 #define NPP_MIN_16U (0)

Minimum 16-bit unsigned integer.

7.2.1.19 #define NPP_MIN_32S (-2147483647 - 1)

Minimum 32-bit signed integer.

7.2.1.20 #define NPP_MIN_32U (0)

Minimum 32-bit unsigned integer.

7.2.1.21 #define NPP_MIN_64S (-9223372036854775807LL - 1)

Minimum 64-bit signed integer.

7.2.1.22 #define NPP_MIN_64U (0)

Minimum 64-bit unsigned integer.

7.2.1.23 #define NPP_MIN_8S (-127 - 1)

Minimum 8-bit signed integer.

7.2.1.24 #define NPP_MIN_8U (0)

Minimum 8-bit unsigned integer.

7.2.1.25 #define NPP_MINABS_32F (1.175494351e-38f)

Smallest positive 32-bit floating point value.

7.2.1.26 #define NPP_MINABS_64F (2.2250738585072014e-308)

Smallest positive 64-bit floating point value.

7.2.2 Enumeration Type Documentation

7.2.2.1 enum NppCmpOp

Enumerator:

NPP_CMP_LESS

NPP_CMP_LESS_EQ

NPP_CMP_EQ

NPP_CMP_GREATER_EQ

NPP_CMP_GREATER

7.2.2.2 enum NppGpuComputeCapability

Enumerator:

NPP_CUDA_UNKNOWN_VERSION	Indicates that the compute-capability query failed.
NPP_CUDA_NOT_CAPABLE	Indicates that no CUDA capable device was found.
NPP_CUDA_1_0	Indicates that CUDA 1.0 capable device is machine's default device.
NPP_CUDA_1_1	Indicates that CUDA 1.1 capable device is machine's default device.
NPP_CUDA_1_2	Indicates that CUDA 1.2 capable device is machine's default device.
NPP_CUDA_1_3	Indicates that CUDA 1.3 capable device is machine's default device.
NPP_CUDA_2_0	Indicates that CUDA 2.0 capable device is machine's default device.
NPP_CUDA_2_1	Indicates that CUDA 2.1 capable device is machine's default device.
NPP_CUDA_3_0	Indicates that CUDA 3.0 capable device is machine's default device.
NPP_CUDA_3_2	Indicates that CUDA 3.2 capable device is machine's default device.
NPP_CUDA_3_5	Indicates that CUDA 3.5 capable device is machine's default device.
NPP_CUDA_3_7	Indicates that CUDA 3.7 capable device is machine's default device.
NPP_CUDA_5_0	Indicates that CUDA 5.0 capable device is machine's default device.
NPP_CUDA_5_2	Indicates that CUDA 5.2 capable device is machine's default device.
NPP_CUDA_5_3	Indicates that CUDA 5.3 capable device is machine's default device.
NPP_CUDA_6_0	Indicates that CUDA 6.0 capable device is machine's default device.
NPP_CUDA_6_1	Indicates that CUDA 6.1 capable device is machine's default device.
NPP_CUDA_6_2	Indicates that CUDA 6.2 capable device is machine's default device.
NPP_CUDA_6_3	Indicates that CUDA 6.3 capable device is machine's default device.
NPP_CUDA_7_0	Indicates that CUDA 7.0 or better is machine's default device.

7.2.2.3 enum NppHintAlgorithm

Enumerator:

NPP_ALG_HINT_NONE
NPP_ALG_HINT_FAST
NPP_ALG_HINT_ACCURATE

7.2.2.4 enum NppiAlphaOp

Enumerator:

NPPI_OP_ALPHA_OVER
NPPI_OP_ALPHA_IN
NPPI_OP_ALPHA_OUT
NPPI_OP_ALPHA_ATOP
NPPI_OP_ALPHA_XOR
NPPI_OP_ALPHA_PLUS
NPPI_OP_ALPHA_OVER_PREMUL
NPPI_OP_ALPHA_IN_PREMUL
NPPI_OP_ALPHA_OUT_PREMUL
NPPI_OP_ALPHA_ATOP_PREMUL
NPPI_OP_ALPHA_XOR_PREMUL
NPPI_OP_ALPHA_PLUS_PREMUL
NPPI_OP_ALPHA_PREMUL

7.2.2.5 enum NppiAxis

Enumerator:

NPP_HORIZONTAL_AXIS
NPP_VERTICAL_AXIS
NPP_BOTH_AXIS

7.2.2.6 enum NppiBayerGridPosition

Bayer Grid Position Registration.

Enumerator:

NPPI_BAYER_BGGR	Default registration position.
NPPI_BAYER_RGGB	
NPPI_BAYER_GBRG	
NPPI_BAYER_GRBG	

7.2.2.7 enum NppiBorderType

Enumerator:

NPP_BORDER_UNDEFINED
NPP_BORDER_NONE
NPP_BORDER_CONSTANT
NPP_BORDER_REPLICATE
NPP_BORDER_WRAP
NPP_BORDER_MIRROR

7.2.2.8 enum NppiDifferentialKernel

Differential Filter types.

Enumerator:

NPP_FILTER_SOBEL
NPP_FILTER_SCHARR

7.2.2.9 enum NppiHuffmanTableType

Enumerator:

nppiDCTable DC Table.
nppiACTable AC Table.

7.2.2.10 enum NppiInterpolationMode

Filtering methods.

Enumerator:

NPPI_INTER_UNDEFINED
NPPI_INTER_NN Nearest neighbor filtering.
NPPI_INTER_LINEAR Linear interpolation.
NPPI_INTER_CUBIC Cubic interpolation.
NPPI_INTER_CUBIC2P_BSPLINE Two-parameter cubic filter (B=1, C=0).
NPPI_INTER_CUBIC2P_CATMULLROM Two-parameter cubic filter (B=0, C=1/2).
NPPI_INTER_CUBIC2P_B05C03 Two-parameter cubic filter (B=1/2, C=3/10).
NPPI_INTER_SUPER Super sampling.
NPPI_INTER_LANCZOS Lanczos filtering.
NPPI_INTER_LANCZOS3_ADVANCED Generic Lanczos filtering with order 3.
NPPI_SMOOTH_EDGE Smooth edge filtering.

7.2.2.11 enum NppiMaskSize

Fixed filter-kernel sizes.

Enumerator:

NPP_MASK_SIZE_1_X_3

NPP_MASK_SIZE_1_X_5

NPP_MASK_SIZE_3_X_1

NPP_MASK_SIZE_5_X_1

NPP_MASK_SIZE_3_X_3

NPP_MASK_SIZE_5_X_5

NPP_MASK_SIZE_7_X_7

NPP_MASK_SIZE_9_X_9

NPP_MASK_SIZE_11_X_11

NPP_MASK_SIZE_13_X_13

NPP_MASK_SIZE_15_X_15

7.2.2.12 enum NppiNorm

Enumerator:

nppiNormInf	maximum
nppiNormL1	sum
nppiNormL2	square root of sum of squares

7.2.2.13 enum NppRoundMode

Rounding Modes.

The enumerated rounding modes are used by a large number of NPP primitives to allow the user to specify the method by which fractional values are converted to integer values. Also see Rounding Modes.

For NPP release 5.5 new names for the three rounding modes are introduced that are based on the naming conventions for rounding modes set forth in the IEEE-754 floating-point standard. Developers are encouraged to use the new, longer names to be future proof as the legacy names will be deprecated in subsequent NPP releases.

Enumerator:

NPP_RND_NEAR Round to the nearest even integer.

All fractional numbers are rounded to their nearest integer. The ambiguous cases (i.e. <integer>.5) are rounded to the closest even integer. E.g.

- roundNear(0.5) = 0
- roundNear(0.6) = 1
- roundNear(1.5) = 2
- roundNear(-1.5) = -2

NPP_ROUND_NEAREST_TIES_TO_EVEN Alias name for NPP_RND_NEAR.

NPP_RND_FINANCIAL Round according to financial rule.

All fractional numbers are rounded to their nearest integer. The ambiguous cases (i.e. <integer>.5) are rounded away from zero. E.g.

- roundFinancial(0.4) = 0
- roundFinancial(0.5) = 1
- roundFinancial(-1.5) = -2

NPP_ROUND_NEAREST_TIES_AWAY_FROM_ZERO Alias name for NPP_RND_-FINANCIAL.

NPP_RND_ZERO Round towards zero (truncation).

All fractional numbers of the form <integer>.<decimals> are truncated to <integer>.

- roundZero(1.5) = 1
- roundZero(1.9) = 1
- roundZero(-2.5) = -2

NPP_ROUND_TOWARD_ZERO Alias name for NPP_RND_ZERO.

7.2.2.14 enum NppStatus

Error Status Codes.

Almost all NPP function return error-status information using these return codes. Negative return codes indicate errors, positive return codes indicate warnings, a return code of 0 indicates success.

Enumerator:

NPP_NOT_SUPPORTED_MODE_ERROR

NPP_INVALID_HOST_POINTER_ERROR

NPP_INVALID_DEVICE_POINTER_ERROR

NPP_LUT_PALETTE_BITSIZE_ERROR

NPP_ZC_MODE_NOT_SUPPORTED_ERROR ZeroCrossing mode not supported.

NPP_NOT_SUFFICIENT_COMPUTE_CAPABILITY

NPP_TEXTURE_BIND_ERROR

NPP_WRONG_INTERSECTION_ROI_ERROR

NPP_HAAR_CLASSIFIER_PIXEL_MATCH_ERROR

NPP_MEMFREE_ERROR

NPP_MEMSET_ERROR

NPP_MEMCPY_ERROR

NPP_ALIGNMENT_ERROR

NPP_CUDA_KERNEL_EXECUTION_ERROR

NPP_ROUND_MODE_NOT_SUPPORTED_ERROR Unsupported round mode.

NPP_QUALITY_INDEX_ERROR Image pixels are constant for quality index.

NPP_RESIZE_NO_OPERATION_ERROR One of the output image dimensions is less than 1 pixel.

NPP_OVERFLOW_ERROR Number overflows the upper or lower limit of the data type.

NPP_NOT_EVEN_STEP_ERROR Step value is not pixel multiple.

NPP_HISTOGRAM_NUMBER_OF_LEVELS_ERROR Number of levels for histogram is less than 2.

NPP_LUT_NUMBER_OF_LEVELS_ERROR Number of levels for LUT is less than 2.

NPP_CORRUPTED_DATA_ERROR Processed data is corrupted.

NPP_CHANNEL_ORDER_ERROR Wrong order of the destination channels.

NPP_ZERO_MASK_VALUE_ERROR All values of the mask are zero.

NPP_QUADRANGLE_ERROR The quadrangle is nonconvex or degenerates into triangle, line or point.

NPP_RECTANGLE_ERROR Size of the rectangle region is less than or equal to 1.

NPP_COEFFICIENT_ERROR Unallowable values of the transformation coefficients.

NPP_NUMBER_OF_CHANNELS_ERROR Bad or unsupported number of channels.

NPP_COI_ERROR Channel of interest is not 1, 2, or 3.

NPP_DIVISOR_ERROR Divisor is equal to zero.

NPP_CHANNEL_ERROR Illegal channel index.

NPP_STRIDE_ERROR Stride is less than the row length.

NPP_ANCHOR_ERROR Anchor point is outside mask.

NPP_MASK_SIZE_ERROR Lower bound is larger than upper bound.

NPP_RESIZE_FACTOR_ERROR

NPP_INTERPOLATION_ERROR

NPP_MIRROR_FLIP_ERROR

NPP_MOMENT_00_ZERO_ERROR

NPP_THRESHOLD_NEGATIVE_LEVEL_ERROR

NPP_THRESHOLD_ERROR

NPP_CONTEXT_MATCH_ERROR

NPP_FFT_FLAG_ERROR

NPP_FFT_ORDER_ERROR

NPP_STEP_ERROR Step is less or equal zero.

NPP_SCALE_RANGE_ERROR

NPP_DATA_TYPE_ERROR

NPP_OUT_OFF_RANGE_ERROR

NPP_DIVIDE_BY_ZERO_ERROR

NPP_MEMORY_ALLOCATION_ERR

NPP_NULL_POINTER_ERROR

NPP_RANGE_ERROR

NPP_SIZE_ERROR

NPP_BAD_ARGUMENT_ERROR

NPP_NO_MEMORY_ERROR

NPP_NOT_IMPLEMENTED_ERROR

NPP_ERROR

NPP_ERROR_RESERVED

NPP_NO_ERROR Error free operation.

NPP_SUCCESS Successful operation (same as NPP_NO_ERROR).

NPP_NO_OPERATION_WARNING Indicates that no operation was performed.

NPP_DIVIDE_BY_ZERO_WARNING Divisor is zero however does not terminate the execution.

NPP_AFFINE_QUAD_INCORRECT_WARNING Indicates that the quadrangle passed to one of affine warping functions doesn't have necessary properties.

First 3 vertices are used, the fourth vertex discarded.

NPP_WRONG_INTERSECTION_ROI_WARNING The given ROI has no interestion with either the source or destination ROI.

Thus no operation was performed.

NPP_WRONG_INTERSECTION_QUAD_WARNING The given quadrangle has no intersection with either the source or destination ROI.

Thus no operation was performed.

NPP_DOUBLE_SIZE_WARNING Image size isn't multiple of two.

Indicates that in case of 422/411/420 sampling the ROI width/height was modified for proper processing.

NPP_MISALIGNED_DST_ROI_WARNING Speed reduction due to uncoalesced memory accesses warning.

7.2.2.15 enum NppsZCType

Enumerator:

nppZCR sign change

nppZCXor sign change XOR

nppZCC sign change count_0

7.3 Basic NPP Data Types

Data Structures

- struct NPP_ALIGN_8

  Complex Number This struct represents an unsigned int complex number.

- struct NPP_ALIGN_16

  Complex Number This struct represents a long long complex number.

Typedefs

- typedef unsigned char Npp8u

  8-bit unsigned chars

- typedef signed char Npp8s

  8-bit signed chars

- typedef unsigned short Npp16u

  16-bit unsigned integers

- typedef short Npp16s

  16-bit signed integers

- typedef unsigned int Npp32u

  32-bit unsigned integers

- typedef int Npp32s

  32-bit signed integers

- typedef unsigned long long Npp64u

  64-bit unsigned integers

- typedef long long Npp64s

  64-bit signed integers

- typedef float Npp32f

  32-bit (IEEE) floating-point numbers

- typedef double Npp64f

  64-bit floating-point numbers

- typedef struct NPP_ALIGN_8 Npp32uc

  Complex Number This struct represents an unsigned int complex number.

- typedef struct NPP_ALIGN_8 Npp32sc

  Complex Number This struct represents a signed int complex number.

- typedef struct NPP_ALIGN_8 Npp32fc

  Complex Number This struct represents a single floating-point complex number.

- typedef struct NPP_ALIGN_16 Npp64sc

  Complex Number This struct represents a long long complex number.

- typedef struct NPP_ALIGN_16 Npp64fc

  Complex Number This struct represents a double floating-point complex number.

Functions

- struct __align__ (2)

  Complex Number This struct represents an unsigned char complex number.

- struct __align__ (4)

  Complex Number This struct represents an unsigned short complex number.

Variables

- Npp8uc
- Npp16uc
- Npp16sc

7.3.1 Typedef Documentation

7.3.1.1 typedef short Npp16s

16-bit signed integers

7.3.1.2 typedef unsigned short Npp16u

16-bit unsigned integers

7.3.1.3 typedef float Npp32f

32-bit (IEEE) floating-point numbers

7.3.1.4 typedef struct NPP_ALIGN_8 Npp32fc

Complex Number This struct represents a single floating-point complex number.

7.3.1.5 typedef int Npp32s

32-bit signed integers

7.3.1.6 typedef struct NPP_ALIGN_8 Npp32sc

Complex Number This struct represents a signed int complex number.

7.3.1.7 typedef unsigned int Npp32u

32-bit unsigned integers

7.3.1.8 typedef struct NPP_ALIGN_8 Npp32uc

Complex Number This struct represents an unsigned int complex number.

7.3.1.9 typedef double Npp64f

64-bit floating-point numbers

7.3.1.10 typedef struct NPP_ALIGN_16 Npp64fc

Complex Number This struct represents a double floating-point complex number.

7.3.1.11 typedef long long Npp64s

64-bit signed integers

7.3.1.12 typedef struct NPP_ALIGN_16 Npp64sc

Complex Number This struct represents a long long complex number.

7.3.1.13 typedef unsigned long long Npp64u

64-bit unsigned integers

7.3.1.14 typedef signed char Npp8s

8-bit signed chars

7.3.1.15 typedef unsigned char Npp8u

8-bit unsigned chars

7.3.2 Function Documentation

7.3.2.1 struct __align__ (4) `[read]`

Complex Number This struct represents an unsigned short complex number.

Complex Number This struct represents a short complex number.

< Real part

< Imaginary part

< Real part

< Imaginary part

7.3.2.2 struct __align__ (2) `[read]`

Complex Number This struct represents an unsigned char complex number.

< Real part

< Imaginary part

7.3.3 Variable Documentation

7.3.3.1 Npp16sc

7.3.3.2 Npp16uc

7.3.3.3 Npp8uc

7.4 Memory Management

Modules

- Malloc

  Signal-allocator methods for allocating 1D arrays of data in device memory.

- Free

  Free signal memory.

7.5 Malloc

Signal-allocator methods for allocating 1D arrays of data in device memory.

Functions

- Npp8u ∗ nppsMalloc_8u (int nSize)

  8-bit unsigned signal allocator.

- Npp8s ∗ nppsMalloc_8s (int nSize)

  8-bit signed signal allocator.

- Npp16u ∗ nppsMalloc_16u (int nSize)

  16-bit unsigned signal allocator.

- Npp16s ∗ nppsMalloc_16s (int nSize)

  16-bit signal allocator.

- Npp16sc ∗ nppsMalloc_16sc (int nSize)

  16-bit complex-value signal allocator.

- Npp32u ∗ nppsMalloc_32u (int nSize)

  32-bit unsigned signal allocator.

- Npp32s ∗ nppsMalloc_32s (int nSize)

  32-bit integer signal allocator.

- Npp32sc ∗ nppsMalloc_32sc (int nSize)

  32-bit complex integer signal allocator.

- Npp32f ∗ nppsMalloc_32f (int nSize)

  32-bit float signal allocator.

- Npp32fc ∗ nppsMalloc_32fc (int nSize)

  32-bit complex float signal allocator.

- Npp64s ∗ nppsMalloc_64s (int nSize)

  64-bit long integer signal allocator.

- Npp64sc ∗ nppsMalloc_64sc (int nSize)

  64-bit complex long integer signal allocator.

- Npp64f ∗ nppsMalloc_64f (int nSize)

  64-bit float (double) signal allocator.

- Npp64fc ∗ nppsMalloc_64fc (int nSize)

  64-bit complex complex signal allocator.

7.5.1 Detailed Description

Signal-allocator methods for allocating 1D arrays of data in device memory.

All allocators have size parameters to specify the size of the signal (1D array) being allocated.

The allocator methods return a pointer to the newly allocated memory of appropriate type. If device-memory allocation is not possible due to resource constaints the allocators return 0 (i.e. NULL pointer).

All signal allocators allocate memory aligned such that it is beneficial to the performance of the majority of the signal-processing primitives. It is no mandatory however to use these allocators. Any valid CUDA device-memory pointers can be passed to NPP primitives.

7.5.2 Function Documentation

7.5.2.1 Npp16s∗ nppsMalloc_16s (int *nSize*)

16-bit signal allocator.

Parameters:

nSize Number of shorts in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.2 Npp16sc∗ nppsMalloc_16sc (int *nSize*)

16-bit complex-value signal allocator.

Parameters:

nSize Number of 16-bit complex numbers in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.3 Npp16u∗ nppsMalloc_16u (int *nSize*)

16-bit unsigned signal allocator.

Parameters:

nSize Number of unsigned shorts in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.4 Npp32f∗ nppsMalloc_32f (int *nSize*)

32-bit float signal allocator.

Parameters:

nSize Number of floats in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.5 Npp32fc∗ nppsMalloc_32fc (int *nSize*)

32-bit complex float signal allocator.

Parameters:

nSize Number of complex float values in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.6 Npp32s∗ nppsMalloc_32s (int *nSize*)

32-bit integer signal allocator.

Parameters:

nSize Number of ints in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.7 Npp32sc∗ nppsMalloc_32sc (int *nSize*)

32-bit complex integer signal allocator.

Parameters:

nSize Number of complex integner values in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.8 Npp32u∗ nppsMalloc_32u (int *nSize*)

32-bit unsigned signal allocator.

Parameters:

nSize Number of unsigned ints in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.9 Npp64f∗ nppsMalloc_64f (int *nSize*)

64-bit float (double) signal allocator.

Parameters:

nSize Number of doubles in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.10 Npp64fc∗ nppsMalloc_64fc (int *nSize*)

64-bit complex complex signal allocator.

Parameters:

nSize Number of complex double valuess in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.11 Npp64s∗ nppsMalloc_64s (int *nSize*)

64-bit long integer signal allocator.

Parameters:

nSize Number of long ints in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.12 Npp64sc∗ nppsMalloc_64sc (int *nSize*)

64-bit complex long integer signal allocator.

Parameters:

nSize Number of complex long int values in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.13 Npp8s∗ nppsMalloc_8s (int *nSize*)

8-bit signed signal allocator.

Parameters:

nSize Number of (signed) chars in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.5.2.14 Npp8u∗ nppsMalloc_8u (int *nSize*)

8-bit unsigned signal allocator.

Parameters:

nSize Number of unsigned chars in the new signal.

Returns:

A pointer to the new signal. 0 (NULL-pointer) indicates that an error occurred during allocation.

7.6 Free

Free signal memory.

Functions

- void nppsFree (void ∗pValues)

    Free method for any signal memory.

7.6.1 Detailed Description

Free signal memory.

7.6.2 Function Documentation

7.6.2.1 void nppsFree (void ∗ *pValues*)

Free method for any signal memory.

Parameters:

pValues A pointer to memory allocated using nppiMalloc_<modifier>.

7.7 Initialization

Modules

- Set
- Zero
- Copy

7.8 Set

Set

Set methods for 1D vectors of various types.

The copy methods operate on vector data given as a pointer to the underlying data-type (e.g. 8-bit vectors would be passed as pointers to Npp8u type) and length of the vectors, i.e. the number of items.

- NppStatus nppsSet_8u (Npp8u nValue, Npp8u ∗pDst, int nLength)
  8-bit unsigned char, vector set method.
- NppStatus nppsSet_8s (Npp8s nValue, Npp8s ∗pDst, int nLength)
  8-bit signed char, vector set method.
- NppStatus nppsSet_16u (Npp16u nValue, Npp16u ∗pDst, int nLength)
  16-bit unsigned integer, vector set method.
- NppStatus nppsSet_16s (Npp16s nValue, Npp16s ∗pDst, int nLength)
  16-bit signed integer, vector set method.
- NppStatus nppsSet_16sc (Npp16sc nValue, Npp16sc ∗pDst, int nLength)
  16-bit integer complex, vector set method.
- NppStatus nppsSet_32u (Npp32u nValue, Npp32u ∗pDst, int nLength)
  32-bit unsigned integer, vector set method.
- NppStatus nppsSet_32s (Npp32s nValue, Npp32s ∗pDst, int nLength)
  32-bit signed integer, vector set method.
- NppStatus nppsSet_32sc (Npp32sc nValue, Npp32sc ∗pDst, int nLength)
  32-bit integer complex, vector set method.
- NppStatus nppsSet_32f (Npp32f nValue, Npp32f ∗pDst, int nLength)
  32-bit float, vector set method.
- NppStatus nppsSet_32fc (Npp32fc nValue, Npp32fc ∗pDst, int nLength)
  32-bit float complex, vector set method.
- NppStatus nppsSet_64s (Npp64s nValue, Npp64s ∗pDst, int nLength)
  64-bit long long integer, vector set method.
- NppStatus nppsSet_64sc (Npp64sc nValue, Npp64sc ∗pDst, int nLength)
  64-bit long long integer complex, vector set method.
- NppStatus nppsSet_64f (Npp64f nValue, Npp64f ∗pDst, int nLength)
  64-bit double, vector set method.
- NppStatus nppsSet_64fc (Npp64fc nValue, Npp64fc ∗pDst, int nLength)
  64-bit double complex, vector set method.

7.8.1 Function Documentation

7.8.1.1 NppStatus nppsSet_16s (Npp16s *nValue*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed integer, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.2 NppStatus nppsSet_16sc (Npp16sc *nValue*, Npp16sc ∗ *pDst*, int *nLength*)

16-bit integer complex, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.3 NppStatus nppsSet_16u (Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned integer, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.4 NppStatus nppsSet_32f (Npp32f *nValue*, Npp32f ∗ *pDst*, int *nLength*)

32-bit float, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.5 NppStatus nppsSet_32fc (Npp32fc *nValue*, Npp32fc $*$ *pDst*, int *nLength*)

32-bit float complex, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.6 NppStatus nppsSet_32s (Npp32s *nValue*, Npp32s $*$ *pDst*, int *nLength*)

32-bit signed integer, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.7 NppStatus nppsSet_32sc (Npp32sc *nValue*, Npp32sc $*$ *pDst*, int *nLength*)

32-bit integer complex, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.8 NppStatus nppsSet_32u (Npp32u *nValue*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.9 NppStatus nppsSet_64f (Npp64f *nValue*, Npp64f ∗ *pDst*, int *nLength*)

64-bit double, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.10 NppStatus nppsSet_64fc (Npp64fc *nValue*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit double complex, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.11 NppStatus nppsSet_64s (Npp64s *nValue*, Npp64s ∗ *pDst*, int *nLength*)

64-bit long long integer, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.
pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.12 NppStatus nppsSet_64sc (Npp64sc *nValue*, Npp64sc ∗ *pDst*, int *nLength*)

64-bit long long integer complex, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.13 NppStatus nppsSet_8s (Npp8s *nValue*, Npp8s ∗ *pDst*, int *nLength*)

8-bit signed char, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.8.1.14 NppStatus nppsSet_8u (Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char, vector set method.

Parameters:

nValue Value used to initialize the vector pDst.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.9 Zero

Zero

Set signals to zero.

- NppStatus nppsZero_8u (Npp8u ∗pDst, int nLength)
  8-bit unsigned char, vector zero method.
- NppStatus nppsZero_16s (Npp16s ∗pDst, int nLength)
  16-bit integer, vector zero method.
- NppStatus nppsZero_16sc (Npp16sc ∗pDst, int nLength)
  16-bit integer complex, vector zero method.
- NppStatus nppsZero_32s (Npp32s ∗pDst, int nLength)
  32-bit integer, vector zero method.
- NppStatus nppsZero_32sc (Npp32sc ∗pDst, int nLength)
  32-bit integer complex, vector zero method.
- NppStatus nppsZero_32f (Npp32f ∗pDst, int nLength)
  32-bit float, vector zero method.
- NppStatus nppsZero_32fc (Npp32fc ∗pDst, int nLength)
  32-bit float complex, vector zero method.
- NppStatus nppsZero_64s (Npp64s ∗pDst, int nLength)
  64-bit long long integer, vector zero method.
- NppStatus nppsZero_64sc (Npp64sc ∗pDst, int nLength)
  64-bit long long integer complex, vector zero method.
- NppStatus nppsZero_64f (Npp64f ∗pDst, int nLength)
  64-bit double, vector zero method.
- NppStatus nppsZero_64fc (Npp64fc ∗pDst, int nLength)
  64-bit double complex, vector zero method.

7.9.1 Function Documentation

7.9.1.1 NppStatus nppsZero_16s (Npp16s ∗ *pDst*, int *nLength*)

16-bit integer, vector zero method.

Parameters:

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.9.1.2 NppStatus nppsZero_16sc (Npp16sc * *pDst*, int *nLength*)

16-bit integer complex, vector zero method.

Parameters:

pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.9.1.3 NppStatus nppsZero_32f (Npp32f * *pDst*, int *nLength*)

32-bit float, vector zero method.

Parameters:

pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.9.1.4 NppStatus nppsZero_32fc (Npp32fc * *pDst*, int *nLength*)

32-bit float complex, vector zero method.

Parameters:

pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.9.1.5 NppStatus nppsZero_32s (Npp32s * *pDst*, int *nLength*)

32-bit integer, vector zero method.

Parameters:

pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.9.1.6 NppStatus nppsZero_32sc (Npp32sc ∗ *pDst*, int *nLength*)

32-bit integer complex, vector zero method.

Parameters:

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.9.1.7 NppStatus nppsZero_64f (Npp64f ∗ *pDst*, int *nLength*)

64-bit double, vector zero method.

Parameters:

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.9.1.8 NppStatus nppsZero_64fc (Npp64fc ∗ *pDst*, int *nLength*)

64-bit double complex, vector zero method.

Parameters:

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.9.1.9 NppStatus nppsZero_64s (Npp64s ∗ *pDst*, int *nLength*)

64-bit long long integer, vector zero method.

Parameters:

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.9.1.10 NppStatus nppsZero_64sc (Npp64sc ∗ *pDst*, int *nLength*)

64-bit long long integer complex, vector zero method.

Parameters:

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.9.1.11 NppStatus nppsZero_8u (Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char, vector zero method.

Parameters:

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.10 Copy

Copy

Copy methods for various type signals.

Copy methods operate on signal data given as a pointer to the underlying data-type (e.g. 8-bit vectors would be passed as pointers to Npp8u type) and length of the vectors, i.e. the number of items.

- NppStatus nppsCopy_8u (const Npp8u ∗pSrc, Npp8u ∗pDst, int nLength)
  8-bit unsigned char, vector copy method
- NppStatus nppsCopy_16s (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength)
  16-bit signed short, vector copy method.
- NppStatus nppsCopy_32s (const Npp32s ∗pSrc, Npp32s ∗pDst, int nLength)
  32-bit signed integer, vector copy method.
- NppStatus nppsCopy_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)
  32-bit float, vector copy method.
- NppStatus nppsCopy_64s (const Npp64s ∗pSrc, Npp64s ∗pDst, int nLength)
  64-bit signed integer, vector copy method.
- NppStatus nppsCopy_16sc (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength)
  16-bit complex short, vector copy method.
- NppStatus nppsCopy_32sc (const Npp32sc ∗pSrc, Npp32sc ∗pDst, int nLength)
  32-bit complex signed integer, vector copy method.
- NppStatus nppsCopy_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength)
  32-bit complex float, vector copy method.
- NppStatus nppsCopy_64sc (const Npp64sc ∗pSrc, Npp64sc ∗pDst, int nLength)
  64-bit complex signed integer, vector copy method.
- NppStatus nppsCopy_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength)
  64-bit complex double, vector copy method.

7.10.1 Function Documentation

7.10.1.1 NppStatus nppsCopy_16s (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.10.1.2 NppStatus nppsCopy_16sc (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pDst*, int *nLength*)

16-bit complex short, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.10.1.3 NppStatus nppsCopy_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit float, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.10.1.4 NppStatus nppsCopy_32fc (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit complex float, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.10.1.5 NppStatus nppsCopy_32s (const Npp32s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*)

32-bit signed integer, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.10.1.6 NppStatus nppsCopy_32sc (const Npp32sc ∗ *pSrc*, Npp32sc ∗ *pDst*, int *nLength*)

32-bit complex signed integer, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.10.1.7 NppStatus nppsCopy_64fc (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit complex double, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.10.1.8 NppStatus nppsCopy_64s (const Npp64s ∗ *pSrc*, Npp64s ∗ *pDst*, int *nLength*)

64-bit signed integer, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.10.1.9 NppStatus nppsCopy_64sc (const Npp64sc ∗ *pSrc*, Npp64sc ∗ *pDst*, int *nLength*)

64-bit complex signed integer, vector copy method.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.10.1.10 NppStatus nppsCopy_8u (const Npp8u ∗ *pSrc*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char, vector copy method

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.11 Conversion Functions

Modules

- Convert
- Threshold

7.12 Convert

Convert

Routines for converting the sample-data type of signals.

- NppStatus nppsConvert_8s16s (const Npp8s ∗pSrc, Npp16s ∗pDst, int nLength)
- NppStatus nppsConvert_8s32f (const Npp8s ∗pSrc, Npp32f ∗pDst, int nLength)
- NppStatus nppsConvert_8u32f (const Npp8u ∗pSrc, Npp32f ∗pDst, int nLength)
- NppStatus nppsConvert_16s8s_Sfs (const Npp16s ∗pSrc, Npp8s ∗pDst, Npp32u nLength, NppRoundMode eRoundMode, int nScaleFactor)
- NppStatus nppsConvert_16s32s (const Npp16s ∗pSrc, Npp32s ∗pDst, int nLength)
- NppStatus nppsConvert_16s32f (const Npp16s ∗pSrc, Npp32f ∗pDst, int nLength)
- NppStatus nppsConvert_16u32f (const Npp16u ∗pSrc, Npp32f ∗pDst, int nLength)
- NppStatus nppsConvert_32s16s (const Npp32s ∗pSrc, Npp16s ∗pDst, int nLength)
- NppStatus nppsConvert_32s32f (const Npp32s ∗pSrc, Npp32f ∗pDst, int nLength)
- NppStatus nppsConvert_32s64f (const Npp32s ∗pSrc, Npp64f ∗pDst, int nLength)
- NppStatus nppsConvert_32f64f (const Npp32f ∗pSrc, Npp64f ∗pDst, int nLength)
- NppStatus nppsConvert_64s64f (const Npp64s ∗pSrc, Npp64f ∗pDst, int nLength)
- NppStatus nppsConvert_64f32f (const Npp64f ∗pSrc, Npp32f ∗pDst, int nLength)
- NppStatus nppsConvert_16s32f_Sfs (const Npp16s ∗pSrc, Npp32f ∗pDst, int nLength, int nScaleFactor)
- NppStatus nppsConvert_16s64f_Sfs (const Npp16s ∗pSrc, Npp64f ∗pDst, int nLength, int nScaleFactor)
- NppStatus nppsConvert_32s16s_Sfs (const Npp32s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)
- NppStatus nppsConvert_32s32f_Sfs (const Npp32s ∗pSrc, Npp32f ∗pDst, int nLength, int nScaleFactor)
- NppStatus nppsConvert_32s64f_Sfs (const Npp32s ∗pSrc, Npp64f ∗pDst, int nLength, int nScaleFactor)
- NppStatus nppsConvert_32f8s_Sfs (const Npp32f ∗pSrc, Npp8s ∗pDst, int nLength, NppRoundMode eRoundMode, int nScaleFactor)
- NppStatus nppsConvert_32f8u_Sfs (const Npp32f ∗pSrc, Npp8u ∗pDst, int nLength, NppRoundMode eRoundMode, int nScaleFactor)
- NppStatus nppsConvert_32f16s_Sfs (const Npp32f ∗pSrc, Npp16s ∗pDst, int nLength, NppRoundMode eRoundMode, int nScaleFactor)
- NppStatus nppsConvert_32f16u_Sfs (const Npp32f ∗pSrc, Npp16u ∗pDst, int nLength, NppRoundMode eRoundMode, int nScaleFactor)
- NppStatus nppsConvert_32f32s_Sfs (const Npp32f ∗pSrc, Npp32s ∗pDst, int nLength, NppRoundMode eRoundMode, int nScaleFactor)
- NppStatus nppsConvert_64s32s_Sfs (const Npp64s ∗pSrc, Npp32s ∗pDst, int nLength, NppRoundMode eRoundMode, int nScaleFactor)
- NppStatus nppsConvert_64f16s_Sfs (const Npp64f ∗pSrc, Npp16s ∗pDst, int nLength, NppRoundMode eRoundMode, int nScaleFactor)
- NppStatus nppsConvert_64f32s_Sfs (const Npp64f ∗pSrc, Npp32s ∗pDst, int nLength, NppRoundMode eRoundMode, int nScaleFactor)
- NppStatus nppsConvert_64f64s_Sfs (const Npp64f ∗pSrc, Npp64s ∗pDst, int nLength, NppRoundMode eRoundMode, int nScaleFactor)

7.12.1 Function Documentation

7.12.1.1 NppStatus nppsConvert_16s32f (const Npp16s ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

7.12.1.2 NppStatus nppsConvert_16s32f_Sfs (const Npp16s ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*, int *nScaleFactor*)

7.12.1.3 NppStatus nppsConvert_16s32s (const Npp16s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*)

7.12.1.4 NppStatus nppsConvert_16s64f_Sfs (const Npp16s ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*, int *nScaleFactor*)

7.12.1.5 NppStatus nppsConvert_16s8s_Sfs (const Npp16s ∗ *pSrc*, Npp8s ∗ *pDst*, Npp32u *nLength*, NppRoundMode *eRoundMode*, int *nScaleFactor*)

7.12.1.6 NppStatus nppsConvert_16u32f (const Npp16u ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

7.12.1.7 NppStatus nppsConvert_32f16s_Sfs (const Npp32f ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, NppRoundMode *eRoundMode*, int *nScaleFactor*)

7.12.1.8 NppStatus nppsConvert_32f16u_Sfs (const Npp32f ∗ *pSrc*, Npp16u ∗ *pDst*, int *nLength*, NppRoundMode *eRoundMode*, int *nScaleFactor*)

7.12.1.9 NppStatus nppsConvert_32f32s_Sfs (const Npp32f ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*, NppRoundMode *eRoundMode*, int *nScaleFactor*)

7.12.1.10 NppStatus nppsConvert_32f64f (const Npp32f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*)

7.12.1.11 NppStatus nppsConvert_32f8s_Sfs (const Npp32f ∗ *pSrc*, Npp8s ∗ *pDst*, int *nLength*, NppRoundMode *eRoundMode*, int *nScaleFactor*)

7.12.1.12 NppStatus nppsConvert_32f8u_Sfs (const Npp32f ∗ *pSrc*, Npp8u ∗ *pDst*, int *nLength*, NppRoundMode *eRoundMode*, int *nScaleFactor*)

7.12.1.13 NppStatus nppsConvert_32s16s (const Npp32s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*)

7.12.1.14 NppStatus nppsConvert_32s16s_Sfs (const Npp32s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

7.12.1.15 NppStatus nppsConvert_32s32f (const Npp32s ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

7.12.1.16 NppStatus nppsConvert_32s32f_Sfs (const Npp32s ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*, int *nScaleFactor*)

7.12.1.17 NppStatus nppsConvert_32s64f (const Npp32s ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*)

7.12.1.18 NppStatus nppsConvert_32s64f_Sfs (const Npp32s ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*, int *nScaleFactor*)

7.12.1.19 NppStatus nppsConvert_64f16s_Sfs (const Npp64f ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, NppRoundMode *eRoundMode*, int *nScaleFactor*)

7.12.1.20 NppStatus nppsConvert_64f32f (const Npp64f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

7.12.1.21 NppStatus nppsConvert_64f32s_Sfs (const Npp64f ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*, NppRoundMode *eRoundMode*, int *nScaleFactor*)

7.12.1.22 NppStatus nppsConvert_64f64s_Sfs (const Npp64f ∗ *pSrc*, Npp64s ∗ *pDst*, int *nLength*, NppRoundMode *eRoundMode*, int *nScaleFactor*)

7.12.1.23 NppStatus nppsConvert_64s32s_Sfs (const Npp64s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*,

7.13 Threshold

Threshold Functions

Performs the threshold operation on the samples of a signal by limiting the sample values by a specified constant value.

- NppStatus nppsThreshold_16s (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, Npp16s nLevel, NppCmpOp nRelOp)

  16-bit signed short signal threshold with constant level.

- NppStatus nppsThreshold_16s_I (Npp16s ∗pSrcDst, int nLength, Npp16s nLevel, NppCmpOp nRelOp)

  16-bit in place signed short signal threshold with constant level.

- NppStatus nppsThreshold_16sc (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength, Npp16s nLevel, NppCmpOp nRelOp)

  16-bit signed short complex number signal threshold with constant level.

- NppStatus nppsThreshold_16sc_I (Npp16sc ∗pSrcDst, int nLength, Npp16s nLevel, NppCmpOp nRelOp)

  16-bit in place signed short complex number signal threshold with constant level.

- NppStatus nppsThreshold_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength, Npp32f nLevel, NppCmpOp nRelOp)

  32-bit floating point signal threshold with constant level.

- NppStatus nppsThreshold_32f_I (Npp32f ∗pSrcDst, int nLength, Npp32f nLevel, NppCmpOp nRelOp)

  32-bit in place floating point signal threshold with constant level.

- NppStatus nppsThreshold_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength, Npp32f nLevel, NppCmpOp nRelOp)

  32-bit floating point complex number signal threshold with constant level.

- NppStatus nppsThreshold_32fc_I (Npp32fc ∗pSrcDst, int nLength, Npp32f nLevel, NppCmpOp nRelOp)

  32-bit in place floating point complex number signal threshold with constant level.

- NppStatus nppsThreshold_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength, Npp64f nLevel, NppCmpOp nRelOp)

  64-bit floating point signal threshold with constant level.

- NppStatus nppsThreshold_64f_I (Npp64f ∗pSrcDst, int nLength, Npp64f nLevel, NppCmpOp nRelOp)

  64-bit in place floating point signal threshold with constant level.

- NppStatus nppsThreshold_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength, Npp64f nLevel, NppCmpOp nRelOp)

  64-bit floating point complex number signal threshold with constant level.

- NppStatus nppsThreshold_64fc_I (Npp64fc ∗pSrcDst, int nLength, Npp64f nLevel, NppCmpOp nRelOp)

  64-bit in place floating point complex number signal threshold with constant level.

- NppStatus nppsThreshold_LT_16s (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, Npp16s nLevel)

  16-bit signed short signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LT_16s_I (Npp16s ∗pSrcDst, int nLength, Npp16s nLevel)

  16-bit in place signed short signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LT_16sc (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength, Npp16s nLevel)

  16-bit signed short complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LT_16sc_I (Npp16sc ∗pSrcDst, int nLength, Npp16s nLevel)

  16-bit in place signed short complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LT_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength, Npp32f nLevel)

  32-bit floating point signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LT_32f_I (Npp32f ∗pSrcDst, int nLength, Npp32f nLevel)

  32-bit in place floating point signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LT_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength, Npp32f nLevel)

  32-bit floating point complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LT_32fc_I (Npp32fc ∗pSrcDst, int nLength, Npp32f nLevel)

  32-bit in place floating point complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LT_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength, Npp64f nLevel)

  64-bit floating point signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LT_64f_I (Npp64f ∗pSrcDst, int nLength, Npp64f nLevel)

  64-bit in place floating point signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LT_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength, Npp64f nLevel)

  64-bit floating point complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LT_64fc_I (Npp64fc ∗pSrcDst, int nLength, Npp64f nLevel)

  64-bit in place floating point complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_GT_16s (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, Npp16s nLevel)

  16-bit signed short signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GT_16s_I (Npp16s ∗pSrcDst, int nLength, Npp16s nLevel)

    16-bit in place signed short signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GT_16sc (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength, Npp16s nLevel)

    16-bit signed short complex number signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GT_16sc_I (Npp16sc ∗pSrcDst, int nLength, Npp16s nLevel)

    16-bit in place signed short complex number signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GT_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength, Npp32f nLevel)

    32-bit floating point signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GT_32f_I (Npp32f ∗pSrcDst, int nLength, Npp32f nLevel)

    32-bit in place floating point signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GT_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength, Npp32f nLevel)

    32-bit floating point complex number signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GT_32fc_I (Npp32fc ∗pSrcDst, int nLength, Npp32f nLevel)

    32-bit in place floating point complex number signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GT_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength, Npp64f nLevel)

    64-bit floating point signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GT_64f_I (Npp64f ∗pSrcDst, int nLength, Npp64f nLevel)

    64-bit in place floating point signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GT_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength, Npp64f nLevel)

    64-bit floating point complex number signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GT_64fc_I (Npp64fc ∗pSrcDst, int nLength, Npp64f nLevel)

    64-bit in place floating point complex number signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_LTVal_16s (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, Npp16s nLevel, Npp16s nValue)

    16-bit signed short signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LTVal_16s_I (Npp16s ∗pSrcDst, int nLength, Npp16s nLevel, Npp16s nValue)

    16-bit in place signed short signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LTVal_16sc (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength, Npp16s nLevel, Npp16sc nValue)

    16-bit signed short complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LTVal_16sc_I (Npp16sc ∗pSrcDst, int nLength, Npp16s nLevel, Npp16sc nValue)

  16-bit in place signed short complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LTVal_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength, Npp32f nLevel, Npp32f nValue)

  32-bit floating point signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LTVal_32f_I (Npp32f ∗pSrcDst, int nLength, Npp32f nLevel, Npp32f nValue)

  32-bit in place floating point signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LTVal_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength, Npp32f nLevel, Npp32fc nValue)

  32-bit floating point complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LTVal_32fc_I (Npp32fc ∗pSrcDst, int nLength, Npp32f nLevel, Npp32fc nValue)

  32-bit in place floating point complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LTVal_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength, Npp64f nLevel, Npp64f nValue)

  64-bit floating point signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LTVal_64f_I (Npp64f ∗pSrcDst, int nLength, Npp64f nLevel, Npp64f nValue)

  64-bit in place floating point signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LTVal_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength, Npp64f nLevel, Npp64fc nValue)

  64-bit floating point complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_LTVal_64fc_I (Npp64fc ∗pSrcDst, int nLength, Npp64f nLevel, Npp64fc nValue)

  64-bit in place floating point complex number signal NPP_CMP_LESS threshold with constant level.

- NppStatus nppsThreshold_GTVal_16s (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, Npp16s nLevel, Npp16s nValue)

  16-bit signed short signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GTVal_16s_I (Npp16s ∗pSrcDst, int nLength, Npp16s nLevel, Npp16s nValue)

  16-bit in place signed short signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GTVal_16sc (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength, Npp16s nLevel, Npp16sc nValue)

  16-bit signed short complex number signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GTVal_16sc_I (Npp16sc ∗pSrcDst, int nLength, Npp16s nLevel, Npp16sc nValue)

  16-bit in place signed short complex number signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GTVal_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength, Npp32f nLevel, Npp32f nValue)

  32-bit floating point signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GTVal_32f_I (Npp32f ∗pSrcDst, int nLength, Npp32f nLevel, Npp32f nValue)

  32-bit in place floating point signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GTVal_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength, Npp32f nLevel, Npp32fc nValue)

  32-bit floating point complex number signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GTVal_32fc_I (Npp32fc ∗pSrcDst, int nLength, Npp32f nLevel, Npp32fc nValue)

  32-bit in place floating point complex number signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GTVal_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength, Npp64f nLevel, Npp64f nValue)

  64-bit floating point signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GTVal_64f_I (Npp64f ∗pSrcDst, int nLength, Npp64f nLevel, Npp64f nValue)

  64-bit in place floating point signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GTVal_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength, Npp64f nLevel, Npp64fc nValue)

  64-bit floating point complex number signal NPP_CMP_GREATER threshold with constant level.

- NppStatus nppsThreshold_GTVal_64fc_I (Npp64fc ∗pSrcDst, int nLength, Npp64f nLevel, Npp64fc nValue)

  64-bit in place floating point complex number signal NPP_CMP_GREATER threshold with constant level.

7.13.1 Function Documentation

7.13.1.1 NppStatus nppsThreshold_16s (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, Npp16s *nLevel*, NppCmpOp *nRelOp*)

16-bit signed short signal threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.2 NppStatus nppsThreshold_16s_I (Npp16s ∗ *pSrcDst*, int *nLength*, Npp16s *nLevel*, NppCmpOp *nRelOp*)

16-bit in place signed short signal threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.3 NppStatus nppsThreshold_16sc (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pDst*, int *nLength*, Npp16s *nLevel*, NppCmpOp *nRelOp*)

16-bit signed short complex number signal threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.4 NppStatus nppsThreshold_16sc_I (Npp16sc ∗ *pSrcDst*, int *nLength*, Npp16s *nLevel*, NppCmpOp *nRelOp*)

16-bit in place signed short complex number signal threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.5 NppStatus nppsThreshold_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*, Npp32f *nLevel*, NppCmpOp *nRelOp*)

32-bit floating point signal threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.6 NppStatus nppsThreshold_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*, Npp32f *nLevel*, NppCmpOp *nRelOp*)

32-bit in place floating point signal threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.7 NppStatus nppsThreshold_32fc (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pDst*, int *nLength*, Npp32f *nLevel*, NppCmpOp *nRelOp*)

32-bit floating point complex number signal threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.8 NppStatus nppsThreshold_32fc_I (Npp32fc ∗ *pSrcDst*, int *nLength*, Npp32f *nLevel*, NppCmpOp *nRelOp*)

32-bit in place floating point complex number signal threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.9 NppStatus nppsThreshold_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*, Npp64f *nLevel*, NppCmpOp *nRelOp*)

64-bit floating point signal threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.10 NppStatus nppsThreshold_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*, Npp64f *nLevel*, NppCmpOp *nRelOp*)

64-bit in place floating point signal threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.11 NppStatus nppsThreshold_64fc (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pDst*, int *nLength*, Npp64f *nLevel*, NppCmpOp *nRelOp*)

64-bit floating point complex number signal threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.12 NppStatus nppsThreshold_64fc_I (Npp64fc ∗ *pSrcDst*, int *nLength*, Npp64f *nLevel*, NppCmpOp *nRelOp*)

64-bit in place floating point complex number signal threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nRelOp NppCmpOp type of thresholding operation (NPP_CMP_LESS or NPP_CMP_GREATER only).

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.13 NppStatus nppsThreshold_GT_16s (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, Npp16s *nLevel*)

16-bit signed short signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.14 NppStatus nppsThreshold_GT_16s_I (Npp16s ∗ *pSrcDst*, int *nLength*, Npp16s *nLevel*)

16-bit in place signed short signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.15 NppStatus nppsThreshold_GT_16sc (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pDst*, int *nLength*, Npp16s *nLevel*)

16-bit signed short complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.16 NppStatus nppsThreshold_GT_16sc_I (Npp16sc ∗ *pSrcDst*, int *nLength*, Npp16s *nLevel*)

16-bit in place signed short complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.17 NppStatus nppsThreshold_GT_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*, Npp32f *nLevel*)

32-bit floating point signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.18 NppStatus nppsThreshold_GT_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*, Npp32f *nLevel*)

32-bit in place floating point signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.19 NppStatus nppsThreshold_GT_32fc (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pDst*, int *nLength*, Npp32f *nLevel*)

32-bit floating point complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.20 NppStatus nppsThreshold_GT_32fc_I (Npp32fc ∗ *pSrcDst*, int *nLength*, Npp32f *nLevel*)

32-bit in place floating point complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.21 NppStatus nppsThreshold_GT_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*, Npp64f *nLevel*)

64-bit floating point signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.22 NppStatus nppsThreshold_GT_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*, Npp64f *nLevel*)

64-bit in place floating point signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.23 NppStatus nppsThreshold_GT_64fc (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pDst*, int *nLength*, Npp64f *nLevel*)

64-bit floating point complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.24 NppStatus nppsThreshold_GT_64fc_I (Npp64fc ∗ *pSrcDst*, int *nLength*, Npp64f *nLevel*)

64-bit in place floating point complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.25 NppStatus nppsThreshold_GTVal_16s (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, Npp16s *nLevel*, Npp16s *nValue*)

16-bit signed short signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.26 NppStatus nppsThreshold_GTVal_16s_I (Npp16s ∗ *pSrcDst*, int *nLength*, Npp16s *nLevel*, Npp16s *nValue*)

16-bit in place signed short signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.27 NppStatus nppsThreshold_GTVal_16sc (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pDst*, int *nLength*, Npp16s *nLevel*, Npp16sc *nValue*)

16-bit signed short complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.28 NppStatus nppsThreshold_GTVal_16sc_I (Npp16sc ∗ *pSrcDst*, int *nLength*, Npp16s *nLevel*, Npp16sc *nValue*)

16-bit in place signed short complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.29 NppStatus nppsThreshold_GTVal_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*, Npp32f *nLevel*, Npp32f *nValue*)

32-bit floating point signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.30 NppStatus nppsThreshold_GTVal_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*, Npp32f *nLevel*, Npp32f *nValue*)

32-bit in place floating point signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.31 NppStatus nppsThreshold_GTVal_32fc (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pDst*, int *nLength*, Npp32f *nLevel*, Npp32fc *nValue*)

32-bit floating point complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.32 NppStatus nppsThreshold_GTVal_32fc_I (Npp32fc ∗ *pSrcDst*, int *nLength*, Npp32f *nLevel*, Npp32fc *nValue*)

32-bit in place floating point complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.33 NppStatus nppsThreshold_GTVal_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*, Npp64f *nLevel*, Npp64f *nValue*)

64-bit floating point signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.34 NppStatus nppsThreshold_GTVal_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*, Npp64f *nLevel*, Npp64f *nValue*)

64-bit in place floating point signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.35 NppStatus nppsThreshold_GTVal_64fc (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pDst*, int *nLength*, Npp64f *nLevel*, Npp64fc *nValue*)

64-bit floating point complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.36 NppStatus nppsThreshold_GTVal_64fc_I (Npp64fc ∗ *pSrcDst*, int *nLength*, Npp64f *nLevel*, Npp64fc *nValue*)

64-bit in place floating point complex number signal NPP_CMP_GREATER threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.37 NppStatus nppsThreshold_LT_16s (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, Npp16s *nLevel*)

16-bit signed short signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.38 NppStatus nppsThreshold_LT_16s_I (Npp16s ∗ *pSrcDst*, int *nLength*, Npp16s *nLevel*)

16-bit in place signed short signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.39 NppStatus nppsThreshold_LT_16sc (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pDst*, int *nLength*, Npp16s *nLevel*)

16-bit signed short complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.40 NppStatus nppsThreshold_LT_16sc_I (Npp16sc ∗ *pSrcDst*, int *nLength*, Npp16s *nLevel*)

16-bit in place signed short complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.41 NppStatus nppsThreshold_LT_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*, Npp32f *nLevel*)

32-bit floating point signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.42 NppStatus nppsThreshold_LT_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*, Npp32f *nLevel*)

32-bit in place floating point signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.43 NppStatus nppsThreshold_LT_32fc (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pDst*, int *nLength*, Npp32f *nLevel*)

32-bit floating point complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.44 NppStatus nppsThreshold_LT_32fc_I (Npp32fc ∗ *pSrcDst*, int *nLength*, Npp32f *nLevel*)

32-bit in place floating point complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.45 NppStatus nppsThreshold_LT_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*, Npp64f *nLevel*)

64-bit floating point signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.46 NppStatus nppsThreshold_LT_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*, Npp64f *nLevel*)

64-bit in place floating point signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.47 NppStatus nppsThreshold_LT_64fc (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pDst*, int *nLength*, Npp64f *nLevel*)

64-bit floating point complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.48 NppStatus nppsThreshold_LT_64fc_I (Npp64fc ∗ *pSrcDst*, int *nLength*, Npp64f *nLevel*)

64-bit in place floating point complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.49 NppStatus nppsThreshold_LTVal_16s (const Npp16s $*$ *pSrc*, Npp16s $*$ *pDst*, int *nLength*, Npp16s *nLevel*, Npp16s *nValue*)

16-bit signed short signal NPP_CMP_LESS threshold with constant level.

Parameters:

- ***pSrc*** Source Signal Pointer.
- ***pDst*** Destination Signal Pointer.
- ***nLength*** Signal Length.
- ***nLevel*** Constant threshold value to be used to limit each signal sample
- ***nValue*** Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.50 NppStatus nppsThreshold_LTVal_16s_I (Npp16s $*$ *pSrcDst*, int *nLength*, Npp16s *nLevel*, Npp16s *nValue*)

16-bit in place signed short signal NPP_CMP_LESS threshold with constant level.

Parameters:

- ***pSrcDst*** In-Place Signal Pointer.
- ***nLength*** Signal Length.
- ***nLevel*** Constant threshold value to be used to limit each signal sample
- ***nValue*** Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.51 NppStatus nppsThreshold_LTVal_16sc (const Npp16sc $*$ *pSrc*, Npp16sc $*$ *pDst*, int *nLength*, Npp16s *nLevel*, Npp16sc *nValue*)

16-bit signed short complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

- ***pSrc*** Source Signal Pointer.
- ***pDst*** Destination Signal Pointer.
- ***nLength*** Signal Length.
- ***nLevel*** Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample
- ***nValue*** Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.52 NppStatus nppsThreshold_LTVal_16sc_I (Npp16sc ∗ *pSrcDst*, int *nLength*, Npp16s *nLevel*, Npp16sc *nValue*)

16-bit in place signed short complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.53 NppStatus nppsThreshold_LTVal_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*, Npp32f *nLevel*, Npp32f *nValue*)

32-bit floating point signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.54 NppStatus nppsThreshold_LTVal_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*, Npp32f *nLevel*, Npp32f *nValue*)

32-bit in place floating point signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.55 NppStatus nppsThreshold_LTVal_32fc (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pDst*, int *nLength*, Npp32f *nLevel*, Npp32fc *nValue*)

32-bit floating point complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.56 NppStatus nppsThreshold_LTVal_32fc_I (Npp32fc ∗ *pSrcDst*, int *nLength*, Npp32f *nLevel*, Npp32fc *nValue*)

32-bit in place floating point complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.57 NppStatus nppsThreshold_LTVal_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*, Npp64f *nLevel*, Npp64f *nValue*)

64-bit floating point signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.58 NppStatus nppsThreshold_LTVal_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*, Npp64f *nLevel*, Npp64f *nValue*)

64-bit in place floating point signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.59 NppStatus nppsThreshold_LTVal_64fc (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pDst*, int *nLength*, Npp64f *nLevel*, Npp64fc *nValue*)

64-bit floating point complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.13.1.60 NppStatus nppsThreshold_LTVal_64fc_I (Npp64fc ∗ *pSrcDst*, int *nLength*, Npp64f *nLevel*, Npp64fc *nValue*)

64-bit in place floating point complex number signal NPP_CMP_LESS threshold with constant level.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nLevel Constant threshold value (real part only and must be greater than 0) to be used to limit each signal sample

nValue Constant value to replace source value when threshold test is true.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.14 Arithmetic and Logical Operations

Modules

- Arithmetic Operations
- Logical And Shift Operations

7.15 Arithmetic Operations

Modules

- AddC

  Adds a constant value to each sample of a signal.

- AddProductC

  Adds product of a constant and each sample of a source signal to the each sample of destination signal.

- MulC

  Multiplies each sample of a signal by a constant value.

- SubC

  Subtracts a constant from each sample of a signal.

- SubCRev

  Subtracts each sample of a signal from a constant.

- DivC

  Divides each sample of a signal by a constant.

- DivCRev

  Divides a constant by each sample of a signal.

- Add

  Sample by sample addition of two signals.

- AddProduct

  Adds sample by sample product of two signals to the destination signal.

- Mul

  Sample by sample multiplication the samples of two signals.

- Sub

  Sample by sample subtraction of the samples of two signals.

- Div

  Sample by sample division of the samples of two signals.

- Div_Round

  Sample by sample division of the samples of two signals with rounding.

- Abs

  Absolute value of each sample of a signal.

- Sqr

  Squares each sample of a signal.

- Sqrt

Square root of each sample of a signal.

- Cubrt

  Cube root of each sample of a signal.

- Exp

  E raised to the power of each sample of a signal.

- Ln

  Natural logarithm of each sample of a signal.

- 10Log10

  Ten times the decimal logarithm of each sample of a signal.

- SumLn

  Sums up the natural logarithm of each sample of a signal.

- Arctan

  Inverse tangent of each sample of a signal.

- Normalize

  Normalize each sample of a real or complex signal using offset and division operations.

- Cauchy, CauchyD, and CauchyDD2

  Determine Cauchy robust error function and its first and second derivatives for each sample of a signal.

7.16 AddC

Adds a constant value to each sample of a signal.

Functions

- NppStatus nppsAddC_8u_ISfs (Npp8u nValue, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal add constant, scale, then clamp to saturated value

- NppStatus nppsAddC_8u_Sfs (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned charvector add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16u_ISfs (Npp16u nValue, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16u_Sfs (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short vector add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16s_ISfs (Npp16s nValue, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16s_Sfs (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16sc_ISfs (Npp16sc nValue, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary)signal add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc nValue, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal add constant, scale, then clamp to saturated value.

- NppStatus nppsAddC_32s_ISfs (Npp32s nValue, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal add constant and scale.

- NppStatus nppsAddC_32s_Sfs (const Npp32s ∗pSrc, Npp32s nValue, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integersignal add constant and scale.

- NppStatus nppsAddC_32sc_ISfs (Npp32sc nValue, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal add constant and scale.

- NppStatus nppsAddC_32sc_Sfs (const Npp32sc ∗pSrc, Npp32sc nValue, Npp32sc ∗pDst, int nLength, int nScaleFactor)

32-bit integer complex number (32 bit real, 32 bit imaginary) signal add constant and scale.

- NppStatus nppsAddC_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal add constant.

- NppStatus nppsAddC_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)

  32-bit floating point signal add constant.

- NppStatus nppsAddC_32fc_I (Npp32fc nValue, Npp32fc ∗pSrcDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal add constant.

- NppStatus nppsAddC_32fc (const Npp32fc ∗pSrc, Npp32fc nValue, Npp32fc ∗pDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) signal add constant.

- NppStatus nppsAddC_64f_I (Npp64f nValue, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point, in place signal add constant.

- NppStatus nppsAddC_64f (const Npp64f ∗pSrc, Npp64f nValue, Npp64f ∗pDst, int nLength)

  64-bit floating pointsignal add constant.

- NppStatus nppsAddC_64fc_I (Npp64fc nValue, Npp64fc ∗pSrcDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal add constant.

- NppStatus nppsAddC_64fc (const Npp64fc ∗pSrc, Npp64fc nValue, Npp64fc ∗pDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) signal add constant.

7.16.1 Detailed Description

Adds a constant value to each sample of a signal.

7.16.2 Function Documentation

7.16.2.1 NppStatus nppsAddC_16s_ISfs (Npp16s *nValue*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal add constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.2 NppStatus nppsAddC_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s *nValue*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal add constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.3 NppStatus nppsAddC_16sc_ISfs (Npp16sc *nValue*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary)signal add constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.4 NppStatus nppsAddC_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc *nValue*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal add constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.5 NppStatus nppsAddC_16u_ISfs (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal add constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.6 NppStatus nppsAddC_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short vector add constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.7 NppStatus nppsAddC_32f (const Npp32f ∗ *pSrc*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal add constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.8 NppStatus nppsAddC_32f_I (Npp32f *nValue*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal add constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.9 NppStatus nppsAddC_32fc (const Npp32fc ∗ *pSrc*, Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) signal add constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.10 NppStatus nppsAddC_32fc_I (Npp32fc *nValue*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal add constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.11 NppStatus nppsAddC_32s_ISfs (Npp32s *nValue*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal add constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.12 NppStatus nppsAddC_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s *nValue*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integersignal add constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.13 NppStatus nppsAddC_32sc_ISfs (Npp32sc *nValue*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal add constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.14 NppStatus nppsAddC_32sc_Sfs (const Npp32sc ∗ *pSrc*, Npp32sc *nValue*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) signal add constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.15 NppStatus nppsAddC_64f (const Npp64f ∗ *pSrc*, Npp64f *nValue*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating pointsignal add constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.16 NppStatus nppsAddC_64f_I (Npp64f *nValue*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point, in place signal add constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Length of the vectors, number of items.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.17 NppStatus nppsAddC_64fc (const Npp64fc ∗ *pSrc*, Npp64fc *nValue*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) signal add constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.18 NppStatus nppsAddC_64fc_I (Npp64fc *nValue*, Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal add constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.19 NppStatus nppsAddC_8u_ISfs (Npp8u *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal add constant, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be added to each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.16.2.20 NppStatus nppsAddC_8u_Sfs (const Npp8u * *pSrc*, Npp8u *nValue*, Npp8u * *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned charvector add constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be added to each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.17 AddProductC

Adds product of a constant and each sample of a source signal to the each sample of destination signal.

Functions

- NppStatus nppsAddProductC_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)

  32-bit floating point signal add product of signal times constant to destination signal.

7.17.1 Detailed Description

Adds product of a constant and each sample of a source signal to the each sample of destination signal.

7.17.2 Function Documentation

7.17.2.1 NppStatus nppsAddProductC_32f (const Npp32f ∗ *pSrc*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal add product of signal times constant to destination signal.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18 MulC

Multiplies each sample of a signal by a constant value.

Functions

- NppStatus nppsMulC_8u_ISfs (Npp8u nValue, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal times constant, scale, then clamp to saturated value

- NppStatus nppsMulC_8u_Sfs (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16u_ISfs (Npp16u nValue, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16u_Sfs (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16s_ISfs (Npp16s nValue, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16s_Sfs (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16sc_ISfs (Npp16sc nValue, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary)signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc nValue, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary)signal times constant, scale, then clamp to saturated value.

- NppStatus nppsMulC_32s_ISfs (Npp32s nValue, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal times constant and scale.

- NppStatus nppsMulC_32s_Sfs (const Npp32s ∗pSrc, Npp32s nValue, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal times constant and scale.

- NppStatus nppsMulC_32sc_ISfs (Npp32sc nValue, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal times constant and scale.

- NppStatus nppsMulC_32sc_Sfs (const Npp32sc ∗pSrc, Npp32sc nValue, Npp32sc ∗pDst, int nLength, int nScaleFactor)

32-bit integer complex number (32 bit real, 32 bit imaginary) signal times constant and scale.

- NppStatus nppsMulC_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal times constant.

- NppStatus nppsMulC_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)

  32-bit floating point signal times constant.

- NppStatus nppsMulC_Low_32f16s (const Npp32f ∗pSrc, Npp32f nValue, Npp16s ∗pDst, int nLength)

  32-bit floating point signal times constant with output converted to 16-bit signed integer.

- NppStatus nppsMulC_32f16s_Sfs (const Npp32f ∗pSrc, Npp32f nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  32-bit floating point signal times constant with output converted to 16-bit signed integer with scaling and saturation of output result.

- NppStatus nppsMulC_32fc_I (Npp32fc nValue, Npp32fc ∗pSrcDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal times constant.

- NppStatus nppsMulC_32fc (const Npp32fc ∗pSrc, Npp32fc nValue, Npp32fc ∗pDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) signal times constant.

- NppStatus nppsMulC_64f_I (Npp64f nValue, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point, in place signal times constant.

- NppStatus nppsMulC_64f (const Npp64f ∗pSrc, Npp64f nValue, Npp64f ∗pDst, int nLength)

  64-bit floating point signal times constant.

- NppStatus nppsMulC_64f64s_ISfs (Npp64f nValue, Npp64s ∗pDst, int nLength, int nScaleFactor)

  64-bit floating point signal times constant with in place conversion to 64-bit signed integer and with scaling and saturation of output result.

- NppStatus nppsMulC_64fc_I (Npp64fc nValue, Npp64fc ∗pSrcDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal times constant.

- NppStatus nppsMulC_64fc (const Npp64fc ∗pSrc, Npp64fc nValue, Npp64fc ∗pDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) signal times constant.

7.18.1 Detailed Description

Multiplies each sample of a signal by a constant value.

7.18.2 Function Documentation

7.18.2.1 NppStatus nppsMulC_16s_ISfs (Npp16s *nValue*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal times constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.2 NppStatus nppsMulC_16s_Sfs (const Npp16s * *pSrc*, Npp16s *nValue*, Npp16s * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal times constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.3 NppStatus nppsMulC_16sc_ISfs (Npp16sc *nValue*, Npp16sc * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary)signal times constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.4 NppStatus nppsMulC_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc *nValue*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary)signal times constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.5 NppStatus nppsMulC_16u_ISfs (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal times constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.6 NppStatus nppsMulC_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal times constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.7 NppStatus nppsMulC_32f (const Npp32f * *pSrc*, Npp32f *nValue*, Npp32f * *pDst*, int *nLength*)

32-bit floating point signal times constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.8 NppStatus nppsMulC_32f16s_Sfs (const Npp32f * *pSrc*, Npp32f *nValue*, Npp16s * *pDst*, int *nLength*, int *nScaleFactor*)

32-bit floating point signal times constant with output converted to 16-bit signed integer with scaling and saturation of output result.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nScaleFactor Integer Result Scaling.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.9 NppStatus nppsMulC_32f_I (Npp32f *nValue*, Npp32f * *pSrcDst*, int *nLength*)

32-bit floating point in place signal times constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.10 NppStatus nppsMulC_32fc (const Npp32fc ∗ *pSrc*, Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) signal times constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.11 NppStatus nppsMulC_32fc_I (Npp32fc *nValue*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal times constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.12 NppStatus nppsMulC_32s_ISfs (Npp32s *nValue*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal times constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.13 NppStatus nppsMulC_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s *nValue*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal times constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.14 NppStatus nppsMulC_32sc_ISfs (Npp32sc *nValue*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal times constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.15 NppStatus nppsMulC_32sc_Sfs (const Npp32sc ∗ *pSrc*, Npp32sc *nValue*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) signal times constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.16 NppStatus nppsMulC_64f (const Npp64f ∗ *pSrc*, Npp64f *nValue*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal times constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.17 NppStatus nppsMulC_64f64s_ISfs (Npp64f *nValue*, Npp64s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

64-bit floating point signal times constant with in place conversion to 64-bit signed integer and with scaling and saturation of output result.

Parameters:

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.18 NppStatus nppsMulC_64f_I (Npp64f *nValue*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point, in place signal times constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Length of the vectors, number of items.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.19 NppStatus nppsMulC_64fc (const Npp64fc * *pSrc*, Npp64fc *nValue*, Npp64fc * *pDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) signal times constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.20 NppStatus nppsMulC_64fc_I (Npp64fc *nValue*, Npp64fc * *pSrcDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal times constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.21 NppStatus nppsMulC_8u_ISfs (Npp8u *nValue*, Npp8u * *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal times constant, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be multiplied by each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.22 NppStatus nppsMulC_8u_Sfs (const Npp8u * *pSrc*, Npp8u *nValue*, Npp8u * *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal times constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.18.2.23 NppStatus nppsMulC_Low_32f16s (const Npp32f * *pSrc*, Npp32f *nValue*, Npp16s * *pDst*, int *nLength*)

32-bit floating point signal times constant with output converted to 16-bit signed integer.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be multiplied by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19 SubC

Subtracts a constant from each sample of a signal.

Functions

- NppStatus nppsSubC_8u_ISfs (Npp8u nValue, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal subtract constant, scale, then clamp to saturated value

- NppStatus nppsSubC_8u_Sfs (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16u_ISfs (Npp16u nValue, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16u_Sfs (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16s_ISfs (Npp16s nValue, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16s_Sfs (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16sc_ISfs (Npp16sc nValue, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc nValue, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract constant, scale, then clamp to saturated value.

- NppStatus nppsSubC_32s_ISfs (Npp32s nValue, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal subtract constant and scale.

- NppStatus nppsSubC_32s_Sfs (const Npp32s ∗pSrc, Npp32s nValue, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal subtract constant and scale.

- NppStatus nppsSubC_32sc_ISfs (Npp32sc nValue, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal subtract constant and scale.

- NppStatus nppsSubC_32sc_Sfs (const Npp32sc ∗pSrc, Npp32sc nValue, Npp32sc ∗pDst, int nLength, int nScaleFactor)

32-bit integer complex number (32 bit real, 32 bit imaginary)signal subtract constant and scale.

- NppStatus nppsSubC_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)
  32-bit floating point in place signal subtract constant.
- NppStatus nppsSubC_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)
  32-bit floating point signal subtract constant.
- NppStatus nppsSubC_32fc_I (Npp32fc nValue, Npp32fc ∗pSrcDst, int nLength)
  32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal subtract constant.
- NppStatus nppsSubC_32fc (const Npp32fc ∗pSrc, Npp32fc nValue, Npp32fc ∗pDst, int nLength)
  32-bit floating point complex number (32 bit real, 32 bit imaginary) signal subtract constant.
- NppStatus nppsSubC_64f_I (Npp64f nValue, Npp64f ∗pSrcDst, int nLength)
  64-bit floating point, in place signal subtract constant.
- NppStatus nppsSubC_64f (const Npp64f ∗pSrc, Npp64f nValue, Npp64f ∗pDst, int nLength)
  64-bit floating point signal subtract constant.
- NppStatus nppsSubC_64fc_I (Npp64fc nValue, Npp64fc ∗pSrcDst, int nLength)
  64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal subtract constant.
- NppStatus nppsSubC_64fc (const Npp64fc ∗pSrc, Npp64fc nValue, Npp64fc ∗pDst, int nLength)
  64-bit floating point complex number (64 bit real, 64 bit imaginary) signal subtract constant.

7.19.1 Detailed Description

Subtracts a constant from each sample of a signal.

7.19.2 Function Documentation

7.19.2.1 NppStatus nppsSubC_16s_ISfs (Npp16s *nValue*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.2 NppStatus nppsSubC_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s *nValue*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be subtracted from each vector element
pDst Destination Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.3 NppStatus nppsSubC_16sc_ISfs (Npp16sc *nValue*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be subtracted from each vector element
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.4 NppStatus nppsSubC_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc *nValue*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be subtracted from each vector element
pDst Destination Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.5 NppStatus nppsSubC_16u_ISfs (Npp16u *nValue*, Npp16u * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.6 NppStatus nppsSubC_16u_Sfs (const Npp16u * *pSrc*, Npp16u *nValue*, Npp16u * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.7 NppStatus nppsSubC_32f (const Npp32f * *pSrc*, Npp32f *nValue*, Npp32f * *pDst*, int *nLength*)

32-bit floating point signal subtract constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.8 NppStatus nppsSubC_32f_I (Npp32f *nValue*, Npp32f * *pSrcDst*, int *nLength*)

32-bit floating point in place signal subtract constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.9 NppStatus nppsSubC_32fc (const Npp32fc * *pSrc*, Npp32fc *nValue*, Npp32fc * *pDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) signal subtract constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.10 NppStatus nppsSubC_32fc_I (Npp32fc *nValue*, Npp32fc * *pSrcDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal subtract constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.11 NppStatus nppsSubC_32s_ISfs (Npp32s *nValue*, Npp32s * *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal subtract constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.12 NppStatus nppsSubC_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s *nValue*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal subtract constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.13 NppStatus nppsSubC_32sc_ISfs (Npp32sc *nValue*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal subtract constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.14 NppStatus nppsSubC_32sc_Sfs (const Npp32sc ∗ *pSrc*, Npp32sc *nValue*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary)signal subtract constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.15 NppStatus nppsSubC_64f (const Npp64f ∗ *pSrc*, Npp64f *nValue*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal subtract constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.16 NppStatus nppsSubC_64f_I (Npp64f *nValue*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point, in place signal subtract constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Length of the vectors, number of items.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.17 NppStatus nppsSubC_64fc (const Npp64fc ∗ *pSrc*, Npp64fc *nValue*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) signal subtract constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.18 NppStatus nppsSubC_64fc_I (Npp64fc *nValue*, Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal subtract constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.19 NppStatus nppsSubC_8u_ISfs (Npp8u *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal subtract constant, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be subtracted from each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.19.2.20 NppStatus nppsSubC_8u_Sfs (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal subtract constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be subtracted from each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20 SubCRev

Subtracts each sample of a signal from a constant.

Functions

- NppStatus nppsSubCRev_8u_ISfs (Npp8u nValue, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal subtract from constant, scale, then clamp to saturated value

- NppStatus nppsSubCRev_8u_Sfs (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16u_ISfs (Npp16u nValue, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16u_Sfs (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16s_ISfs (Npp16s nValue, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16s_Sfs (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16sc_ISfs (Npp16sc nValue, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc nValue, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract from constant, scale, then clamp to saturated value.

- NppStatus nppsSubCRev_32s_ISfs (Npp32s nValue, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal subtract from constant and scale.

- NppStatus nppsSubCRev_32s_Sfs (const Npp32s ∗pSrc, Npp32s nValue, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integersignal subtract from constant and scale.

- NppStatus nppsSubCRev_32sc_ISfs (Npp32sc nValue, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal subtract from constant and scale.

- NppStatus nppsSubCRev_32sc_Sfs (const Npp32sc *pSrc, Npp32sc nValue, Npp32sc *pDst, int nLength, int nScaleFactor)

  32-bit integer complex number (32 bit real, 32 bit imaginary) signal subtract from constant and scale.

- NppStatus nppsSubCRev_32f_I (Npp32f nValue, Npp32f *pSrcDst, int nLength)

  32-bit floating point in place signal subtract from constant.

- NppStatus nppsSubCRev_32f (const Npp32f *pSrc, Npp32f nValue, Npp32f *pDst, int nLength)

  32-bit floating point signal subtract from constant.

- NppStatus nppsSubCRev_32fc_I (Npp32fc nValue, Npp32fc *pSrcDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal subtract from constant.

- NppStatus nppsSubCRev_32fc (const Npp32fc *pSrc, Npp32fc nValue, Npp32fc *pDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) signal subtract from constant.

- NppStatus nppsSubCRev_64f_I (Npp64f nValue, Npp64f *pSrcDst, int nLength)

  64-bit floating point, in place signal subtract from constant.

- NppStatus nppsSubCRev_64f (const Npp64f *pSrc, Npp64f nValue, Npp64f *pDst, int nLength)

  64-bit floating point signal subtract from constant.

- NppStatus nppsSubCRev_64fc_I (Npp64fc nValue, Npp64fc *pSrcDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal subtract from constant.

- NppStatus nppsSubCRev_64fc (const Npp64fc *pSrc, Npp64fc nValue, Npp64fc *pDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) signal subtract from constant.

7.20.1 Detailed Description

Subtracts each sample of a signal from a constant.

7.20.2 Function Documentation

7.20.2.1 NppStatus nppsSubCRev_16s_ISfs (Npp16s *nValue*, Npp16s * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.2 NppStatus nppsSubCRev_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s *nValue*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.3 NppStatus nppsSubCRev_16sc_ISfs (Npp16sc *nValue*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.4 NppStatus nppsSubCRev_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc *nValue*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.5 NppStatus nppsSubCRev_16u_ISfs (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.6 NppStatus nppsSubCRev_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.7 NppStatus nppsSubCRev_32f (const Npp32f ∗ *pSrc*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal subtract from constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.8 NppStatus nppsSubCRev_32f_I (Npp32f *nValue*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal subtract from constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.9 NppStatus nppsSubCRev_32fc (const Npp32fc ∗ *pSrc*, Npp32fc *nValue*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) signal subtract from constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.10 NppStatus nppsSubCRev_32fc_I (Npp32fc *nValue*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal subtract from constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.11 NppStatus nppsSubCRev_32s_ISfs (Npp32s *nValue*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal subtract from constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.12 NppStatus nppsSubCRev_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s *nValue*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integersignal subtract from constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.13 NppStatus nppsSubCRev_32sc_ISfs (Npp32sc *nValue*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) in place signal subtract from constant and scale.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.14 NppStatus nppsSubCRev_32sc_Sfs (const Npp32sc ∗ *pSrc*, Npp32sc *nValue*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit integer complex number (32 bit real, 32 bit imaginary) signal subtract from constant and scale.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.15 NppStatus nppsSubCRev_64f (const Npp64f ∗ *pSrc*, Npp64f *nValue*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal subtract from constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.16 NppStatus nppsSubCRev_64f_I (Npp64f *nValue*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point, in place signal subtract from constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Length of the vectors, number of items.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.17 NppStatus nppsSubCRev_64fc (const Npp64fc * *pSrc*, Npp64fc *nValue*, Npp64fc * *pDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) signal subtract from constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.18 NppStatus nppsSubCRev_64fc_I (Npp64fc *nValue*, Npp64fc * *pSrcDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal subtract from constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.19 NppStatus nppsSubCRev_8u_ISfs (Npp8u *nValue*, Npp8u * *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal subtract from constant, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value each vector element is to be subtracted from

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.20.2.20 NppStatus nppsSubCRev_8u_Sfs (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal subtract from constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value each vector element is to be subtracted from

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21 DivC

Divides each sample of a signal by a constant.

Functions

- NppStatus nppsDivC_8u_ISfs (Npp8u nValue, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal divided by constant, scale, then clamp to saturated value

- NppStatus nppsDivC_8u_Sfs (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16u_ISfs (Npp16u nValue, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16u_Sfs (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16s_ISfs (Npp16s nValue, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16s_Sfs (const Npp16s ∗pSrc, Npp16s nValue, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16sc_ISfs (Npp16sc nValue, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary)signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc nValue, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit integer complex number (16 bit real, 16 bit imaginary) signal divided by constant, scale, then clamp to saturated value.

- NppStatus nppsDivC_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal divided by constant.

- NppStatus nppsDivC_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)

  32-bit floating point signal divided by constant.

- NppStatus nppsDivC_32fc_I (Npp32fc nValue, Npp32fc ∗pSrcDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal divided by constant.

- NppStatus nppsDivC_32fc (const Npp32fc ∗pSrc, Npp32fc nValue, Npp32fc ∗pDst, int nLength)

  32-bit floating point complex number (32 bit real, 32 bit imaginary) signal divided by constant.

- NppStatus nppsDivC_64f_I (Npp64f nValue, Npp64f ∗pSrcDst, int nLength)

64-bit floating point in place signal divided by constant.

- NppStatus nppsDivC_64f (const Npp64f ∗pSrc, Npp64f nValue, Npp64f ∗pDst, int nLength)

  64-bit floating point signal divided by constant.
- NppStatus nppsDivC_64fc_I (Npp64fc nValue, Npp64fc ∗pSrcDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal divided by constant.
- NppStatus nppsDivC_64fc (const Npp64fc ∗pSrc, Npp64fc nValue, Npp64fc ∗pDst, int nLength)

  64-bit floating point complex number (64 bit real, 64 bit imaginary) signal divided by constant.

7.21.1 Detailed Description

Divides each sample of a signal by a constant.

7.21.2 Function Documentation

7.21.2.1 NppStatus nppsDivC_16s_ISfs (Npp16s *nValue*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.2 NppStatus nppsDivC_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s *nValue*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.3 NppStatus nppsDivC_16sc_ISfs (Npp16sc *nValue*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary)signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.4 NppStatus nppsDivC_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc *nValue*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit integer complex number (16 bit real, 16 bit imaginary) signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.5 NppStatus nppsDivC_16u_ISfs (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.6 NppStatus nppsDivC_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.7 NppStatus nppsDivC_32f (const Npp32f ∗ *pSrc*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal divided by constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.8 NppStatus nppsDivC_32f_I (Npp32f *nValue*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal divided by constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.9 NppStatus nppsDivC_32fc (const Npp32fc * *pSrc*, Npp32fc *nValue*, Npp32fc * *pDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) signal divided by constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.10 NppStatus nppsDivC_32fc_I (Npp32fc *nValue*, Npp32fc * *pSrcDst*, int *nLength*)

32-bit floating point complex number (32 bit real, 32 bit imaginary) in place signal divided by constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.11 NppStatus nppsDivC_64f (const Npp64f * *pSrc*, Npp64f *nValue*, Npp64f * *pDst*, int *nLength*)

64-bit floating point signal divided by constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.12 NppStatus nppsDivC_64f_I (Npp64f *nValue*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point in place signal divided by constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Length of the vectors, number of items.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.13 NppStatus nppsDivC_64fc (const Npp64fc ∗ *pSrc*, Npp64fc *nValue*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) signal divided by constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.14 NppStatus nppsDivC_64fc_I (Npp64fc *nValue*, Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit floating point complex number (64 bit real, 64 bit imaginary) in place signal divided by constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.15 NppStatus nppsDivC_8u_ISfs (Npp8u *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal divided by constant, scale, then clamp to saturated value

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided into each vector element

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.21.2.16 NppStatus nppsDivC_8u_Sfs (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal divided by constant, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided into each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.22 DivCRev

Divides a constant by each sample of a signal.

Functions

- NppStatus nppsDivCRev_16u_I (Npp16u nValue, Npp16u ∗pSrcDst, int nLength)
  16-bit unsigned short in place constant divided by signal, then clamp to saturated value.
- NppStatus nppsDivCRev_16u (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength)
  16-bit unsigned short signal divided by constant, then clamp to saturated value.
- NppStatus nppsDivCRev_32f_I (Npp32f nValue, Npp32f ∗pSrcDst, int nLength)
  32-bit floating point in place constant divided by signal.
- NppStatus nppsDivCRev_32f (const Npp32f ∗pSrc, Npp32f nValue, Npp32f ∗pDst, int nLength)
  32-bit floating point constant divided by signal.

7.22.1 Detailed Description

Divides a constant by each sample of a signal.

7.22.2 Function Documentation

7.22.2.1 NppStatus nppsDivCRev_16u (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal divided by constant, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.22.2.2 NppStatus nppsDivCRev_16u_I (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place constant divided by signal, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.22.2.3 NppStatus nppsDivCRev_32f (const Npp32f ∗ *pSrc*, Npp32f *nValue*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point constant divided by signal.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be divided by each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.22.2.4 NppStatus nppsDivCRev_32f_I (Npp32f *nValue*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place constant divided by signal.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be divided by each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23 Add

Sample by sample addition of two signals.

Functions

- NppStatus nppsAdd_16s (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength)

  16-bit signed short signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength)

  16-bit unsigned short signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, Npp32u ∗pDst, int nLength)

  32-bit unsigned int signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, Npp32f ∗pDst, int nLength)

  32-bit floating point signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, Npp64f ∗pDst, int nLength)

  64-bit floating point signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_8u16u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp16u ∗pDst, int nLength)

  8-bit unsigned char signal add signal with 16-bit unsigned result, then clamp to saturated value.

- NppStatus nppsAdd_16s32f (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp32f ∗pDst, int nLength)

  16-bit signed short signal add signal with 32-bit floating point result, then clamp to saturated value.

- NppStatus nppsAdd_8u_Sfs (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_16u_Sfs (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

16-bit signed short add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_64s_Sfs (const Npp64s ∗pSrc1, const Npp64s ∗pSrc2, Npp64s ∗pDst, int nLength, int nScaleFactor)

  64-bit signed integer add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_16sc_Sfs (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit signed complex short add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_32sc_Sfs (const Npp32sc ∗pSrc1, const Npp32sc ∗pSrc2, Npp32sc ∗pDst, int nLength, int nScaleFactor)

  32-bit signed complex integer add signal, scale, then clamp to saturated value.

- NppStatus nppsAdd_16s_I (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength)

  16-bit signed short in place signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_32f_I (const Npp32f ∗pSrc, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_64f_I (const Npp64f ∗pSrc, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point in place signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_32fc_I (const Npp32fc ∗pSrc, Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point in place signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_64fc_I (const Npp64fc ∗pSrc, Npp64fc ∗pSrcDst, int nLength)

  64-bit complex floating point in place signal add signal, then clamp to saturated value.

- NppStatus nppsAdd_16s32s_I (const Npp16s ∗pSrc, Npp32s ∗pSrcDst, int nLength)

  16/32-bit signed short in place signal add signal with 32-bit signed integer results, then clamp to saturated value.

- NppStatus nppsAdd_8u_ISfs (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal add signal, with scaling, then clamp to saturated value.

- NppStatus nppsAdd_16u_ISfs (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal add signal, with scaling, then clamp to saturated value.

- NppStatus nppsAdd_16s_ISfs (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal add signal, with scaling, then clamp to saturated value.

- NppStatus nppsAdd_32s_ISfs (const Npp32s ∗pSrc, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal add signal, with scaling, then clamp to saturated value.

- NppStatus nppsAdd_16sc_ISfs (const Npp16sc ∗pSrc, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit complex signed short in place signal add signal, with scaling, then clamp to saturated value.

- NppStatus nppsAdd_32sc_ISfs (const Npp32sc ∗pSrc, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit complex signed integer in place signal add signal, with scaling, then clamp to saturated value.

7.23.1 Detailed Description

Sample by sample addition of two signals.

7.23.2 Function Documentation

7.23.2.1 NppStatus nppsAdd_16s (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.2 NppStatus nppsAdd_16s32f (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp32f ∗ *pDst*, int *nLength*)

16-bit signed short signal add signal with 32-bit floating point result, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.3 NppStatus nppsAdd_16s32s_I (const Npp16s ∗ *pSrc*, Npp32s ∗ *pSrcDst*, int *nLength*)

16/32-bit signed short in place signal add signal with 32-bit signed integer results, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.4 NppStatus nppsAdd_16s_I (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short in place signal add signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.5 NppStatus nppsAdd_16s_ISfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.6 NppStatus nppsAdd_16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.7 NppStatus nppsAdd_16sc_ISfs (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.8 NppStatus nppsAdd_16sc_Sfs (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed complex short add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.9 NppStatus nppsAdd_16u (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.10 NppStatus nppsAdd_16u_ISfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.11 NppStatus nppsAdd_16u_Sfs (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.12 NppStatus nppsAdd_32f (const Npp32f * *pSrc1*, const Npp32f * *pSrc2*, Npp32f * *pDst*, int *nLength*)

32-bit floating point signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.13 NppStatus nppsAdd_32f_I (const Npp32f * *pSrc*, Npp32f * *pSrcDst*, int *nLength*)

32-bit floating point in place signal add signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.14 NppStatus nppsAdd_32fc (const Npp32fc * *pSrc1*, const Npp32fc * *pSrc2*, Npp32fc * *pDst*, int *nLength*)

32-bit complex floating point signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.15 NppStatus nppsAdd_32fc_I (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit complex floating point in place signal add signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.16 NppStatus nppsAdd_32s_ISfs (const Npp32s ∗ *pSrc*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.17 NppStatus nppsAdd_32s_Sfs (const Npp32s ∗ *pSrc1*, const Npp32s ∗ *pSrc2*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.18 NppStatus nppsAdd_32sc_ISfs (const Npp32sc ∗ *pSrc*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit complex signed integer in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.19 NppStatus nppsAdd_32sc_Sfs (const Npp32sc ∗ *pSrc1*, const Npp32sc ∗ *pSrc2*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed complex integer add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.20 NppStatus nppsAdd_32u (const Npp32u ∗ *pSrc1*, const Npp32u ∗ *pSrc2*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned int signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.21 NppStatus nppsAdd_64f (const Npp64f * *pSrc1*, const Npp64f * *pSrc2*, Npp64f * *pDst*, int *nLength*)

64-bit floating point signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.22 NppStatus nppsAdd_64f_I (const Npp64f * *pSrc*, Npp64f * *pSrcDst*, int *nLength*)

64-bit floating point in place signal add signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.23 NppStatus nppsAdd_64fc (const Npp64fc * *pSrc1*, const Npp64fc * *pSrc2*, Npp64fc * *pDst*, int *nLength*)

64-bit complex floating point signal add signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.24 NppStatus nppsAdd_64fc_I (const Npp64fc * *pSrc*, Npp64fc * *pSrcDst*, int *nLength*)

64-bit complex floating point in place signal add signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.25 NppStatus nppsAdd_64s_Sfs (const Npp64s * *pSrc1*, const Npp64s * *pSrc2*, Npp64s * *pDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.26 NppStatus nppsAdd_8u16u (const Npp8u * *pSrc1*, const Npp8u * *pSrc2*, Npp16u * *pDst*, int *nLength*)

8-bit unsigned char signal add signal with 16-bit unsigned result, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be added to signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.27 NppStatus nppsAdd_8u_ISfs (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal add signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be added to signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.23.2.28 NppStatus nppsAdd_8u_Sfs (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char add signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be added to signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.24 AddProduct

Adds sample by sample product of two signals to the destination signal.

Functions

- NppStatus nppsAddProduct_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, Npp32f ∗pDst, int nLength)

  32-bit floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

- NppStatus nppsAddProduct_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, Npp64f ∗pDst, int nLength)

  64-bit floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

- NppStatus nppsAddProduct_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

- NppStatus nppsAddProduct_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

- NppStatus nppsAddProduct_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal add product of source signal1 times source signal2 to destination signal, with scaling, then clamp to saturated value.

- NppStatus nppsAddProduct_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed short signal add product of source signal1 times source signal2 to destination signal, with scaling, then clamp to saturated value.

- NppStatus nppsAddProduct_16s32s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal add product of source signal1 times source signal2 to 32-bit signed integer destination signal, with scaling, then clamp to saturated value.

7.24.1 Detailed Description

Adds sample by sample product of two signals to the destination signal.

7.24.2 Function Documentation

7.24.2.1 NppStatus nppsAddProduct_16s32s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal add product of source signal1 times source signal2 to 32-bit signed integer destination signal, with scaling, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.24.2.2 NppStatus nppsAddProduct_16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal add product of source signal1 times source signal2 to destination signal, with scaling, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.24.2.3 NppStatus nppsAddProduct_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.24.2.4 NppStatus nppsAddProduct_32fc (const Npp32fc $*$ *pSrc1*, const Npp32fc $*$ *pSrc2*, Npp32fc $*$ *pDst*, int *nLength*)

32-bit complex floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.24.2.5 NppStatus nppsAddProduct_32s_Sfs (const Npp32s $*$ *pSrc1*, const Npp32s $*$ *pSrc2*, Npp32s $*$ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed short signal add product of source signal1 times source signal2 to destination signal, with scaling, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.24.2.6 NppStatus nppsAddProduct_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.24.2.7 NppStatus nppsAddProduct_64fc (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit complex floating point signal add product of source signal times destination signal to destination signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

pDst Destination Signal Pointer. product of source1 and source2 signal elements to be added to destination elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25 Mul

Sample by sample multiplication the samples of two signals.

Functions

- NppStatus nppsMul_16s (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength)

  16-bit signed short signal times signal, then clamp to saturated value.

- NppStatus nppsMul_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, Npp32f ∗pDst, int nLength)

  32-bit floating point signal times signal, then clamp to saturated value.

- NppStatus nppsMul_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, Npp64f ∗pDst, int nLength)

  64-bit floating point signal times signal, then clamp to saturated value.

- NppStatus nppsMul_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal times signal, then clamp to saturated value.

- NppStatus nppsMul_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal times signal, then clamp to saturated value.

- NppStatus nppsMul_8u16u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp16u ∗pDst, int nLength)

  8-bit unsigned char signal times signal with 16-bit unsigned result, then clamp to saturated value.

- NppStatus nppsMul_16s32f (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp32f ∗pDst, int nLength)

  16-bit signed short signal times signal with 32-bit floating point result, then clamp to saturated value.

- NppStatus nppsMul_32f32fc (const Npp32f ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

  32-bit floating point signal times 32-bit complex floating point signal with complex 32-bit floating point result, then clamp to saturated value.

- NppStatus nppsMul_8u_Sfs (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_16u_Sfs (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal time signal, scale, then clamp to saturated value.

- NppStatus nppsMul_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

32-bit signed integer signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_16sc_Sfs (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit signed complex short signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_32sc_Sfs (const Npp32sc ∗pSrc1, const Npp32sc ∗pSrc2, Npp32sc ∗pDst, int nLength, int nScaleFactor)

  32-bit signed complex integer signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_16u16s_Sfs (const Npp16u ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal times 16-bit signed short signal, scale, then clamp to 16-bit signed saturated value.

- NppStatus nppsMul_16s32s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal times signal, scale, then clamp to 32-bit signed saturated value.

- NppStatus nppsMul_32s32sc_Sfs (const Npp32s ∗pSrc1, const Npp32sc ∗pSrc2, Npp32sc ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal times 32-bit complex signed integer signal, scale, then clamp to 32-bit complex integer saturated value.

- NppStatus nppsMul_Low_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal times signal, scale, then clamp to saturated value.

- NppStatus nppsMul_16s_I (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength)

  16-bit signed short in place signal times signal, then clamp to saturated value.

- NppStatus nppsMul_32f_I (const Npp32f ∗pSrc, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal times signal, then clamp to saturated value.

- NppStatus nppsMul_64f_I (const Npp64f ∗pSrc, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point in place signal times signal, then clamp to saturated value.

- NppStatus nppsMul_32fc_I (const Npp32fc ∗pSrc, Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point in place signal times signal, then clamp to saturated value.

- NppStatus nppsMul_64fc_I (const Npp64fc ∗pSrc, Npp64fc ∗pSrcDst, int nLength)

  64-bit complex floating point in place signal times signal, then clamp to saturated value.

- NppStatus nppsMul_32f32fc_I (const Npp32f ∗pSrc, Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point in place signal times 32-bit floating point signal, then clamp to 32-bit complex floating point saturated value.

- NppStatus nppsMul_8u_ISfs (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_16u_ISfs (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_16s_ISfs (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_32s_ISfs (const Npp32s ∗pSrc, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_16sc_ISfs (const Npp16sc ∗pSrc, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit complex signed short in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_32sc_ISfs (const Npp32sc ∗pSrc, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit complex signed integer in place signal times signal, with scaling, then clamp to saturated value.

- NppStatus nppsMul_32s32sc_ISfs (const Npp32s ∗pSrc, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit complex signed integer in place signal times 32-bit signed integer signal, with scaling, then clamp to saturated value.

7.25.1 Detailed Description

Sample by sample multiplication the samples of two signals.

7.25.2 Function Documentation

7.25.2.1 NppStatus nppsMul_16s (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short signal times signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.2 NppStatus nppsMul_16s32f (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp32f ∗ *pDst*, int *nLength*)

16-bit signed short signal times signal with 32-bit floating point result, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.3 NppStatus nppsMul_16s32s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal times signal, scale, then clamp to 32-bit signed saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.4 NppStatus nppsMul_16s_I (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short in place signal times signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.5 NppStatus nppsMul_16s_ISfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.6 NppStatus nppsMul_16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.7 NppStatus nppsMul_16sc_ISfs (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.8 NppStatus nppsMul_16sc_Sfs (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed complex short signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.9 NppStatus nppsMul_16u16s_Sfs (const Npp16u ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal times 16-bit signed short signal, scale, then clamp to 16-bit signed saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.10 NppStatus nppsMul_16u_ISfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.11 NppStatus nppsMul_16u_Sfs (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal time signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.12 NppStatus nppsMul_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal times signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.13 NppStatus nppsMul_32f32fc (const Npp32f ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit floating point signal times 32-bit complex floating point signal with complex 32-bit floating point result, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.14 NppStatus nppsMul_32f32fc_I (const Npp32f ∗ *pSrc*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit complex floating point in place signal times 32-bit floating point signal, then clamp to 32-bit complex floating point saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.15 NppStatus nppsMul_32f_I (const Npp32f ∗ *pSrc*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal times signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.16 NppStatus nppsMul_32fc (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit complex floating point signal times signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.17 NppStatus nppsMul_32fc_I (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit complex floating point in place signal times signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.18 NppStatus nppsMul_32s32sc_ISfs (const Npp32s ∗ *pSrc*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit complex signed integer in place signal times 32-bit signed integer signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.19 NppStatus nppsMul_32s32sc_Sfs (const Npp32s ∗ *pSrc1*, const Npp32sc ∗ *pSrc2*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal times 32-bit complex signed integer signal, scale, then clamp to 32-bit complex integer saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.20 NppStatus nppsMul_32s_ISfs (const Npp32s ∗ *pSrc*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.21 NppStatus nppsMul_32s_Sfs (const Npp32s ∗ *pSrc1*, const Npp32s ∗ *pSrc2*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.22 NppStatus nppsMul_32sc_ISfs (const Npp32sc ∗ *pSrc*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit complex signed integer in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.23 NppStatus nppsMul_32sc_Sfs (const Npp32sc ∗ *pSrc1*, const Npp32sc ∗ *pSrc2*, Npp32sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed complex integer signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.24 NppStatus nppsMul_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal times signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.25 NppStatus nppsMul_64f_I (const Npp64f ∗ *pSrc*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point in place signal times signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.26 NppStatus nppsMul_64fc (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit complex floating point signal times signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.27 NppStatus nppsMul_64fc_I (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit complex floating point in place signal times signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.28 NppStatus nppsMul_8u16u (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal times signal with 16-bit unsigned result, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be multiplied by signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.29 NppStatus nppsMul_8u_ISfs (const Npp8u $*$ *pSrc*, Npp8u $*$ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal times signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be multiplied by signal1 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.30 NppStatus nppsMul_8u_Sfs (const Npp8u $*$ *pSrc1*, const Npp8u $*$ *pSrc2*, Npp8u $*$ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.25.2.31 NppStatus nppsMul_Low_32s_Sfs (const Npp32s $*$ *pSrc1*, const Npp32s $*$ *pSrc2*, Npp32s $*$ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal times signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal2 elements to be multiplied by signal1 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26 Sub

Sample by sample subtraction of the samples of two signals.

Functions

- NppStatus nppsSub_16s (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength)

  16-bit signed short signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, Npp32f ∗pDst, int nLength)

  32-bit floating point signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, Npp64f ∗pDst, int nLength)

  64-bit floating point signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal subtract signal, then clamp to saturated value.

- NppStatus nppsSub_16s32f (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp32f ∗pDst, int nLength)

  16-bit signed short signal subtract 16-bit signed short signal, then clamp and convert to 32-bit floating point saturated value.

- NppStatus nppsSub_8u_Sfs (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_16u_Sfs (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_16sc_Sfs (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit signed complex short signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_32sc_Sfs (const Npp32sc ∗pSrc1, const Npp32sc ∗pSrc2, Npp32sc ∗pDst, int nLength, int nScaleFactor)

32-bit signed complex integer signal subtract signal, scale, then clamp to saturated value.

- NppStatus nppsSub_16s_I (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength)
  16-bit signed short in place signal subtract signal, then clamp to saturated value.
- NppStatus nppsSub_32f_I (const Npp32f ∗pSrc, Npp32f ∗pSrcDst, int nLength)
  32-bit floating point in place signal subtract signal, then clamp to saturated value.
- NppStatus nppsSub_64f_I (const Npp64f ∗pSrc, Npp64f ∗pSrcDst, int nLength)
  64-bit floating point in place signal subtract signal, then clamp to saturated value.
- NppStatus nppsSub_32fc_I (const Npp32fc ∗pSrc, Npp32fc ∗pSrcDst, int nLength)
  32-bit complex floating point in place signal subtract signal, then clamp to saturated value.
- NppStatus nppsSub_64fc_I (const Npp64fc ∗pSrc, Npp64fc ∗pSrcDst, int nLength)
  64-bit complex floating point in place signal subtract signal, then clamp to saturated value.
- NppStatus nppsSub_8u_ISfs (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)
  8-bit unsigned char in place signal subtract signal, with scaling, then clamp to saturated value.
- NppStatus nppsSub_16u_ISfs (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)
  16-bit unsigned short in place signal subtract signal, with scaling, then clamp to saturated value.
- NppStatus nppsSub_16s_ISfs (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)
  16-bit signed short in place signal subtract signal, with scaling, then clamp to saturated value.
- NppStatus nppsSub_32s_ISfs (const Npp32s ∗pSrc, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)
  32-bit signed integer in place signal subtract signal, with scaling, then clamp to saturated value.
- NppStatus nppsSub_16sc_ISfs (const Npp16sc ∗pSrc, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)
  16-bit complex signed short in place signal subtract signal, with scaling, then clamp to saturated value.
- NppStatus nppsSub_32sc_ISfs (const Npp32sc ∗pSrc, Npp32sc ∗pSrcDst, int nLength, int nScaleFactor)
  32-bit complex signed integer in place signal subtract signal, with scaling, then clamp to saturated value.

7.26.1 Detailed Description

Sample by sample subtraction of the samples of two signals.

7.26.2 Function Documentation

7.26.2.1 NppStatus nppsSub_16s (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short signal subtract signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.2 NppStatus nppsSub_16s32f (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp32f ∗ *pDst*, int *nLength*)

16-bit signed short signal subtract 16-bit signed short signal, then clamp and convert to 32-bit floating point saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.3 NppStatus nppsSub_16s_I (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short in place signal subtract signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.4 NppStatus nppsSub_16s_ISfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.5 NppStatus nppsSub_16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.6 NppStatus nppsSub_16sc_ISfs (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.7 NppStatus nppsSub_16sc_Sfs (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed complex short signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.8 NppStatus nppsSub_16u_ISfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.9 NppStatus nppsSub_16u_Sfs (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.10 NppStatus nppsSub_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal subtract signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.11 NppStatus nppsSub_32f_I (const Npp32f ∗ *pSrc*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place signal subtract signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.12 NppStatus nppsSub_32fc (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit complex floating point signal subtract signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.13 NppStatus nppsSub_32fc_I (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit complex floating point in place signal subtract signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.14 NppStatus nppsSub_32s_ISfs (const Npp32s ∗ *pSrc*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.15 NppStatus nppsSub_32s_Sfs (const Npp32s ∗ *pSrc1*, const Npp32s ∗ *pSrc2*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.16 NppStatus nppsSub_32sc_ISfs (const Npp32sc ∗ *pSrc*, Npp32sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit complex signed integer in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.17 NppStatus nppsSub_32sc_Sfs (const Npp32sc * *pSrc1*, const Npp32sc * *pSrc2*, Npp32sc * *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed complex integer signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.18 NppStatus nppsSub_64f (const Npp64f * *pSrc1*, const Npp64f * *pSrc2*, Npp64f * *pDst*, int *nLength*)

64-bit floating point signal subtract signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.19 NppStatus nppsSub_64f_I (const Npp64f * *pSrc*, Npp64f * *pSrcDst*, int *nLength*)

64-bit floating point in place signal subtract signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.20 NppStatus nppsSub_64fc (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit complex floating point signal subtract signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal1 elements to be subtracted from signal2 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.21 NppStatus nppsSub_64fc_I (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit complex floating point in place signal subtract signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.22 NppStatus nppsSub_8u_ISfs (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal subtract signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 elements to be subtracted from signal2 elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.26.2.23 NppStatus nppsSub_8u_Sfs (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal subtract signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 elements to be subtracted from signal2 elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27 Div

Sample by sample division of the samples of two signals.

Functions

- NppStatus nppsDiv_8u_Sfs (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_16u_Sfs (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_16sc_Sfs (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit signed complex short signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_32s16s_Sfs (const Npp16s ∗pSrc1, const Npp32s ∗pSrc2, Npp16s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal divided by 16-bit signed short signal, scale, then clamp to 16-bit signed short saturated value.

- NppStatus nppsDiv_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, Npp32f ∗pDst, int nLength)

  32-bit floating point signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, Npp64f ∗pDst, int nLength)

  64-bit floating point signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_8u_ISfs (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char in place signal divide signal, with scaling, then clamp to saturated value.

- NppStatus nppsDiv_16u_ISfs (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

16-bit unsigned short in place signal divide signal, with scaling, then clamp to saturated value.

- NppStatus nppsDiv_16s_ISfs (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short in place signal divide signal, with scaling, then clamp to saturated value.

- NppStatus nppsDiv_16sc_ISfs (const Npp16sc ∗pSrc, Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit complex signed short in place signal divide signal, with scaling, then clamp to saturated value.

- NppStatus nppsDiv_32s_ISfs (const Npp32s ∗pSrc, Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer in place signal divide signal, with scaling, then clamp to saturated value.

- NppStatus nppsDiv_32f_I (const Npp32f ∗pSrc, Npp32f ∗pSrcDst, int nLength)

  32-bit floating point in place signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_64f_I (const Npp64f ∗pSrc, Npp64f ∗pSrcDst, int nLength)

  64-bit floating point in place signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_32fc_I (const Npp32fc ∗pSrc, Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point in place signal divide signal, then clamp to saturated value.

- NppStatus nppsDiv_64fc_I (const Npp64fc ∗pSrc, Npp64fc ∗pSrcDst, int nLength)

  64-bit complex floating point in place signal divide signal, then clamp to saturated value.

7.27.1 Detailed Description

Sample by sample division of the samples of two signals.

7.27.2 Function Documentation

7.27.2.1 NppStatus nppsDiv_16s_ISfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short in place signal divide signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.2 NppStatus nppsDiv_16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.3 NppStatus nppsDiv_16sc_ISfs (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short in place signal divide signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.4 NppStatus nppsDiv_16sc_Sfs (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed complex short signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.5 NppStatus nppsDiv_16u_ISfs (const Npp16u * *pSrc*, Npp16u * *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short in place signal divide signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.6 NppStatus nppsDiv_16u_Sfs (const Npp16u * *pSrc1*, const Npp16u * *pSrc2*, Npp16u * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.7 NppStatus nppsDiv_32f (const Npp32f * *pSrc1*, const Npp32f * *pSrc2*, Npp32f * *pDst*, int *nLength*)

32-bit floating point signal divide signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.8 NppStatus nppsDiv_32f_I (const Npp32f $*$ *pSrc*, Npp32f $*$ *pSrcDst*, int *nLength*)

32-bit floating point in place signal divide signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.9 NppStatus nppsDiv_32fc (const Npp32fc $*$ *pSrc1*, const Npp32fc $*$ *pSrc2*, Npp32fc $*$ *pDst*, int *nLength*)

32-bit complex floating point signal divide signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.10 NppStatus nppsDiv_32fc_I (const Npp32fc $*$ *pSrc*, Npp32fc $*$ *pSrcDst*, int *nLength*)

32-bit complex floating point in place signal divide signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.11 NppStatus nppsDiv_32s16s_Sfs (const Npp16s $*$ *pSrc1*, const Npp32s $*$ *pSrc2*, Npp16s $*$ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal divided by 16-bit signed short signal, scale, then clamp to 16-bit signed short saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.12 NppStatus nppsDiv_32s_ISfs (const Npp32s ∗ *pSrc*, Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer in place signal divide signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.13 NppStatus nppsDiv_32s_Sfs (const Npp32s ∗ *pSrc1*, const Npp32s ∗ *pSrc2*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.14 NppStatus nppsDiv_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal divide signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.15 NppStatus nppsDiv_64f_I (const Npp64f ∗ *pSrc*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point in place signal divide signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.16 NppStatus nppsDiv_64fc (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit complex floating point signal divide signal, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.17 NppStatus nppsDiv_64fc_I (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit complex floating point in place signal divide signal, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.18 NppStatus nppsDiv_8u_ISfs (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char in place signal divide signal, with scaling, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.27.2.19 NppStatus nppsDiv_8u_Sfs (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.28 Div_Round

Sample by sample division of the samples of two signals with rounding.

Functions

- NppStatus nppsDiv_Round_8u_Sfs (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  8-bit unsigned char signal divide signal, scale, then clamp to saturated value.

- NppStatus nppsDiv_Round_16u_Sfs (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  16-bit unsigned short signal divide signal, scale, round, then clamp to saturated value.

- NppStatus nppsDiv_Round_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, Npp16s ∗pDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  16-bit signed short signal divide signal, scale, round, then clamp to saturated value.

- NppStatus nppsDiv_Round_8u_ISfs (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  8-bit unsigned char in place signal divide signal, with scaling, rounding then clamp to saturated value.

- NppStatus nppsDiv_Round_16u_ISfs (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  16-bit unsigned short in place signal divide signal, with scaling, rounding then clamp to saturated value.

- NppStatus nppsDiv_Round_16s_ISfs (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength, NppRoundMode nRndMode, int nScaleFactor)

  16-bit signed short in place signal divide signal, with scaling, rounding then clamp to saturated value.

7.28.1 Detailed Description

Sample by sample division of the samples of two signals with rounding.

7.28.2 Function Documentation

7.28.2.1 NppStatus nppsDiv_Round_16s_ISfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

16-bit signed short in place signal divide signal, with scaling, rounding then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.28.2.2 NppStatus nppsDiv_Round_16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, Npp16s ∗ *pDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

16-bit signed short signal divide signal, scale, round, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.28.2.3 NppStatus nppsDiv_Round_16u_ISfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

16-bit unsigned short in place signal divide signal, with scaling, rounding then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.28.2.4 NppStatus nppsDiv_Round_16u_Sfs (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

16-bit unsigned short signal divide signal, scale, round, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.28.2.5 NppStatus nppsDiv_Round_8u_ISfs (const Npp8u $*$ *pSrc*, Npp8u $*$ *pSrcDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

8-bit unsigned char in place signal divide signal, with scaling, rounding then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal1 divisor elements to be divided into signal2 dividend elements

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.28.2.6 NppStatus nppsDiv_Round_8u_Sfs (const Npp8u $*$ *pSrc1*, const Npp8u $*$ *pSrc2*, Npp8u $*$ *pDst*, int *nLength*, NppRoundMode *nRndMode*, int *nScaleFactor*)

8-bit unsigned char signal divide signal, scale, then clamp to saturated value.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer, signal1 divisor elements to be divided into signal2 dividend elements.

pDst Destination Signal Pointer.

nLength Signal Length.

nRndMode various rounding modes.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.29 Abs

Absolute value of each sample of a signal.

Functions

- NppStatus nppsAbs_16s (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength)
  16-bit signed short signal absolute value.
- NppStatus nppsAbs_32s (const Npp32s ∗pSrc, Npp32s ∗pDst, int nLength)
  32-bit signed integer signal absolute value.
- NppStatus nppsAbs_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)
  32-bit floating point signal absolute value.
- NppStatus nppsAbs_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength)
  64-bit floating point signal absolute value.
- NppStatus nppsAbs_16s_I (Npp16s ∗pSrcDst, int nLength)
  16-bit signed short signal absolute value.
- NppStatus nppsAbs_32s_I (Npp32s ∗pSrcDst, int nLength)
  32-bit signed integer signal absolute value.
- NppStatus nppsAbs_32f_I (Npp32f ∗pSrcDst, int nLength)
  32-bit floating point signal absolute value.
- NppStatus nppsAbs_64f_I (Npp64f ∗pSrcDst, int nLength)
  64-bit floating point signal absolute value.

7.29.1 Detailed Description

Absolute value of each sample of a signal.

7.29.2 Function Documentation

7.29.2.1 NppStatus nppsAbs_16s (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short signal absolute value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.29.2.2 NppStatus nppsAbs_16s_I (Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short signal absolute value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.29.2.3 NppStatus nppsAbs_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal absolute value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.29.2.4 NppStatus nppsAbs_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point signal absolute value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.29.2.5 NppStatus nppsAbs_32s (const Npp32s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*)

32-bit signed integer signal absolute value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.29.2.6 NppStatus nppsAbs_32s_I (Npp32s $*$ *pSrcDst*, int *nLength*)

32-bit signed integer signal absolute value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.29.2.7 NppStatus nppsAbs_64f (const Npp64f $*$ *pSrc*, Npp64f $*$ *pDst*, int *nLength*)

64-bit floating point signal absolute value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.29.2.8 NppStatus nppsAbs_64f_I (Npp64f $*$ *pSrcDst*, int *nLength*)

64-bit floating point signal absolute value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30 Sqr

Squares each sample of a signal.

Functions

- NppStatus nppsSqr_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)

  32-bit floating point signal squared.

- NppStatus nppsSqr_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength)

  64-bit floating point signal squared.

- NppStatus nppsSqr_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal squared.

- NppStatus nppsSqr_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal squared.

- NppStatus nppsSqr_32f_I (Npp32f ∗pSrcDst, int nLength)

  32-bit floating point signal squared.

- NppStatus nppsSqr_64f_I (Npp64f ∗pSrcDst, int nLength)

  64-bit floating point signal squared.

- NppStatus nppsSqr_32fc_I (Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point signal squared.

- NppStatus nppsSqr_64fc_I (Npp64fc ∗pSrcDst, int nLength)

  64-bit complex floating point signal squared.

- NppStatus nppsSqr_8u_Sfs (const Npp8u ∗pSrc, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_16u_Sfs (const Npp16u ∗pSrc, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_16s_Sfs (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit complex signed short signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_8u_ISfs (Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_16u_ISfs (Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal squared, scale, then clamp to saturated value.

- NppStatus nppsSqr_16s_ISfs (Npp16s ∗pSrcDst, int nLength, int nScaleFactor)
  16-bit signed short signal squared, scale, then clamp to saturated value.
- NppStatus nppsSqr_16sc_ISfs (Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)
  16-bit complex signed short signal squared, scale, then clamp to saturated value.

7.30.1 Detailed Description

Squares each sample of a signal.

7.30.2 Function Documentation

7.30.2.1 NppStatus nppsSqr_16s_ISfs (Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal squared, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.2 NppStatus nppsSqr_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal squared, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.3 NppStatus nppsSqr_16sc_ISfs (Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short signal squared, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.4 NppStatus nppsSqr_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short signal squared, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.5 NppStatus nppsSqr_16u_ISfs (Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal squared, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.6 NppStatus nppsSqr_16u_Sfs (const Npp16u ∗ *pSrc*, Npp16u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal squared, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.7 NppStatus nppsSqr_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal squared.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.8 NppStatus nppsSqr_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point signal squared.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.9 NppStatus nppsSqr_32fc (const Npp32fc ∗ *pSrc*, Npp32fc ∗ *pDst*, int *nLength*)

32-bit complex floating point signal squared.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.10 NppStatus nppsSqr_32fc_I (Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit complex floating point signal squared.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.11 NppStatus nppsSqr_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal squared.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.12 NppStatus nppsSqr_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point signal squared.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.13 NppStatus nppsSqr_64fc (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit complex floating point signal squared.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.14 NppStatus nppsSqr_64fc_I (Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit complex floating point signal squared.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.15 NppStatus nppsSqr_8u_ISfs (Npp8u $*$ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal squared, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.30.2.16 NppStatus nppsSqr_8u_Sfs (const Npp8u $*$ *pSrc*, Npp8u $*$ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal squared, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31 Sqrt

Square root of each sample of a signal.

Functions

- NppStatus nppsSqrt_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)

  32-bit floating point signal square root.
- NppStatus nppsSqrt_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength)

  64-bit floating point signal square root.
- NppStatus nppsSqrt_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength)

  32-bit complex floating point signal square root.
- NppStatus nppsSqrt_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength)

  64-bit complex floating point signal square root.
- NppStatus nppsSqrt_32f_I (Npp32f ∗pSrcDst, int nLength)

  32-bit floating point signal square root.
- NppStatus nppsSqrt_64f_I (Npp64f ∗pSrcDst, int nLength)

  64-bit floating point signal square root.
- NppStatus nppsSqrt_32fc_I (Npp32fc ∗pSrcDst, int nLength)

  32-bit complex floating point signal square root.
- NppStatus nppsSqrt_64fc_I (Npp64fc ∗pSrcDst, int nLength)

  64-bit complex floating point signal square root.
- NppStatus nppsSqrt_8u_Sfs (const Npp8u ∗pSrc, Npp8u ∗pDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16u_Sfs (const Npp16u ∗pSrc, Npp16u ∗pDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16s_Sfs (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength, int nScaleFactor)

  16-bit complex signed short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_64s_Sfs (const Npp64s ∗pSrc, Npp64s ∗pDst, int nLength, int nScaleFactor)

  64-bit signed integer signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_32s16s_Sfs (const Npp32s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal square root, scale, then clamp to 16-bit signed integer saturated value.

- NppStatus nppsSqrt_64s16s_Sfs (const Npp64s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)

  64-bit signed integer signal square root, scale, then clamp to 16-bit signed integer saturated value.
- NppStatus nppsSqrt_8u_ISfs (Npp8u ∗pSrcDst, int nLength, int nScaleFactor)

  8-bit unsigned char signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16u_ISfs (Npp16u ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit unsigned short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16s_ISfs (Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_16sc_ISfs (Npp16sc ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit complex signed short signal square root, scale, then clamp to saturated value.
- NppStatus nppsSqrt_64s_ISfs (Npp64s ∗pSrcDst, int nLength, int nScaleFactor)

  64-bit signed integer signal square root, scale, then clamp to saturated value.

7.31.1 Detailed Description

Square root of each sample of a signal.

7.31.2 Function Documentation

7.31.2.1 NppStatus nppsSqrt_16s_ISfs (Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal square root, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.2 NppStatus nppsSqrt_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal square root, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.3 NppStatus nppsSqrt_16sc_ISfs (Npp16sc ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short signal square root, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.4 NppStatus nppsSqrt_16sc_Sfs (const Npp16sc ∗ *pSrc*, Npp16sc ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit complex signed short signal square root, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.5 NppStatus nppsSqrt_16u_ISfs (Npp16u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal square root, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.6 NppStatus nppsSqrt_16u_Sfs (const Npp16u * *pSrc*, Npp16u * *pDst*, int *nLength*, int *nScaleFactor*)

16-bit unsigned short signal square root, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.7 NppStatus nppsSqrt_32f (const Npp32f * *pSrc*, Npp32f * *pDst*, int *nLength*)

32-bit floating point signal square root.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.8 NppStatus nppsSqrt_32f_I (Npp32f * *pSrcDst*, int *nLength*)

32-bit floating point signal square root.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.9 NppStatus nppsSqrt_32fc (const Npp32fc * *pSrc*, Npp32fc * *pDst*, int *nLength*)

32-bit complex floating point signal square root.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.10 NppStatus nppsSqrt_32fc_I (Npp32fc ∗ *pSrcDst*, int *nLength*)

32-bit complex floating point signal square root.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.11 NppStatus nppsSqrt_32s16s_Sfs (const Npp32s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal square root, scale, then clamp to 16-bit signed integer saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.12 NppStatus nppsSqrt_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal square root.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.13 NppStatus nppsSqrt_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point signal square root.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.14 NppStatus nppsSqrt_64fc (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pDst*, int *nLength*)

64-bit complex floating point signal square root.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.15 NppStatus nppsSqrt_64fc_I (Npp64fc ∗ *pSrcDst*, int *nLength*)

64-bit complex floating point signal square root.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.16 NppStatus nppsSqrt_64s16s_Sfs (const Npp64s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer signal square root, scale, then clamp to 16-bit signed integer saturated value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.17 NppStatus nppsSqrt_64s_ISfs (Npp64s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer signal square root, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.18 NppStatus nppsSqrt_64s_Sfs (const Npp64s ∗ *pSrc*, Npp64s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer signal square root, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.19 NppStatus nppsSqrt_8u_ISfs (Npp8u ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal square root, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.31.2.20 NppStatus nppsSqrt_8u_Sfs (const Npp8u ∗ *pSrc*, Npp8u ∗ *pDst*, int *nLength*, int *nScaleFactor*)

8-bit unsigned char signal square root, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.32 Cubrt

Cube root of each sample of a signal.

Functions

- NppStatus nppsCubrt_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)
  32-bit floating point signal cube root.
- NppStatus nppsCubrt_32s16s_Sfs (const Npp32s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)
  32-bit signed integer signal cube root, scale, then clamp to 16-bit signed integer saturated value.

7.32.1 Detailed Description

Cube root of each sample of a signal.

7.32.2 Function Documentation

7.32.2.1 NppStatus nppsCubrt_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal cube root.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.32.2.2 NppStatus nppsCubrt_32s16s_Sfs (const Npp32s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal cube root, scale, then clamp to 16-bit signed integer saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.33 Exp

E raised to the power of each sample of a signal.

Functions

- NppStatus nppsExp_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)
  32-bit floating point signal exponent.
- NppStatus nppsExp_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength)
  64-bit floating point signal exponent.
- NppStatus nppsExp_32f64f (const Npp32f ∗pSrc, Npp64f ∗pDst, int nLength)
  32-bit floating point signal exponent with 64-bit floating point result.
- NppStatus nppsExp_32f_I (Npp32f ∗pSrcDst, int nLength)
  32-bit floating point signal exponent.
- NppStatus nppsExp_64f_I (Npp64f ∗pSrcDst, int nLength)
  64-bit floating point signal exponent.
- NppStatus nppsExp_16s_Sfs (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)
  16-bit signed short signal exponent, scale, then clamp to saturated value.
- NppStatus nppsExp_32s_Sfs (const Npp32s ∗pSrc, Npp32s ∗pDst, int nLength, int nScaleFactor)
  32-bit signed integer signal exponent, scale, then clamp to saturated value.
- NppStatus nppsExp_64s_Sfs (const Npp64s ∗pSrc, Npp64s ∗pDst, int nLength, int nScaleFactor)
  64-bit signed integer signal exponent, scale, then clamp to saturated value.
- NppStatus nppsExp_16s_ISfs (Npp16s ∗pSrcDst, int nLength, int nScaleFactor)
  16-bit signed short signal exponent, scale, then clamp to saturated value.
- NppStatus nppsExp_32s_ISfs (Npp32s ∗pSrcDst, int nLength, int nScaleFactor)
  32-bit signed integer signal exponent, scale, then clamp to saturated value.
- NppStatus nppsExp_64s_ISfs (Npp64s ∗pSrcDst, int nLength, int nScaleFactor)
  64-bit signed integer signal exponent, scale, then clamp to saturated value.

7.33.1 Detailed Description

E raised to the power of each sample of a signal.

7.33.2 Function Documentation

7.33.2.1 NppStatus nppsExp_16s_ISfs (Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal exponent, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.33.2.2 NppStatus nppsExp_16s_Sfs (const Npp16s $*$ *pSrc*, Npp16s $*$ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal exponent, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.
nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.33.2.3 NppStatus nppsExp_32f (const Npp32f $*$ *pSrc*, Npp32f $*$ *pDst*, int *nLength*)

32-bit floating point signal exponent.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.33.2.4 NppStatus nppsExp_32f64f (const Npp32f $*$ *pSrc*, Npp64f $*$ *pDst*, int *nLength*)

32-bit floating point signal exponent with 64-bit floating point result.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.33.2.5 NppStatus nppsExp_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point signal exponent.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.33.2.6 NppStatus nppsExp_32s_ISfs (Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal exponent, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.33.2.7 NppStatus nppsExp_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal exponent, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.33.2.8 NppStatus nppsExp_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*)

64-bit floating point signal exponent.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.33.2.9 NppStatus nppsExp_64f_I (Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point signal exponent.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.33.2.10 NppStatus nppsExp_64s_ISfs (Npp64s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer signal exponent, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.33.2.11 NppStatus nppsExp_64s_Sfs (const Npp64s ∗ *pSrc*, Npp64s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

64-bit signed integer signal exponent, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.34 Ln

Natural logarithm of each sample of a signal.

Functions

- NppStatus nppsLn_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)

  32-bit floating point signal natural logarithm.

- NppStatus nppsLn_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength)

  64-bit floating point signal natural logarithm.

- NppStatus nppsLn_64f32f (const Npp64f ∗pSrc, Npp32f ∗pDst, int nLength)

  64-bit floating point signal natural logarithm with 32-bit floating point result.

- NppStatus nppsLn_32f_I (Npp32f ∗pSrcDst, int nLength)

  32-bit floating point signal natural logarithm.

- NppStatus nppsLn_64f_I (Npp64f ∗pSrcDst, int nLength)

  64-bit floating point signal natural logarithm.

- NppStatus nppsLn_16s_Sfs (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)

  16-bit signed short signal natural logarithm, scale, then clamp to saturated value.

- NppStatus nppsLn_32s_Sfs (const Npp32s ∗pSrc, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal natural logarithm, scale, then clamp to saturated value.

- NppStatus nppsLn_32s16s_Sfs (const Npp32s ∗pSrc, Npp16s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal natural logarithm, scale, then clamp to 16-bit signed short saturated value.

- NppStatus nppsLn_16s_ISfs (Npp16s ∗pSrcDst, int nLength, int nScaleFactor)

  16-bit signed short signal natural logarithm, scale, then clamp to saturated value.

- NppStatus nppsLn_32s_ISfs (Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer signal natural logarithm, scale, then clamp to saturated value.

7.34.1 Detailed Description

Natural logarithm of each sample of a signal.

7.34.2 Function Documentation

7.34.2.1 NppStatus nppsLn_16s_ISfs (Npp16s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal natural logarithm, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.34.2.2 NppStatus nppsLn_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

16-bit signed short signal natural logarithm, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.34.2.3 NppStatus nppsLn_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal natural logarithm.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.34.2.4 NppStatus nppsLn_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point signal natural logarithm.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.34.2.5 NppStatus nppsLn_32s16s_Sfs (const Npp32s $*$ *pSrc*, Npp16s $*$ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal natural logarithm, scale, then clamp to 16-bit signed short saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.34.2.6 NppStatus nppsLn_32s_ISfs (Npp32s $*$ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal natural logarithm, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.34.2.7 NppStatus nppsLn_32s_Sfs (const Npp32s $*$ *pSrc*, Npp32s $*$ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal natural logarithm, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.34.2.8 NppStatus nppsLn_64f (const Npp64f $*$ *pSrc*, Npp64f $*$ *pDst*, int *nLength*)

64-bit floating point signal natural logarithm.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.34.2.9 NppStatus nppsLn_64f32f (const Npp64f $*$ *pSrc*, Npp32f $*$ *pDst*, int *nLength*)

64-bit floating point signal natural logarithm with 32-bit floating point result.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.34.2.10 NppStatus nppsLn_64f_I (Npp64f $*$ *pSrcDst*, int *nLength*)

64-bit floating point signal natural logarithm.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.35 10Log10

Ten times the decimal logarithm of each sample of a signal.

Functions

- NppStatus npps10Log10_32s_Sfs (const Npp32s ∗pSrc, Npp32s ∗pDst, int nLength, int nScaleFactor)

  32-bit signed integer signal 10 times base 10 logarithm, scale, then clamp to saturated value.

- NppStatus npps10Log10_32s_ISfs (Npp32s ∗pSrcDst, int nLength, int nScaleFactor)

  32-bit signed integer signal 10 times base 10 logarithm, scale, then clamp to saturated value.

7.35.1 Detailed Description

Ten times the decimal logarithm of each sample of a signal.

7.35.2 Function Documentation

7.35.2.1 NppStatus npps10Log10_32s_ISfs (Npp32s ∗ *pSrcDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal 10 times base 10 logarithm, scale, then clamp to saturated value.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.35.2.2 NppStatus npps10Log10_32s_Sfs (const Npp32s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*, int *nScaleFactor*)

32-bit signed integer signal 10 times base 10 logarithm, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.36 SumLn

Sums up the natural logarithm of each sample of a signal.

Functions

- NppStatus nppsSumLnGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for 32f SumLn.

- NppStatus nppsSumLn_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit floating point signal sum natural logarithm.

- NppStatus nppsSumLnGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for 64f SumLn.

- NppStatus nppsSumLn_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit floating point signal sum natural logarithm.

- NppStatus nppsSumLnGetBufferSize_32f64f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for 32f64f SumLn.

- NppStatus nppsSumLn_32f64f (const Npp32f ∗pSrc, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit flaoting point input, 64-bit floating point output signal sum natural logarithm.

- NppStatus nppsSumLnGetBufferSize_16s32f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for 16s32f SumLn.

- NppStatus nppsSumLn_16s32f (const Npp16s ∗pSrc, int nLength, Npp32f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer input, 32-bit floating point output signal sum natural logarithm.

7.36.1 Detailed Description

Sums up the natural logarithm of each sample of a signal.

7.36.2 Function Documentation

7.36.2.1 NppStatus nppsSumLn_16s32f (const Npp16s ∗ *pSrc*, int *nLength*, Npp32f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer input, 32-bit floating point output signal sum natural logarithm.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.36.2.2 NppStatus nppsSumLn_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit floating point signal sum natural logarithm.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.36.2.3 NppStatus nppsSumLn_32f64f (const Npp32f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit flaoting point input, 64-bit floating point output signal sum natural logarithm.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.36.2.4 NppStatus nppsSumLn_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit floating point signal sum natural logarithm.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.36.2.5 NppStatus nppsSumLnGetBufferSize_16s32f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for 16s32f SumLn.

This primitive provides the correct buffer size for nppsSumLn_16s32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.36.2.6 NppStatus nppsSumLnGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for 32f SumLn.

This primitive provides the correct buffer size for nppsSumLn_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.36.2.7 NppStatus nppsSumLnGetBufferSize_32f64f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for 32f64f SumLn.

This primitive provides the correct buffer size for nppsSumLn_32f64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.36.2.8 NppStatus nppsSumLnGetBufferSize_64f (int *nLength*, int $*$ *hpBufferSize*)

Device scratch buffer size (in bytes) for 64f SumLn.

This primitive provides the correct buffer size for nppsSumLn_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.37 Arctan

Inverse tangent of each sample of a signal.

Functions

- NppStatus nppsArctan_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength)
  32-bit floating point signal inverse tangent.
- NppStatus nppsArctan_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength)
  64-bit floating point signal inverse tangent.
- NppStatus nppsArctan_32f_I (Npp32f ∗pSrcDst, int nLength)
  32-bit floating point signal inverse tangent.
- NppStatus nppsArctan_64f_I (Npp64f ∗pSrcDst, int nLength)
  64-bit floating point signal inverse tangent.

7.37.1 Detailed Description

Inverse tangent of each sample of a signal.

7.37.2 Function Documentation

7.37.2.1 NppStatus nppsArctan_32f (const Npp32f ∗ *pSrc*, Npp32f ∗ *pDst*, int *nLength*)

32-bit floating point signal inverse tangent.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.37.2.2 NppStatus nppsArctan_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point signal inverse tangent.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.37.2.3 NppStatus nppsArctan_64f (const Npp64f * *pSrc*, Npp64f * *pDst*, int *nLength*)

64-bit floating point signal inverse tangent.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.37.2.4 NppStatus nppsArctan_64f_I (Npp64f * *pSrcDst*, int *nLength*)

64-bit floating point signal inverse tangent.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.38 Normalize

Normalize each sample of a real or complex signal using offset and division operations.

Functions

- NppStatus nppsNormalize_32f (const Npp32f ∗pSrc, Npp32f ∗pDst, int nLength, Npp32f vSub, Npp32f vDiv)

  32-bit floating point signal normalize.

- NppStatus nppsNormalize_32fc (const Npp32fc ∗pSrc, Npp32fc ∗pDst, int nLength, Npp32fc vSub, Npp32f vDiv)

  32-bit complex floating point signal normalize.

- NppStatus nppsNormalize_64f (const Npp64f ∗pSrc, Npp64f ∗pDst, int nLength, Npp64f vSub, Npp64f vDiv)

  64-bit floating point signal normalize.

- NppStatus nppsNormalize_64fc (const Npp64fc ∗pSrc, Npp64fc ∗pDst, int nLength, Npp64fc vSub, Npp64f vDiv)

  64-bit complex floating point signal normalize.

- NppStatus nppsNormalize_16s_Sfs (const Npp16s ∗pSrc, Npp16s ∗pDst, int nLength, Npp16s vSub, int vDiv, int nScaleFactor)

  16-bit signed short signal normalize, scale, then clamp to saturated value.

- NppStatus nppsNormalize_16sc_Sfs (const Npp16sc ∗pSrc, Npp16sc ∗pDst, int nLength, Npp16sc vSub, int vDiv, int nScaleFactor)

  16-bit complex signed short signal normalize, scale, then clamp to saturated value.

7.38.1 Detailed Description

Normalize each sample of a real or complex signal using offset and division operations.

7.38.2 Function Documentation

7.38.2.1 NppStatus nppsNormalize_16s_Sfs (const Npp16s ∗ *pSrc*, Npp16s ∗ *pDst*, int *nLength*, Npp16s *vSub*, int *vDiv*, int *nScaleFactor*)

16-bit signed short signal normalize, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

vSub value subtracted from each signal element before division

vDiv divisor of post-subtracted signal element dividend

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.38.2.2 NppStatus nppsNormalize_16sc_Sfs (const Npp16sc * *pSrc*, Npp16sc * *pDst*, int *nLength*, Npp16sc *vSub*, int *vDiv*, int *nScaleFactor*)

16-bit complex signed short signal normalize, scale, then clamp to saturated value.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

vSub value subtracted from each signal element before division

vDiv divisor of post-subtracted signal element dividend

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.38.2.3 NppStatus nppsNormalize_32f (const Npp32f * *pSrc*, Npp32f * *pDst*, int *nLength*, Npp32f *vSub*, Npp32f *vDiv*)

32-bit floating point signal normalize.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

vSub value subtracted from each signal element before division

vDiv divisor of post-subtracted signal element dividend

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.38.2.4 NppStatus nppsNormalize_32fc (const Npp32fc * *pSrc*, Npp32fc * *pDst*, int *nLength*, Npp32fc *vSub*, Npp32f *vDiv*)

32-bit complex floating point signal normalize.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

vSub value subtracted from each signal element before division

vDiv divisor of post-subtracted signal element dividend

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.38.2.5 NppStatus nppsNormalize_64f (const Npp64f ∗ *pSrc*, Npp64f ∗ *pDst*, int *nLength*, Npp64f *vSub*, Npp64f *vDiv*)

64-bit floating point signal normalize.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

vSub value subtracted from each signal element before division

vDiv divisor of post-subtracted signal element dividend

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.38.2.6 NppStatus nppsNormalize_64fc (const Npp64fc ∗ *pSrc*, Npp64fc ∗ *pDst*, int *nLength*, Npp64fc *vSub*, Npp64f *vDiv*)

64-bit complex floating point signal normalize.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

vSub value subtracted from each signal element before division

vDiv divisor of post-subtracted signal element dividend

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.39 Cauchy, CauchyD, and CauchyDD2

Determine Cauchy robust error function and its first and second derivatives for each sample of a signal.

Functions

- NppStatus nppsCauchy_32f_I (Npp32f ∗pSrcDst, int nLength, Npp32f nParam)
  32-bit floating point signal Cauchy error calculation.
- NppStatus nppsCauchyD_32f_I (Npp32f ∗pSrcDst, int nLength, Npp32f nParam)
  32-bit floating point signal Cauchy first derivative.
- NppStatus nppsCauchyDD2_32f_I (Npp32f ∗pSrcDst, Npp32f ∗pD2FVal, int nLength, Npp32f nParam)
  32-bit floating point signal Cauchy first and second derivatives.

7.39.1 Detailed Description

Determine Cauchy robust error function and its first and second derivatives for each sample of a signal.

7.39.2 Function Documentation

7.39.2.1 NppStatus nppsCauchy_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*, Npp32f *nParam*)

32-bit floating point signal Cauchy error calculation.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.
nParam constant used in Cauchy formula

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.39.2.2 NppStatus nppsCauchyD_32f_I (Npp32f ∗ *pSrcDst*, int *nLength*, Npp32f *nParam*)

32-bit floating point signal Cauchy first derivative.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.
nParam constant used in Cauchy formula

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.39.2.3 NppStatus nppsCauchyDD2_32f_I (Npp32f * *pSrcDst*, Npp32f * *pD2FVal*, int *nLength*, Npp32f *nParam*)

32-bit floating point signal Cauchy first and second derivatives.

Parameters:

pSrcDst In-Place Signal Pointer.

pD2FVal Source Signal Pointer. This signal contains the second derivative of the source signal.

nLength Signal Length.

nParam constant used in Cauchy formula

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.40 Logical And Shift Operations

Modules

- AndC

  Bitwise AND of a constant and each sample of a signal.

- And

  Sample by sample bitwise AND of samples from two signals.

- OrC

  Bitwise OR of a constant and each sample of a signal.

- Or

  Sample by sample bitwise OR of the samples from two signals.

- XorC

  Bitwise XOR of a constant and each sample of a signal.

- Xor

  Sample by sample bitwise XOR of the samples from two signals.

- Not

  Bitwise NOT of each sample of a signal.

- LShiftC

  Left shifts the bits of each sample of a signal by a constant amount.

- RShiftC

  Right shifts the bits of each sample of a signal by a constant amount.

7.41 AndC

Bitwise AND of a constant and each sample of a signal.

Functions

- NppStatus nppsAndC_8u (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength)

  8-bit unsigned char signal and with constant.

- NppStatus nppsAndC_16u (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength)

  16-bit unsigned short signal and with constant.

- NppStatus nppsAndC_32u (const Npp32u ∗pSrc, Npp32u nValue, Npp32u ∗pDst, int nLength)

  32-bit unsigned integer signal and with constant.

- NppStatus nppsAndC_8u_I (Npp8u nValue, Npp8u ∗pSrcDst, int nLength)

  8-bit unsigned char in place signal and with constant.

- NppStatus nppsAndC_16u_I (Npp16u nValue, Npp16u ∗pSrcDst, int nLength)

  16-bit unsigned short in place signal and with constant.

- NppStatus nppsAndC_32u_I (Npp32u nValue, Npp32u ∗pSrcDst, int nLength)

  32-bit unsigned signed integer in place signal and with constant.

7.41.1 Detailed Description

Bitwise AND of a constant and each sample of a signal.

7.41.2 Function Documentation

7.41.2.1 NppStatus nppsAndC_16u (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal and with constant.

Parameters:

pSrc	Source Signal Pointer.
nValue	Constant value to be anded with each vector element
pDst	Destination Signal Pointer.
nLength	Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.41.2.2 NppStatus nppsAndC_16u_I (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal and with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be anded with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.41.2.3 NppStatus nppsAndC_32u (const Npp32u ∗ *pSrc*, Npp32u *nValue*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal and with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be anded with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.41.2.4 NppStatus nppsAndC_32u_I (Npp32u *nValue*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place signal and with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be anded with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.41.2.5 NppStatus nppsAndC_8u (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal and with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be anded with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.41.2.6 NppStatus nppsAndC_8u_I (Npp8u *nValue*, Npp8u * *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal and with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be anded with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.42 And

Sample by sample bitwise AND of samples from two signals.

Functions

- NppStatus nppsAnd_8u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength)

  8-bit unsigned char signal and with signal.
- NppStatus nppsAnd_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength)

  16-bit unsigned short signal and with signal.
- NppStatus nppsAnd_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, Npp32u ∗pDst, int nLength)

  32-bit unsigned integer signal and with signal.
- NppStatus nppsAnd_8u_I (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength)

  8-bit unsigned char in place signal and with signal.
- NppStatus nppsAnd_16u_I (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength)

  16-bit unsigned short in place signal and with signal.
- NppStatus nppsAnd_32u_I (const Npp32u ∗pSrc, Npp32u ∗pSrcDst, int nLength)

  32-bit unsigned integer in place signal and with signal.

7.42.1 Detailed Description

Sample by sample bitwise AND of samples from two signals.

7.42.2 Function Documentation

7.42.2.1 NppStatus nppsAnd_16u (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal and with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be anded with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.42.2.2 NppStatus nppsAnd_16u_I (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal and with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be anded with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.42.2.3 NppStatus nppsAnd_32u (const Npp32u ∗ *pSrc1*, const Npp32u ∗ *pSrc2*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal and with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be anded with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.42.2.4 NppStatus nppsAnd_32u_I (const Npp32u ∗ *pSrc*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned integer in place signal and with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be anded with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.42.2.5 NppStatus nppsAnd_8u (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal and with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be anded with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.42.2.6 NppStatus nppsAnd_8u_I (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal and with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be anded with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.43 OrC

Bitwise OR of a constant and each sample of a signal.

Functions

- NppStatus nppsOrC_8u (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength)
  8-bit unsigned char signal or with constant.
- NppStatus nppsOrC_16u (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength)
  16-bit unsigned short signal or with constant.
- NppStatus nppsOrC_32u (const Npp32u ∗pSrc, Npp32u nValue, Npp32u ∗pDst, int nLength)
  32-bit unsigned integer signal or with constant.
- NppStatus nppsOrC_8u_I (Npp8u nValue, Npp8u ∗pSrcDst, int nLength)
  8-bit unsigned char in place signal or with constant.
- NppStatus nppsOrC_16u_I (Npp16u nValue, Npp16u ∗pSrcDst, int nLength)
  16-bit unsigned short in place signal or with constant.
- NppStatus nppsOrC_32u_I (Npp32u nValue, Npp32u ∗pSrcDst, int nLength)
  32-bit unsigned signed integer in place signal or with constant.

7.43.1 Detailed Description

Bitwise OR of a constant and each sample of a signal.

7.43.2 Function Documentation

7.43.2.1 NppStatus nppsOrC_16u (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal or with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be ored with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.43.2.2 NppStatus nppsOrC_16u_I (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be ored with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.43.2.3 NppStatus nppsOrC_32u (const Npp32u ∗ *pSrc*, Npp32u *nValue*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal or with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be ored with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.43.2.4 NppStatus nppsOrC_32u_I (Npp32u *nValue*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place signal or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be ored with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.43.2.5 NppStatus nppsOrC_8u (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal or with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be ored with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.43.2.6 NppStatus nppsOrC_8u_I (Npp8u *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be ored with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.44 Or

Sample by sample bitwise OR of the samples from two signals.

Functions

- NppStatus nppsOr_8u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength)
  8-bit unsigned char signal or with signal.
- NppStatus nppsOr_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength)
  16-bit unsigned short signal or with signal.
- NppStatus nppsOr_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, Npp32u ∗pDst, int nLength)
  32-bit unsigned integer signal or with signal.
- NppStatus nppsOr_8u_I (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength)
  8-bit unsigned char in place signal or with signal.
- NppStatus nppsOr_16u_I (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength)
  16-bit unsigned short in place signal or with signal.
- NppStatus nppsOr_32u_I (const Npp32u ∗pSrc, Npp32u ∗pSrcDst, int nLength)
  32-bit unsigned integer in place signal or with signal.

7.44.1 Detailed Description

Sample by sample bitwise OR of the samples from two signals.

7.44.2 Function Documentation

7.44.2.1 NppStatus nppsOr_16u (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal or with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be ored with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.44.2.2 NppStatus nppsOr_16u_I (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal or with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be ored with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.44.2.3 NppStatus nppsOr_32u (const Npp32u ∗ *pSrc1*, const Npp32u ∗ *pSrc2*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal or with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be ored with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.44.2.4 NppStatus nppsOr_32u_I (const Npp32u ∗ *pSrc*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned integer in place signal or with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be ored with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.44.2.5 NppStatus nppsOr_8u (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal or with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be ored with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.44.2.6 NppStatus nppsOr_8u_I (const Npp8u * *pSrc*, Npp8u * *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal or with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be ored with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.45 XorC

Bitwise XOR of a constant and each sample of a signal.

Functions

- NppStatus nppsXorC_8u (const Npp8u ∗pSrc, Npp8u nValue, Npp8u ∗pDst, int nLength)

    8-bit unsigned char signal exclusive or with constant.

- NppStatus nppsXorC_16u (const Npp16u ∗pSrc, Npp16u nValue, Npp16u ∗pDst, int nLength)

    16-bit unsigned short signal exclusive or with constant.

- NppStatus nppsXorC_32u (const Npp32u ∗pSrc, Npp32u nValue, Npp32u ∗pDst, int nLength)

    32-bit unsigned integer signal exclusive or with constant.

- NppStatus nppsXorC_8u_I (Npp8u nValue, Npp8u ∗pSrcDst, int nLength)

    8-bit unsigned char in place signal exclusive or with constant.

- NppStatus nppsXorC_16u_I (Npp16u nValue, Npp16u ∗pSrcDst, int nLength)

    16-bit unsigned short in place signal exclusive or with constant.

- NppStatus nppsXorC_32u_I (Npp32u nValue, Npp32u ∗pSrcDst, int nLength)

    32-bit unsigned signed integer in place signal exclusive or with constant.

7.45.1 Detailed Description

Bitwise XOR of a constant and each sample of a signal.

7.45.2 Function Documentation

7.45.2.1 NppStatus nppsXorC_16u (const Npp16u ∗ *pSrc*, Npp16u *nValue*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal exclusive or with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be exclusive ored with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.45.2.2 NppStatus nppsXorC_16u_I (Npp16u *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal exclusive or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be exclusive ored with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.45.2.3 NppStatus nppsXorC_32u (const Npp32u ∗ *pSrc*, Npp32u *nValue*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal exclusive or with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be exclusive ored with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.45.2.4 NppStatus nppsXorC_32u_I (Npp32u *nValue*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place signal exclusive or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be exclusive ored with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.45.2.5 NppStatus nppsXorC_8u (const Npp8u ∗ *pSrc*, Npp8u *nValue*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal exclusive or with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be exclusive ored with each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.45.2.6 NppStatus nppsXorC_8u_I (Npp8u *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal exclusive or with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be exclusive ored with each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.46 Xor

Sample by sample bitwise XOR of the samples from two signals.

Functions

- NppStatus nppsXor_8u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, Npp8u ∗pDst, int nLength)

  8-bit unsigned char signal exclusive or with signal.

- NppStatus nppsXor_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, Npp16u ∗pDst, int nLength)

  16-bit unsigned short signal exclusive or with signal.

- NppStatus nppsXor_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, Npp32u ∗pDst, int nLength)

  32-bit unsigned integer signal exclusive or with signal.

- NppStatus nppsXor_8u_I (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength)

  8-bit unsigned char in place signal exclusive or with signal.

- NppStatus nppsXor_16u_I (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength)

  16-bit unsigned short in place signal exclusive or with signal.

- NppStatus nppsXor_32u_I (const Npp32u ∗pSrc, Npp32u ∗pSrcDst, int nLength)

  32-bit unsigned integer in place signal exclusive or with signal.

7.46.1 Detailed Description

Sample by sample bitwise XOR of the samples from two signals.

7.46.2 Function Documentation

7.46.2.1 NppStatus nppsXor_16u (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal exclusive or with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be exclusive ored with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.46.2.2 NppStatus nppsXor_16u_I (const Npp16u * *pSrc*, Npp16u * *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal exclusive or with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be exclusive ored with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.46.2.3 NppStatus nppsXor_32u (const Npp32u * *pSrc1*, const Npp32u * *pSrc2*, Npp32u * *pDst*, int *nLength*)

32-bit unsigned integer signal exclusive or with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be exclusive ored with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.46.2.4 NppStatus nppsXor_32u_I (const Npp32u * *pSrc*, Npp32u * *pSrcDst*, int *nLength*)

32-bit unsigned integer in place signal exclusive or with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be exclusive ored with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.46.2.5 NppStatus nppsXor_8u (const Npp8u * *pSrc1*, const Npp8u * *pSrc2*, Npp8u * *pDst*, int *nLength*)

8-bit unsigned char signal exclusive or with signal.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer. signal2 elements to be exclusive ored with signal1 elements

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.46.2.6 NppStatus nppsXor_8u_I (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal exclusive or with signal.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer. signal2 elements to be exclusive ored with signal1 elements

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.47 Not

Bitwise NOT of each sample of a signal.

Functions

- NppStatus nppsNot_8u (const Npp8u ∗pSrc, Npp8u ∗pDst, int nLength)

  8-bit unsigned char not signal.

- NppStatus nppsNot_16u (const Npp16u ∗pSrc, Npp16u ∗pDst, int nLength)

  16-bit unsigned short not signal.

- NppStatus nppsNot_32u (const Npp32u ∗pSrc, Npp32u ∗pDst, int nLength)

  32-bit unsigned integer not signal.

- NppStatus nppsNot_8u_I (Npp8u ∗pSrcDst, int nLength)

  8-bit unsigned char in place not signal.

- NppStatus nppsNot_16u_I (Npp16u ∗pSrcDst, int nLength)

  16-bit unsigned short in place not signal.

- NppStatus nppsNot_32u_I (Npp32u ∗pSrcDst, int nLength)

  32-bit unsigned signed integer in place not signal.

7.47.1 Detailed Description

Bitwise NOT of each sample of a signal.

7.47.2 Function Documentation

7.47.2.1 NppStatus nppsNot_16u (const Npp16u ∗ *pSrc*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short not signal.

Parameters:

pSrc Source Signal Pointer.

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.47.2.2 NppStatus nppsNot_16u_I (Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place not signal.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.47.2.3 NppStatus nppsNot_32u (const Npp32u ∗ *pSrc*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer not signal.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.47.2.4 NppStatus nppsNot_32u_I (Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place not signal.

Parameters:

pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.47.2.5 NppStatus nppsNot_8u (const Npp8u ∗ *pSrc*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char not signal.

Parameters:

pSrc Source Signal Pointer.
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.47.2.6 NppStatus nppsNot_8u_I (Npp8u $*$ *pSrcDst*, int *nLength*)

8-bit unsigned char in place not signal.

Parameters:

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.48 LShiftC

Left shifts the bits of each sample of a signal by a constant amount.

Functions

- NppStatus nppsLShiftC_8u (const Npp8u ∗pSrc, int nValue, Npp8u ∗pDst, int nLength)

  8-bit unsigned char signal left shift with constant.
- NppStatus nppsLShiftC_16u (const Npp16u ∗pSrc, int nValue, Npp16u ∗pDst, int nLength)

  16-bit unsigned short signal left shift with constant.
- NppStatus nppsLShiftC_16s (const Npp16s ∗pSrc, int nValue, Npp16s ∗pDst, int nLength)

  16-bit signed short signal left shift with constant.
- NppStatus nppsLShiftC_32u (const Npp32u ∗pSrc, int nValue, Npp32u ∗pDst, int nLength)

  32-bit unsigned integer signal left shift with constant.
- NppStatus nppsLShiftC_32s (const Npp32s ∗pSrc, int nValue, Npp32s ∗pDst, int nLength)

  32-bit signed integer signal left shift with constant.
- NppStatus nppsLShiftC_8u_I (int nValue, Npp8u ∗pSrcDst, int nLength)

  8-bit unsigned char in place signal left shift with constant.
- NppStatus nppsLShiftC_16u_I (int nValue, Npp16u ∗pSrcDst, int nLength)

  16-bit unsigned short in place signal left shift with constant.
- NppStatus nppsLShiftC_16s_I (int nValue, Npp16s ∗pSrcDst, int nLength)

  16-bit signed short in place signal left shift with constant.
- NppStatus nppsLShiftC_32u_I (int nValue, Npp32u ∗pSrcDst, int nLength)

  32-bit unsigned signed integer in place signal left shift with constant.
- NppStatus nppsLShiftC_32s_I (int nValue, Npp32s ∗pSrcDst, int nLength)

  32-bit signed signed integer in place signal left shift with constant.

7.48.1 Detailed Description

Left shifts the bits of each sample of a signal by a constant amount.

7.48.2 Function Documentation

7.48.2.1 NppStatus nppsLShiftC_16s (const Npp16s ∗ *pSrc*, int *nValue*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short signal left shift with constant.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be used to left shift each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.48.2.2 NppStatus nppsLShiftC_16s_I (int *nValue*, Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short in place signal left shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be used to left shift each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.48.2.3 NppStatus nppsLShiftC_16u (const Npp16u ∗ *pSrc*, int *nValue*, Npp16u ∗ *pDst*, int *nLength*)

16-bit unsigned short signal left shift with constant.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be used to left shift each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.48.2.4 NppStatus nppsLShiftC_16u_I (int *nValue*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal left shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be used to left shift each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.48.2.5 NppStatus nppsLShiftC_32s (const Npp32s ∗ *pSrc*, int *nValue*, Npp32s ∗ *pDst*, int *nLength*)

32-bit signed integer signal left shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to left shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.48.2.6 NppStatus nppsLShiftC_32s_I (int *nValue*, Npp32s ∗ *pSrcDst*, int *nLength*)

32-bit signed signed integer in place signal left shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to left shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.48.2.7 NppStatus nppsLShiftC_32u (const Npp32u ∗ *pSrc*, int *nValue*, Npp32u ∗ *pDst*, int *nLength*)

32-bit unsigned integer signal left shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to left shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.48.2.8 NppStatus nppsLShiftC_32u_I (int *nValue*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place signal left shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to left shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.48.2.9 NppStatus nppsLShiftC_8u (const Npp8u ∗ *pSrc*, int *nValue*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal left shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to left shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.48.2.10 NppStatus nppsLShiftC_8u_I (int *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal left shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to left shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.49 RShiftC

Right shifts the bits of each sample of a signal by a constant amount.

Functions

- NppStatus nppsRShiftC_8u (const Npp8u ∗pSrc, int nValue, Npp8u ∗pDst, int nLength)

    8-bit unsigned char signal right shift with constant.
- NppStatus nppsRShiftC_16u (const Npp16u ∗pSrc, int nValue, Npp16u ∗pDst, int nLength)

    16-bit unsigned short signal right shift with constant.
- NppStatus nppsRShiftC_16s (const Npp16s ∗pSrc, int nValue, Npp16s ∗pDst, int nLength)

    16-bit signed short signal right shift with constant.
- NppStatus nppsRShiftC_32u (const Npp32u ∗pSrc, int nValue, Npp32u ∗pDst, int nLength)

    32-bit unsigned integer signal right shift with constant.
- NppStatus nppsRShiftC_32s (const Npp32s ∗pSrc, int nValue, Npp32s ∗pDst, int nLength)

    32-bit signed integer signal right shift with constant.
- NppStatus nppsRShiftC_8u_I (int nValue, Npp8u ∗pSrcDst, int nLength)

    8-bit unsigned char in place signal right shift with constant.
- NppStatus nppsRShiftC_16u_I (int nValue, Npp16u ∗pSrcDst, int nLength)

    16-bit unsigned short in place signal right shift with constant.
- NppStatus nppsRShiftC_16s_I (int nValue, Npp16s ∗pSrcDst, int nLength)

    16-bit signed short in place signal right shift with constant.
- NppStatus nppsRShiftC_32u_I (int nValue, Npp32u ∗pSrcDst, int nLength)

    32-bit unsigned signed integer in place signal right shift with constant.
- NppStatus nppsRShiftC_32s_I (int nValue, Npp32s ∗pSrcDst, int nLength)

    32-bit signed signed integer in place signal right shift with constant.

7.49.1 Detailed Description

Right shifts the bits of each sample of a signal by a constant amount.

7.49.2 Function Documentation

7.49.2.1 NppStatus nppsRShiftC_16s (const Npp16s ∗ *pSrc*, int *nValue*, Npp16s ∗ *pDst*, int *nLength*)

16-bit signed short signal right shift with constant.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be used to right shift each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.49.2.2 NppStatus nppsRShiftC_16s_I (int *nValue*, Npp16s * *pSrcDst*, int *nLength*)

16-bit signed short in place signal right shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be used to right shift each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.49.2.3 NppStatus nppsRShiftC_16u (const Npp16u * *pSrc*, int *nValue*, Npp16u * *pDst*, int *nLength*)

16-bit unsigned short signal right shift with constant.

Parameters:

pSrc Source Signal Pointer.
nValue Constant value to be used to right shift each vector element
pDst Destination Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.49.2.4 NppStatus nppsRShiftC_16u_I (int *nValue*, Npp16u * *pSrcDst*, int *nLength*)

16-bit unsigned short in place signal right shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.
nValue Constant value to be used to right shift each vector element
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.49.2.5 NppStatus nppsRShiftC_32s (const Npp32s $*$ *pSrc*, int *nValue*, Npp32s $*$ *pDst*, int *nLength*)

32-bit signed integer signal right shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to right shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.49.2.6 NppStatus nppsRShiftC_32s_I (int *nValue*, Npp32s $*$ *pSrcDst*, int *nLength*)

32-bit signed signed integer in place signal right shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to right shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.49.2.7 NppStatus nppsRShiftC_32u (const Npp32u $*$ *pSrc*, int *nValue*, Npp32u $*$ *pDst*, int *nLength*)

32-bit unsigned integer signal right shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to right shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.49.2.8 NppStatus nppsRShiftC_32u_I (int *nValue*, Npp32u ∗ *pSrcDst*, int *nLength*)

32-bit unsigned signed integer in place signal right shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to right shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.49.2.9 NppStatus nppsRShiftC_8u (const Npp8u ∗ *pSrc*, int *nValue*, Npp8u ∗ *pDst*, int *nLength*)

8-bit unsigned char signal right shift with constant.

Parameters:

pSrc Source Signal Pointer.

nValue Constant value to be used to right shift each vector element

pDst Destination Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.49.2.10 NppStatus nppsRShiftC_8u_I (int *nValue*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit unsigned char in place signal right shift with constant.

Parameters:

pSrcDst In-Place Signal Pointer.

nValue Constant value to be used to right shift each vector element

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.50 Statistical Functions

Functions that provide global signal statistics like: sum, mean, standard deviation, min, max, etc.

Modules

- MinEvery And MaxEvery Functions

  Performs the min or max operation on the samples of a signal.

- Sum

  signal_min_every_or_max_every

- Maximum
- Minimum
- Mean
- Standard Deviation
- Mean And Standard Deviation
- Minimum_Maximum
- Infinity Norm
- L1 Norm
- L2 Norm
- Infinity Norm Diff
- L1 Norm Diff
- L2 Norm Diff
- Dot Product
- Count In Range
- Count Zero Crossings
- MaximumError

  Primitives for computing the maximum error between two signals.

- AverageError

  Primitives for computing the Average error between two signals.

- MaximumRelativeError

  Primitives for computing the MaximumRelative error between two signals.

- AverageRelativeError

  Primitives for computing the AverageRelative error between two signals.

7.50.1 Detailed Description

Functions that provide global signal statistics like: sum, mean, standard deviation, min, max, etc.

7.51 MinEvery And MaxEvery Functions

Performs the min or max operation on the samples of a signal.

Functions

- NppStatus nppsMinEvery_8u_I (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength)
  8-bit in place min value for each pair of elements.
- NppStatus nppsMinEvery_16u_I (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength)
  16-bit unsigned short integer in place min value for each pair of elements.
- NppStatus nppsMinEvery_16s_I (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength)
  16-bit signed short integer in place min value for each pair of elements.
- NppStatus nppsMinEvery_32s_I (const Npp32s ∗pSrc, Npp32s ∗pSrcDst, int nLength)
  32-bit signed integer in place min value for each pair of elements.
- NppStatus nppsMinEvery_32f_I (const Npp32f ∗pSrc, Npp32f ∗pSrcDst, int nLength)
  32-bit floating point in place min value for each pair of elements.
- NppStatus nppsMinEvery_64f_I (const Npp64f ∗pSrc, Npp64f ∗pSrcDst, int nLength)
  64-bit floating point in place min value for each pair of elements.
- NppStatus nppsMaxEvery_8u_I (const Npp8u ∗pSrc, Npp8u ∗pSrcDst, int nLength)
  8-bit in place max value for each pair of elements.
- NppStatus nppsMaxEvery_16u_I (const Npp16u ∗pSrc, Npp16u ∗pSrcDst, int nLength)
  16-bit unsigned short integer in place max value for each pair of elements.
- NppStatus nppsMaxEvery_16s_I (const Npp16s ∗pSrc, Npp16s ∗pSrcDst, int nLength)
  16-bit signed short integer in place max value for each pair of elements.
- NppStatus nppsMaxEvery_32s_I (const Npp32s ∗pSrc, Npp32s ∗pSrcDst, int nLength)
  32-bit signed integer in place max value for each pair of elements.
- NppStatus nppsMaxEvery_32f_I (const Npp32f ∗pSrc, Npp32f ∗pSrcDst, int nLength)
  32-bit floating point in place max value for each pair of elements.

7.51.1 Detailed Description

Performs the min or max operation on the samples of a signal.

7.51.2 Function Documentation

7.51.2.1 NppStatus nppsMaxEvery_16s_I (const Npp16s ∗ *pSrc*, Npp16s ∗ *pSrcDst*, int *nLength*)

16-bit signed short integer in place max value for each pair of elements.

Parameters:

pSrc Source Signal Pointer.
pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.51.2.2 NppStatus nppsMaxEvery_16u_I (const Npp16u ∗ *pSrc*, Npp16u ∗ *pSrcDst*, int *nLength*)

16-bit unsigned short integer in place max value for each pair of elements.

Parameters:

pSrc Source Signal Pointer.
pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.51.2.3 NppStatus nppsMaxEvery_32f_I (const Npp32f ∗ *pSrc*, Npp32f ∗ *pSrcDst*, int *nLength*)

32-bit floating point in place max value for each pair of elements.

Parameters:

pSrc Source Signal Pointer.
pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.51.2.4 NppStatus nppsMaxEvery_32s_I (const Npp32s ∗ *pSrc*, Npp32s ∗ *pSrcDst*, int *nLength*)

32-bit signed integer in place max value for each pair of elements.

Parameters:

pSrc Source Signal Pointer.
pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.51.2.5 NppStatus nppsMaxEvery_8u_I (const Npp8u * *pSrc*, Npp8u * *pSrcDst*, int *nLength*)

8-bit in place max value for each pair of elements.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.51.2.6 NppStatus nppsMinEvery_16s_I (const Npp16s * *pSrc*, Npp16s * *pSrcDst*, int *nLength*)

16-bit signed short integer in place min value for each pair of elements.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.51.2.7 NppStatus nppsMinEvery_16u_I (const Npp16u * *pSrc*, Npp16u * *pSrcDst*, int *nLength*)

16-bit unsigned short integer in place min value for each pair of elements.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.51.2.8 NppStatus nppsMinEvery_32f_I (const Npp32f * *pSrc*, Npp32f * *pSrcDst*, int *nLength*)

32-bit floating point in place min value for each pair of elements.

Parameters:

pSrc Source Signal Pointer.

pSrcDst In-Place Signal Pointer.

nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.51.2.9 NppStatus nppsMinEvery_32s_I (const Npp32s ∗ *pSrc*, Npp32s ∗ *pSrcDst*, int *nLength*)

32-bit signed integer in place min value for each pair of elements.

Parameters:

pSrc Source Signal Pointer.
pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.51.2.10 NppStatus nppsMinEvery_64f_I (const Npp64f ∗ *pSrc*, Npp64f ∗ *pSrcDst*, int *nLength*)

64-bit floating point in place min value for each pair of elements.

Parameters:

pSrc Source Signal Pointer.
pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.51.2.11 NppStatus nppsMinEvery_8u_I (const Npp8u ∗ *pSrc*, Npp8u ∗ *pSrcDst*, int *nLength*)

8-bit in place min value for each pair of elements.

Parameters:

pSrc Source Signal Pointer.
pSrcDst In-Place Signal Pointer.
nLength Signal Length.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.52 Sum

signal_min_every_or_max_every

Functions

- NppStatus nppsSumGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsSum_32f.

- NppStatus nppsSumGetBufferSize_32fc (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsSum_32fc.

- NppStatus nppsSumGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsSum_64f.

- NppStatus nppsSumGetBufferSize_64fc (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsSum_64fc.

- NppStatus nppsSumGetBufferSize_16s_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsSum_16s_Sfs.

- NppStatus nppsSumGetBufferSize_16sc_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsSum_16sc_Sfs.

- NppStatus nppsSumGetBufferSize_16sc32sc_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsSum_16sc32sc_Sfs.

- NppStatus nppsSumGetBufferSize_32s_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsSum_32s_Sfs.

- NppStatus nppsSumGetBufferSize_16s32s_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsSum_16s32s_Sfs.

- NppStatus nppsSum_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pSum, Npp8u ∗pDeviceBuffer)

  32-bit float vector sum method

- NppStatus nppsSum_32fc (const Npp32fc ∗pSrc, int nLength, Npp32fc ∗pSum, Npp8u ∗pDeviceBuffer)

  32-bit float complex vector sum method

- NppStatus nppsSum_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pSum, Npp8u ∗pDeviceBuffer)

  64-bit double vector sum method

- NppStatus nppsSum_64fc (const Npp64fc ∗pSrc, int nLength, Npp64fc ∗pSum, Npp8u ∗pDeviceBuffer)

  64-bit double complex vector sum method

- NppStatus nppsSum_16s_Sfs (const Npp16s ∗pSrc, int nLength, Npp16s ∗pSum, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit short vector sum with integer scaling method

- NppStatus nppsSum_32s_Sfs (const Npp32s ∗pSrc, int nLength, Npp32s ∗pSum, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  32-bit integer vector sum with integer scaling method

- NppStatus nppsSum_16sc_Sfs (const Npp16sc ∗pSrc, int nLength, Npp16sc ∗pSum, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit short complex vector sum with integer scaling method

- NppStatus nppsSum_16sc32sc_Sfs (const Npp16sc ∗pSrc, int nLength, Npp32sc ∗pSum, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit short complex vector sum (32bit int complex) with integer scaling method

- NppStatus nppsSum_16s32s_Sfs (const Npp16s ∗pSrc, int nLength, Npp32s ∗pSum, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit integer vector sum (32bit) with integer scaling method

7.52.1 Detailed Description

signal_min_every_or_max_every

7.52.2 Function Documentation

7.52.2.1 NppStatus nppsSum_16s32s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pSum*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit integer vector sum (32bit) with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsSumGetBufferSize_16s32s_Sfs to determine the minium number of bytes required.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.52.2.2 NppStatus nppsSum_16s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pSum*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit short vector sum with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsSumGetBufferSize_16s_Sfs to determine the minium number of bytes required.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.52.2.3 NppStatus nppsSum_16sc32sc_Sfs (const Npp16sc ∗ *pSrc*, int *nLength*, Npp32sc ∗ *pSum*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit short complex vector sum (32bit int complex) with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsSumGetBufferSize_16sc32sc_Sfs to determine the minium number of bytes required.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.52.2.4 NppStatus nppsSum_16sc_Sfs (const Npp16sc ∗ *pSrc*, int *nLength*, Npp16sc ∗ *pSum*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit short complex vector sum with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsSumGetBufferSize_16sc_Sfs to determine the minium number of bytes required.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.52.2.5 NppStatus nppsSum_32f (const Npp32f * *pSrc*, int *nLength*, Npp32f * *pSum*, Npp8u * *pDeviceBuffer*)

32-bit float vector sum method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsSumGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.52.2.6 NppStatus nppsSum_32fc (const Npp32fc * *pSrc*, int *nLength*, Npp32fc * *pSum*, Npp8u * *pDeviceBuffer*)

32-bit float complex vector sum method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsSumGetBufferSize_32fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.52.2.7 NppStatus nppsSum_32s_Sfs (const Npp32s * *pSrc*, int *nLength*, Npp32s * *pSum*, int *nScaleFactor*, Npp8u * *pDeviceBuffer*)

32-bit integer vector sum with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsSumGetBufferSize_32s_Sfs to determine the minium number of bytes required.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.52.2.8 NppStatus nppsSum_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pSum*, Npp8u ∗ *pDeviceBuffer*)

64-bit double vector sum method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsSumGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.52.2.9 NppStatus nppsSum_64fc (const Npp64fc ∗ *pSrc*, int *nLength*, Npp64fc ∗ *pSum*, Npp8u ∗ *pDeviceBuffer*)

64-bit double complex vector sum method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pSum Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsSumGetBufferSize_64fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.52.2.10 NppStatus nppsSumGetBufferSize_16s32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsSum_16s32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.52.2.11 NppStatus nppsSumGetBufferSize_16s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsSum_16s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.52.2.12 NppStatus nppsSumGetBufferSize_16sc32sc_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsSum_16sc32sc_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.52.2.13 NppStatus nppsSumGetBufferSize_16sc_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsSum_16sc_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.52.2.14 NppStatus nppsSumGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsSum_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.52.2.15 NppStatus nppsSumGetBufferSize_32fc (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsSum_32fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.52.2.16 NppStatus nppsSumGetBufferSize_32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsSum_32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.52.2.17 NppStatus nppsSumGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsSum_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.52.2.18 NppStatus nppsSumGetBufferSize_64fc (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsSum_64fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53 Maximum

Functions

- NppStatus nppsMaxGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMax_16s.

- NppStatus nppsMaxGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMax_32s.

- NppStatus nppsMaxGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMax_32f.

- NppStatus nppsMaxGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMax_64f.

- NppStatus nppsMax_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMax, Npp8u ∗pDeviceBuffer)

  16-bit integer vector max method

- NppStatus nppsMax_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMax, Npp8u ∗pDeviceBuffer)

  32-bit integer vector max method

- NppStatus nppsMax_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pMax, Npp8u ∗pDeviceBuffer)

  32-bit float vector max method

- NppStatus nppsMax_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pMax, Npp8u ∗pDeviceBuffer)

  64-bit float vector max method

- NppStatus nppsMaxIndxGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMaxIndx_16s.

- NppStatus nppsMaxIndxGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMaxIndx_32s.

- NppStatus nppsMaxIndxGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMaxIndx_32f.

- NppStatus nppsMaxIndxGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMaxIndx_64f.

- NppStatus nppsMaxIndx_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMax, int ∗pIndx, Npp8u ∗pDeviceBuffer)

  16-bit integer vector max index method

- NppStatus nppsMaxIndx_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMax, int ∗pIndx, Npp8u ∗pDeviceBuffer)

32-bit integer vector max index method

- NppStatus nppsMaxIndx_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pMax, int ∗pIndx, Npp8u ∗pDeviceBuffer)

  32-bit float vector max index method

- NppStatus nppsMaxIndx_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pMax, int ∗pIndx, Npp8u ∗pDeviceBuffer)

  64-bit float vector max index method

- NppStatus nppsMaxAbsGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMaxAbs_16s.

- NppStatus nppsMaxAbsGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMaxAbs_32s.

- NppStatus nppsMaxAbs_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMaxAbs, Npp8u ∗pDeviceBuffer)

  16-bit integer vector max absolute method

- NppStatus nppsMaxAbs_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMaxAbs, Npp8u ∗pDeviceBuffer)

  32-bit integer vector max absolute method

- NppStatus nppsMaxAbsIndxGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMaxAbsIndx_16s.

- NppStatus nppsMaxAbsIndxGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMaxAbsIndx_32s.

- NppStatus nppsMaxAbsIndx_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMaxAbs, int ∗pIndx, Npp8u ∗pDeviceBuffer)

  16-bit integer vector max absolute index method

- NppStatus nppsMaxAbsIndx_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMaxAbs, int ∗pIndx, Npp8u ∗pDeviceBuffer)

  32-bit integer vector max absolute index method

7.53.1 Function Documentation

7.53.1.1 NppStatus nppsMax_16s (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

16-bit integer vector max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMax Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.2 NppStatus nppsMax_32f (const Npp32f * *pSrc*, int *nLength*, Npp32f * *pMax*, Npp8u * *pDeviceBuffer*)

32-bit float vector max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMax Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.3 NppStatus nppsMax_32s (const Npp32s * *pSrc*, int *nLength*, Npp32s * *pMax*, Npp8u * *pDeviceBuffer*)

32-bit integer vector max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMax Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.4 NppStatus nppsMax_64f (const Npp64f * *pSrc*, int *nLength*, Npp64f * *pMax*, Npp8u * *pDeviceBuffer*)

64-bit float vector max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMax Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.5 NppStatus nppsMaxAbs_16s (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMaxAbs*, Npp8u ∗ *pDeviceBuffer*)

16-bit integer vector max absolute method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMaxAbs Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxAbsGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.6 NppStatus nppsMaxAbs_32s (const Npp32s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pMaxAbs*, Npp8u ∗ *pDeviceBuffer*)

32-bit integer vector max absolute method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMaxAbs Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxAbsGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.7 NppStatus nppsMaxAbsGetBufferSize_16s (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMaxAbs_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53.1.8 NppStatus nppsMaxAbsGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMaxAbs_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53.1.9 NppStatus nppsMaxAbsIndx_16s (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMaxAbs*, int ∗ *pIndx*, Npp8u ∗ *pDeviceBuffer*)

16-bit integer vector max absolute index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMaxAbs Pointer to the output result.

pIndx Pointer to the index value of the first maximum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxAbsIndxGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.10 NppStatus nppsMaxAbsIndx_32s (const Npp32s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pMaxAbs*, int ∗ *pIndx*, Npp8u ∗ *pDeviceBuffer*)

32-bit integer vector max absolute index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMaxAbs Pointer to the output result.

pIndx Pointer to the index value of the first maximum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxAbsIndxGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.11 NppStatus nppsMaxAbsIndxGetBufferSize_16s (int *nLength*, int * *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMaxAbsIndx_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53.1.12 NppStatus nppsMaxAbsIndxGetBufferSize_32s (int *nLength*, int * *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMaxAbsIndx_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53.1.13 NppStatus nppsMaxGetBufferSize_16s (int *nLength*, int * *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMax_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53.1.14 NppStatus nppsMaxGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMax_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53.1.15 NppStatus nppsMaxGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMax_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53.1.16 NppStatus nppsMaxGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMax_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53.1.17 NppStatus nppsMaxIndx_16s (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMax*, int ∗ *pIndx*, Npp8u ∗ *pDeviceBuffer*)

16-bit integer vector max index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMax Pointer to the output result.

pIndx Pointer to the index value of the first maximum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxIndxGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.18 NppStatus nppsMaxIndx_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pMax*, int ∗ *pIndx*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector max index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMax Pointer to the output result.

pIndx Pointer to the index value of the first maximum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxIndxGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.19 NppStatus nppsMaxIndx_32s (const Npp32s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pMax*, int ∗ *pIndx*, Npp8u ∗ *pDeviceBuffer*)

32-bit integer vector max index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMax Pointer to the output result.

pIndx Pointer to the index value of the first maximum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxIndxGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.20 NppStatus nppsMaxIndx_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pMax*, int ∗ *pIndx*, Npp8u ∗ *pDeviceBuffer*)

64-bit float vector max index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMax Pointer to the output result.

pIndx Pointer to the index value of the first maximum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaxIndxGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.53.1.21 NppStatus nppsMaxIndxGetBufferSize_16s (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMaxIndx_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53.1.22 NppStatus nppsMaxIndxGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMaxIndx_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53.1.23 NppStatus nppsMaxIndxGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMaxIndx_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.53.1.24 NppStatus nppsMaxIndxGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMaxIndx_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54 Minimum

Functions

- NppStatus nppsMinGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMin_16s.

- NppStatus nppsMinGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMin_32s.

- NppStatus nppsMinGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMin_32f.

- NppStatus nppsMinGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMin_64f.

- NppStatus nppsMin_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMin, Npp8u ∗pDeviceBuffer)

  16-bit integer vector min method

- NppStatus nppsMin_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMin, Npp8u ∗pDeviceBuffer)

  32-bit integer vector min method

- NppStatus nppsMin_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pMin, Npp8u ∗pDeviceBuffer)

  32-bit integer vector min method

- NppStatus nppsMin_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pMin, Npp8u ∗pDeviceBuffer)

  64-bit integer vector min method

- NppStatus nppsMinIndxGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMinIndx_16s.

- NppStatus nppsMinIndxGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMinIndx_32s.

- NppStatus nppsMinIndxGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMinIndx_32f.

- NppStatus nppsMinIndxGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMinIndx_64f.

- NppStatus nppsMinIndx_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMin, int ∗pIndx, Npp8u ∗pDeviceBuffer)

  16-bit integer vector min index method

- NppStatus nppsMinIndx_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMin, int ∗pIndx, Npp8u ∗pDeviceBuffer)

32-bit integer vector min index method

- NppStatus nppsMinIndx_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pMin, int ∗pIndx, Npp8u ∗pDeviceBuffer)

  32-bit float vector min index method

- NppStatus nppsMinIndx_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pMin, int ∗pIndx, Npp8u ∗pDeviceBuffer)

  64-bit float vector min index method

- NppStatus nppsMinAbsGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMinAbs_16s.

- NppStatus nppsMinAbsGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMinAbs_32s.

- NppStatus nppsMinAbs_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMinAbs, Npp8u ∗pDeviceBuffer)

  16-bit integer vector min absolute method

- NppStatus nppsMinAbs_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMinAbs, Npp8u ∗pDeviceBuffer)

  32-bit integer vector min absolute method

- NppStatus nppsMinAbsIndxGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMinAbsIndx_16s.

- NppStatus nppsMinAbsIndxGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMinAbsIndx_32s.

- NppStatus nppsMinAbsIndx_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMinAbs, int ∗pIndx, Npp8u ∗pDeviceBuffer)

  16-bit integer vector min absolute index method

- NppStatus nppsMinAbsIndx_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMinAbs, int ∗pIndx, Npp8u ∗pDeviceBuffer)

  32-bit integer vector min absolute index method

7.54.1 Function Documentation

7.54.1.1 NppStatus nppsMin_16s (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMin*, Npp8u ∗ *pDeviceBuffer*)

16-bit integer vector min method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.2 NppStatus nppsMin_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pMin*, Npp8u ∗ *pDeviceBuffer*)

32-bit integer vector min method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.3 NppStatus nppsMin_32s (const Npp32s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pMin*, Npp8u ∗ *pDeviceBuffer*)

32-bit integer vector min method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.4 NppStatus nppsMin_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pMin*, Npp8u ∗ *pDeviceBuffer*)

64-bit integer vector min method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.5 NppStatus nppsMinAbs_16s (const Npp16s * *pSrc*, int *nLength*, Npp16s * *pMinAbs*, Npp8u * *pDeviceBuffer*)

16-bit integer vector min absolute method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMinAbs Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinAbsGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.6 NppStatus nppsMinAbs_32s (const Npp32s * *pSrc*, int *nLength*, Npp32s * *pMinAbs*, Npp8u * *pDeviceBuffer*)

32-bit integer vector min absolute method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMinAbs Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinAbsGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.7 NppStatus nppsMinAbsGetBufferSize_16s (int *nLength*, int * *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMinAbs_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54.1.8 NppStatus nppsMinAbsGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMinAbs_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54.1.9 NppStatus nppsMinAbsIndx_16s (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMinAbs*, int ∗ *pIndx*, Npp8u ∗ *pDeviceBuffer*)

16-bit integer vector min absolute index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMinAbs Pointer to the output result.

pIndx Pointer to the index value of the first minimum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinAbsIndxGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.10 NppStatus nppsMinAbsIndx_32s (const Npp32s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pMinAbs*, int ∗ *pIndx*, Npp8u ∗ *pDeviceBuffer*)

32-bit integer vector min absolute index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMinAbs Pointer to the output result.

pIndx Pointer to the index value of the first minimum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinAbsIndxGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.11 NppStatus nppsMinAbsIndxGetBufferSize_16s (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMinAbsIndx_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54.1.12 NppStatus nppsMinAbsIndxGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMinAbsIndx_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54.1.13 NppStatus nppsMinGetBufferSize_16s (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMin_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54.1.14 NppStatus nppsMinGetBufferSize_32f (int *nLength*, int $*$ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMin_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54.1.15 NppStatus nppsMinGetBufferSize_32s (int *nLength*, int $*$ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMin_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54.1.16 NppStatus nppsMinGetBufferSize_64f (int *nLength*, int $*$ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMin_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54.1.17 NppStatus nppsMinIndx_16s (const Npp16s $*$ *pSrc*, int *nLength*, Npp16s $*$ *pMin*, int $*$ *pIndx*, Npp8u $*$ *pDeviceBuffer*)

16-bit integer vector min index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pIndx Pointer to the index value of the first minimum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinIndxGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.18 NppStatus nppsMinIndx_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pMin*, int ∗ *pIndx*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector min index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pIndx Pointer to the index value of the first minimum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinIndxGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.19 NppStatus nppsMinIndx_32s (const Npp32s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pMin*, int ∗ *pIndx*, Npp8u ∗ *pDeviceBuffer*)

32-bit integer vector min index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pIndx Pointer to the index value of the first minimum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinIndxGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.20 NppStatus nppsMinIndx_64f (const Npp64f * *pSrc*, int *nLength*, Npp64f * *pMin*, int * *pIndx*, Npp8u * *pDeviceBuffer*)

64-bit float vector min index method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the output result.

pIndx Pointer to the index value of the first minimum element.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinIndxGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.54.1.21 NppStatus nppsMinIndxGetBufferSize_16s (int *nLength*, int * *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMinIndx_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54.1.22 NppStatus nppsMinIndxGetBufferSize_32f (int *nLength*, int * *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMinIndx_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54.1.23 NppStatus nppsMinIndxGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMinIndx_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.54.1.24 NppStatus nppsMinIndxGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMinIndx_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.55 Mean

Functions

- NppStatus nppsMeanGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMean_32f.

- NppStatus nppsMeanGetBufferSize_32fc (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMean_32fc.

- NppStatus nppsMeanGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMean_64f.

- NppStatus nppsMeanGetBufferSize_64fc (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMean_64fc.

- NppStatus nppsMeanGetBufferSize_16s_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMean_16s_Sfs.

- NppStatus nppsMeanGetBufferSize_32s_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMean_32s_Sfs.

- NppStatus nppsMeanGetBufferSize_16sc_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMean_16sc_Sfs.

- NppStatus nppsMean_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pMean, Npp8u ∗pDeviceBuffer)

  32-bit float vector mean method

- NppStatus nppsMean_32fc (const Npp32fc ∗pSrc, int nLength, Npp32fc ∗pMean, Npp8u ∗pDeviceBuffer)

  32-bit float complex vector mean method

- NppStatus nppsMean_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pMean, Npp8u ∗pDeviceBuffer)

  64-bit double vector mean method

- NppStatus nppsMean_64fc (const Npp64fc ∗pSrc, int nLength, Npp64fc ∗pMean, Npp8u ∗pDeviceBuffer)

  64-bit double complex vector mean method

- NppStatus nppsMean_16s_Sfs (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMean, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit short vector mean with integer scaling method

- NppStatus nppsMean_32s_Sfs (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMean, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  32-bit integer vector mean with integer scaling method

- NppStatus nppsMean_16sc_Sfs (const Npp16sc ∗pSrc, int nLength, Npp16sc ∗pMean, int nScaleFactor, Npp8u ∗pDeviceBuffer)

16-bit short complex vector mean with integer scaling method

7.55.1 Function Documentation

7.55.1.1 NppStatus nppsMean_16s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMean*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit short vector mean with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMean Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMeanGetBufferSize_16s_Sfs to determine the minium number of bytes required.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.55.1.2 NppStatus nppsMean_16sc_Sfs (const Npp16sc ∗ *pSrc*, int *nLength*, Npp16sc ∗ *pMean*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit short complex vector mean with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMean Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMeanGetBufferSize_16sc_Sfs to determine the minium number of bytes required.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.55.1.3 NppStatus nppsMean_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pMean*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector mean method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMean Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMeanGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.55.1.4 NppStatus nppsMean_32fc (const Npp32fc ∗ *pSrc*, int *nLength*, Npp32fc ∗ *pMean*, Npp8u ∗ *pDeviceBuffer*)

32-bit float complex vector mean method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMean Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMeanGetBufferSize_32fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.55.1.5 NppStatus nppsMean_32s_Sfs (const Npp32s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pMean*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

32-bit integer vector mean with integer scaling method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMean Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMeanGetBufferSize_32s_Sfs to determine the minium number of bytes required.

nScaleFactor Integer Result Scaling.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.55.1.6 NppStatus nppsMean_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pMean*, Npp8u ∗ *pDeviceBuffer*)

64-bit double vector mean method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMean Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMeanGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.55.1.7 NppStatus nppsMean_64fc (const Npp64fc ∗ *pSrc*, int *nLength*, Npp64fc ∗ *pMean*, Npp8u ∗ *pDeviceBuffer*)

64-bit double complex vector mean method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMean Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMeanGetBufferSize_64fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.55.1.8 NppStatus nppsMeanGetBufferSize_16s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMean_16s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.55.1.9 NppStatus nppsMeanGetBufferSize_16sc_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMean_16sc_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.55.1.10 NppStatus nppsMeanGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMean_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.55.1.11 NppStatus nppsMeanGetBufferSize_32fc (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMean_32fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.55.1.12 NppStatus nppsMeanGetBufferSize_32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMean_32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.55.1.13 NppStatus nppsMeanGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMean_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.55.1.14 NppStatus nppsMeanGetBufferSize_64fc (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMean_64fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.56 Standard Deviation

Functions

- NppStatus nppsStdDevGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsStdDev_32f.
- NppStatus nppsStdDevGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsStdDev_64f.
- NppStatus nppsStdDevGetBufferSize_16s32s_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsStdDev_16s32s_Sfs.
- NppStatus nppsStdDevGetBufferSize_16s_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsStdDev_16s_Sfs.
- NppStatus nppsStdDev_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pStdDev, Npp8u ∗pDeviceBuffer)

  32-bit float vector standard deviation method
- NppStatus nppsStdDev_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pStdDev, Npp8u ∗pDeviceBuffer)

  64-bit float vector standard deviation method
- NppStatus nppsStdDev_16s32s_Sfs (const Npp16s ∗pSrc, int nLength, Npp32s ∗pStdDev, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit float vector standard deviation method (return value is 32-bit)
- NppStatus nppsStdDev_16s_Sfs (const Npp16s ∗pSrc, int nLength, Npp16s ∗pStdDev, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit float vector standard deviation method (return value is also 16-bit)

7.56.1 Function Documentation

7.56.1.1 NppStatus nppsStdDev_16s32s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pStdDev*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit float vector standard deviation method (return value is 32-bit)

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pStdDev Pointer to the output result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsStdDevGetBufferSize_16s32s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.56.1.2 NppStatus nppsStdDev_16s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pStdDev*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit float vector standard deviation method (return value is also 16-bit)

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pStdDev Pointer to the output result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsStdDevGetBufferSize_16s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.56.1.3 NppStatus nppsStdDev_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pStdDev*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector standard deviation method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pStdDev Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsStdDevGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.56.1.4 NppStatus nppsStdDev_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pStdDev*, Npp8u ∗ *pDeviceBuffer*)

64-bit float vector standard deviation method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pStdDev Pointer to the output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsStdDevGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.56.1.5 NppStatus nppsStdDevGetBufferSize_16s32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsStdDev_16s32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.56.1.6 NppStatus nppsStdDevGetBufferSize_16s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsStdDev_16s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.56.1.7 NppStatus nppsStdDevGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsStdDev_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.56.1.8 NppStatus nppsStdDevGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsStdDev_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.57 Mean And Standard Deviation

Functions

- NppStatus nppsMeanStdDevGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMeanStdDev_32f.

- NppStatus nppsMeanStdDevGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMeanStdDev_64f.

- NppStatus nppsMeanStdDevGetBufferSize_16s32s_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMeanStdDev_16s32s_Sfs.

- NppStatus nppsMeanStdDevGetBufferSize_16s_Sfs (int nLength, int ∗hpBufferSize)

  Device scratch buffer size (in bytes) for nppsMeanStdDev_16s_Sfs.

- NppStatus nppsMeanStdDev_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pMean, Npp32f ∗pStdDev, Npp8u ∗pDeviceBuffer)

  32-bit float vector mean and standard deviation method

- NppStatus nppsMeanStdDev_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pMean, Npp64f ∗pStdDev, Npp8u ∗pDeviceBuffer)

  64-bit float vector mean and standard deviation method

- NppStatus nppsMeanStdDev_16s32s_Sfs (const Npp16s ∗pSrc, int nLength, Npp32s ∗pMean, Npp32s ∗pStdDev, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit float vector mean and standard deviation method (return values are 32-bit)

- NppStatus nppsMeanStdDev_16s_Sfs (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMean, Npp16s ∗pStdDev, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit float vector mean and standard deviation method (return values are also 16-bit)

7.57.1 Function Documentation

7.57.1.1 NppStatus nppsMeanStdDev_16s32s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pMean*, Npp32s ∗ *pStdDev*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit float vector mean and standard deviation method (return values are 32-bit)

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMean Pointer to the output mean value.

pStdDev Pointer to the output standard deviation value.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMeanStdDevGetBufferSize_16s32s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.57.1.2 NppStatus nppsMeanStdDev_16s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMean*, Npp16s ∗ *pStdDev*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit float vector mean and standard deviation method (return values are also 16-bit)

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMean Pointer to the output mean value.

pStdDev Pointer to the output standard deviation value.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMeanStdDevGetBufferSize_16s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.57.1.3 NppStatus nppsMeanStdDev_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pMean*, Npp32f ∗ *pStdDev*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector mean and standard deviation method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMean Pointer to the output mean value.

pStdDev Pointer to the output standard deviation value.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMeanStdDevGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.57.1.4 NppStatus nppsMeanStdDev_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pMean*, Npp64f ∗ *pStdDev*, Npp8u ∗ *pDeviceBuffer*)

64-bit float vector mean and standard deviation method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMean Pointer to the output mean value.

pStdDev Pointer to the output standard deviation value.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMeanStdDevGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.57.1.5 NppStatus nppsMeanStdDevGetBufferSize_16s32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMeanStdDev_16s32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.57.1.6 NppStatus nppsMeanStdDevGetBufferSize_16s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMeanStdDev_16s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.57.1.7 NppStatus nppsMeanStdDevGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMeanStdDev_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.57.1.8 NppStatus nppsMeanStdDevGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device scratch buffer size (in bytes) for nppsMeanStdDev_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.* Scratch Buffer and Host Pointer.

Returns:

NPP_SUCCESS

7.58 Minimum_Maximum

Functions

- NppStatus nppsMinMaxGetBufferSize_8u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_8u.

- NppStatus nppsMinMaxGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_16s.

- NppStatus nppsMinMaxGetBufferSize_16u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_16u.

- NppStatus nppsMinMaxGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_32s.

- NppStatus nppsMinMaxGetBufferSize_32u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_32u.

- NppStatus nppsMinMaxGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_32f.

- NppStatus nppsMinMaxGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMinMax_64f.

- NppStatus nppsMinMax_8u (const Npp8u ∗pSrc, int nLength, Npp8u ∗pMin, Npp8u ∗pMax, Npp8u ∗pDeviceBuffer)

  8-bit char vector min and max method

- NppStatus nppsMinMax_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMin, Npp16s ∗pMax, Npp8u ∗pDeviceBuffer)

  16-bit signed short vector min and max method

- NppStatus nppsMinMax_16u (const Npp16u ∗pSrc, int nLength, Npp16u ∗pMin, Npp16u ∗pMax, Npp8u ∗pDeviceBuffer)

  16-bit unsigned short vector min and max method

- NppStatus nppsMinMax_32u (const Npp32u ∗pSrc, int nLength, Npp32u ∗pMin, Npp32u ∗pMax, Npp8u ∗pDeviceBuffer)

  32-bit unsigned int vector min and max method

- NppStatus nppsMinMax_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMin, Npp32s ∗pMax, Npp8u ∗pDeviceBuffer)

  32-bit signed int vector min and max method

- NppStatus nppsMinMax_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pMin, Npp32f ∗pMax, Npp8u ∗pDeviceBuffer)

  32-bit float vector min and max method

- NppStatus nppsMinMax_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pMin, Npp64f ∗pMax, Npp8u ∗pDeviceBuffer)

64-bit double vector min and max method

- NppStatus nppsMinMaxIndxGetBufferSize_8u (int nLength, int ∗hpBufferSize)
  Device-buffer size (in bytes) for nppsMinMaxIndx_8u.
- NppStatus nppsMinMaxIndxGetBufferSize_16s (int nLength, int ∗hpBufferSize)
  Device-buffer size (in bytes) for nppsMinMaxIndx_16s.
- NppStatus nppsMinMaxIndxGetBufferSize_16u (int nLength, int ∗hpBufferSize)
  Device-buffer size (in bytes) for nppsMinMaxIndx_16u.
- NppStatus nppsMinMaxIndxGetBufferSize_32s (int nLength, int ∗hpBufferSize)
  Device-buffer size (in bytes) for nppsMinMaxIndx_32s.
- NppStatus nppsMinMaxIndxGetBufferSize_32u (int nLength, int ∗hpBufferSize)
  Device-buffer size (in bytes) for nppsMinMaxIndx_32u.
- NppStatus nppsMinMaxIndxGetBufferSize_32f (int nLength, int ∗hpBufferSize)
  Device-buffer size (in bytes) for nppsMinMaxIndx_32f.
- NppStatus nppsMinMaxIndxGetBufferSize_64f (int nLength, int ∗hpBufferSize)
  Device-buffer size (in bytes) for nppsMinMaxIndx_64f.
- NppStatus nppsMinMaxIndx_8u (const Npp8u ∗pSrc, int nLength, Npp8u ∗pMin, int ∗pMinIndx, Npp8u ∗pMax, int ∗pMaxIndx, Npp8u ∗pDeviceBuffer)
  8-bit char vector min and max with indices method
- NppStatus nppsMinMaxIndx_16s (const Npp16s ∗pSrc, int nLength, Npp16s ∗pMin, int ∗pMinIndx, Npp16s ∗pMax, int ∗pMaxIndx, Npp8u ∗pDeviceBuffer)
  16-bit signed short vector min and max with indices method
- NppStatus nppsMinMaxIndx_16u (const Npp16u ∗pSrc, int nLength, Npp16u ∗pMin, int ∗pMinIndx, Npp16u ∗pMax, int ∗pMaxIndx, Npp8u ∗pDeviceBuffer)
  16-bit unsigned short vector min and max with indices method
- NppStatus nppsMinMaxIndx_32s (const Npp32s ∗pSrc, int nLength, Npp32s ∗pMin, int ∗pMinIndx, Npp32s ∗pMax, int ∗pMaxIndx, Npp8u ∗pDeviceBuffer)
  32-bit signed short vector min and max with indices method
- NppStatus nppsMinMaxIndx_32u (const Npp32u ∗pSrc, int nLength, Npp32u ∗pMin, int ∗pMinIndx, Npp32u ∗pMax, int ∗pMaxIndx, Npp8u ∗pDeviceBuffer)
  32-bit unsigned short vector min and max with indices method
- NppStatus nppsMinMaxIndx_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pMin, int ∗pMinIndx, Npp32f ∗pMax, int ∗pMaxIndx, Npp8u ∗pDeviceBuffer)
  32-bit float vector min and max with indices method
- NppStatus nppsMinMaxIndx_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pMin, int ∗pMinIndx, Npp64f ∗pMax, int ∗pMaxIndx, Npp8u ∗pDeviceBuffer)
  64-bit float vector min and max with indices method

7.58.1 Function Documentation

7.58.1.1 NppStatus nppsMinMax_16s (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMin*, Npp16s ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.2 NppStatus nppsMinMax_16u (const Npp16u ∗ *pSrc*, int *nLength*, Npp16u ∗ *pMin*, Npp16u ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

16-bit unsigned short vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxGetBufferSize_16u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.3 NppStatus nppsMinMax_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pMin*, Npp32f ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.4 NppStatus nppsMinMax_32s (const Npp32s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pMin*, Npp32s ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

32-bit signed int vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.5 NppStatus nppsMinMax_32u (const Npp32u ∗ *pSrc*, int *nLength*, Npp32u ∗ *pMin*, Npp32u ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

32-bit unsigned int vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxGetBufferSize_32u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.6 NppStatus nppsMinMax_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pMin*, Npp64f ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

64-bit double vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.7 NppStatus nppsMinMax_8u (const Npp8u ∗ *pSrc*, int *nLength*, Npp8u ∗ *pMin*, Npp8u ∗ *pMax*, Npp8u ∗ *pDeviceBuffer*)

8-bit char vector min and max method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMax Pointer to the max output result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxGetBufferSize_8u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.8 NppStatus nppsMinMaxGetBufferSize_16s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.9 NppStatus nppsMinMaxGetBufferSize_16u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_16u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.10 NppStatus nppsMinMaxGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.11 NppStatus nppsMinMaxGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.12 NppStatus nppsMinMaxGetBufferSize_32u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_32u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.13 NppStatus nppsMinMaxGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.14 NppStatus nppsMinMaxGetBufferSize_8u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMax_8u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.15 NppStatus nppsMinMaxIndx_16s (const Npp16s ∗ *pSrc*, int *nLength*, Npp16s ∗ *pMin*, int ∗ *pMinIndx*, Npp16s ∗ *pMax*, int ∗ *pMaxIndx*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short vector min and max with indices method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMinIndx Pointer to the index of the first min value.

pMax Pointer to the max output result.

pMaxIndx Pointer to the index of the first max value.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxIndxGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.16 NppStatus nppsMinMaxIndx_16u (const Npp16u ∗ *pSrc*, int *nLength*, Npp16u ∗ *pMin*, int ∗ *pMinIndx*, Npp16u ∗ *pMax*, int ∗ *pMaxIndx*, Npp8u ∗ *pDeviceBuffer*)

16-bit unsigned short vector min and max with indices method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMinIndx Pointer to the index of the first min value.

pMax Pointer to the max output result.

pMaxIndx Pointer to the index of the first max value.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxIndxGetBufferSize_16u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.17 NppStatus nppsMinMaxIndx_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pMin*, int ∗ *pMinIndx*, Npp32f ∗ *pMax*, int ∗ *pMaxIndx*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector min and max with indices method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMinIndx Pointer to the index of the first min value.

pMax Pointer to the max output result.

pMaxIndx Pointer to the index of the first max value.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxIndxGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.18 NppStatus nppsMinMaxIndx_32s (const Npp32s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pMin*, int ∗ *pMinIndx*, Npp32s ∗ *pMax*, int ∗ *pMaxIndx*, Npp8u ∗ *pDeviceBuffer*)

32-bit signed short vector min and max with indices method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMinIndx Pointer to the index of the first min value.

pMax Pointer to the max output result.

pMaxIndx Pointer to the index of the first max value.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxIndxGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.19 NppStatus nppsMinMaxIndx_32u (const Npp32u ∗ *pSrc*, int *nLength*, Npp32u ∗ *pMin*, int ∗ *pMinIndx*, Npp32u ∗ *pMax*, int ∗ *pMaxIndx*, Npp8u ∗ *pDeviceBuffer*)

32-bit unsigned short vector min and max with indices method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMinIndx Pointer to the index of the first min value.

pMax Pointer to the max output result.

pMaxIndx Pointer to the index of the first max value.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxIndxGetBufferSize_32u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.20 NppStatus nppsMinMaxIndx_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pMin*, int ∗ *pMinIndx*, Npp64f ∗ *pMax*, int ∗ *pMaxIndx*, Npp8u ∗ *pDeviceBuffer*)

64-bit float vector min and max with indices method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMinIndx Pointer to the index of the first min value.

pMax Pointer to the max output result.

pMaxIndx Pointer to the index of the first max value.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxIndxGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.21 NppStatus nppsMinMaxIndx_8u (const Npp8u ∗ *pSrc*, int *nLength*, Npp8u ∗ *pMin*, int ∗ *pMinIndx*, Npp8u ∗ *pMax*, int ∗ *pMaxIndx*, Npp8u ∗ *pDeviceBuffer*)

8-bit char vector min and max with indices method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pMin Pointer to the min output result.

pMinIndx Pointer to the index of the first min value.

pMax Pointer to the max output result.

pMaxIndx Pointer to the index of the first max value.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMinMaxIndxGetBufferSize_8u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.58.1.22 NppStatus nppsMinMaxIndxGetBufferSize_16s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMaxIndx_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.23 NppStatus nppsMinMaxIndxGetBufferSize_16u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMaxIndx_16u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.24 NppStatus nppsMinMaxIndxGetBufferSize_32f (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMaxIndx_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.25 NppStatus nppsMinMaxIndxGetBufferSize_32s (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMaxIndx_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.26 NppStatus nppsMinMaxIndxGetBufferSize_32u (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMaxIndx_32u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.27 NppStatus nppsMinMaxIndxGetBufferSize_64f (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMaxIndx_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.58.1.28 NppStatus nppsMinMaxIndxGetBufferSize_8u (int *nLength*, int $*$ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMinMaxIndx_8u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.59 Infinity Norm

Functions

- NppStatus nppsNormInfGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_Inf_32f.

- NppStatus nppsNorm_Inf_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float vector C norm method

- NppStatus nppsNormInfGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_Inf_64f.

- NppStatus nppsNorm_Inf_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float vector C norm method

- NppStatus nppsNormInfGetBufferSize_16s32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_Inf_16s32f.

- NppStatus nppsNorm_Inf_16s32f (const Npp16s ∗pSrc, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer vector C norm method, return value is 32-bit float.

- NppStatus nppsNormInfGetBufferSize_32fc32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_Inf_32fc32f.

- NppStatus nppsNorm_Inf_32fc32f (const Npp32fc ∗pSrc, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float complex vector C norm method, return value is 32-bit float.

- NppStatus nppsNormInfGetBufferSize_64fc64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_Inf_64fc64f.

- NppStatus nppsNorm_Inf_64fc64f (const Npp64fc ∗pSrc, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float complex vector C norm method, return value is 64-bit float.

- NppStatus nppsNormInfGetBufferSize_16s32s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_Inf_16s32s_Sfs.

- NppStatus nppsNorm_Inf_16s32s_Sfs (const Npp16s ∗pSrc, int nLength, Npp32s ∗pNorm, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer vector C norm method, return value is 32-bit signed integer.

7.59.1 Function Documentation

7.59.1.1 NppStatus nppsNorm_Inf_16s32f (const Npp16s ∗ *pSrc*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer vector C norm method, return value is 32-bit float.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormInfGetBufferSize_16s32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.59.1.2 NppStatus nppsNorm_Inf_16s32s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pNorm*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer vector C norm method, return value is 32-bit signed integer.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormInfGetBufferSize_16s32s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.59.1.3 NppStatus nppsNorm_Inf_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector C norm method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormInfGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.59.1.4 NppStatus nppsNorm_Inf_32fc32f (const Npp32fc ∗ *pSrc*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

32-bit float complex vector C norm method, return value is 32-bit float.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormInfGetBufferSize_32fc32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.59.1.5 NppStatus nppsNorm_Inf_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float vector C norm method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormInfGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.59.1.6 NppStatus nppsNorm_Inf_64fc64f (const Npp64fc ∗ *pSrc*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float complex vector C norm method, return value is 64-bit float.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormInfGetBufferSize_64fc64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.59.1.7 NppStatus nppsNormInfGetBufferSize_16s32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_Inf_16s32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.59.1.8 NppStatus nppsNormInfGetBufferSize_16s32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_Inf_16s32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.59.1.9 NppStatus nppsNormInfGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_Inf_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.59.1.10 NppStatus nppsNormInfGetBufferSize_32fc32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_Inf_32fc32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.59.1.11 NppStatus nppsNormInfGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_Inf_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.59.1.12 NppStatus nppsNormInfGetBufferSize_64fc64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_Inf_64fc64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.60 L1 Norm

Functions

- NppStatus nppsNormL1GetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L1_32f.

- NppStatus nppsNorm_L1_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float vector L1 norm method

- NppStatus nppsNormL1GetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L1_64f.

- NppStatus nppsNorm_L1_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float vector L1 norm method

- NppStatus nppsNormL1GetBufferSize_16s32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L1_16s32f.

- NppStatus nppsNorm_L1_16s32f (const Npp16s ∗pSrc, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer vector L1 norm method, return value is 32-bit float.

- NppStatus nppsNormL1GetBufferSize_32fc64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L1_32fc64f.

- NppStatus nppsNorm_L1_32fc64f (const Npp32fc ∗pSrc, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float complex vector L1 norm method, return value is 64-bit float.

- NppStatus nppsNormL1GetBufferSize_64fc64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L1_64fc64f.

- NppStatus nppsNorm_L1_64fc64f (const Npp64fc ∗pSrc, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float complex vector L1 norm method, return value is 64-bit float.

- NppStatus nppsNormL1GetBufferSize_16s32s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L1_16s32s_Sfs.

- NppStatus nppsNorm_L1_16s32s_Sfs (const Npp16s ∗pSrc, int nLength, Npp32s ∗pNorm, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer vector L1 norm method, return value is 32-bit signed integer.

- NppStatus nppsNormL1GetBufferSize_16s64s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L1_16s64s_Sfs.

- NppStatus nppsNorm_L1_16s64s_Sfs (const Npp16s ∗pSrc, int nLength, Npp64s ∗pNorm, int nScaleFactor, Npp8u ∗pDeviceBuffer)

16-bit signed short integer vector L1 norm method, return value is 64-bit signed integer.

7.60.1 Function Documentation

7.60.1.1 NppStatus nppsNorm_L1_16s32f (const Npp16s * *pSrc*, int *nLength*, Npp32f * *pNorm*, Npp8u * *pDeviceBuffer*)

16-bit signed short integer vector L1 norm method, return value is 32-bit float.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the L1 norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL1GetBufferSize_16s32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.60.1.2 NppStatus nppsNorm_L1_16s32s_Sfs (const Npp16s * *pSrc*, int *nLength*, Npp32s * *pNorm*, int *nScaleFactor*, Npp8u * *pDeviceBuffer*)

16-bit signed short integer vector L1 norm method, return value is 32-bit signed integer.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL1GetBufferSize_16s32s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.60.1.3 NppStatus nppsNorm_L1_16s64s_Sfs (const Npp16s * *pSrc*, int *nLength*, Npp64s * *pNorm*, int *nScaleFactor*, Npp8u * *pDeviceBuffer*)

16-bit signed short integer vector L1 norm method, return value is 64-bit signed integer.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL1GetBufferSize_16s64s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.60.1.4 NppStatus nppsNorm_L1_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector L1 norm method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL1GetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.60.1.5 NppStatus nppsNorm_L1_32fc64f (const Npp32fc ∗ *pSrc*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

32-bit float complex vector L1 norm method, return value is 64-bit float.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL1GetBufferSize_32fc64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.60.1.6 NppStatus nppsNorm_L1_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float vector L1 norm method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL1GetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.60.1.7 NppStatus nppsNorm_L1_64fc64f (const Npp64fc ∗ *pSrc*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float complex vector L1 norm method, return value is 64-bit float.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL1GetBufferSize_64fc64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.60.1.8 NppStatus nppsNormL1GetBufferSize_16s32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L1_16s32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.60.1.9 NppStatus nppsNormL1GetBufferSize_16s32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L1_16s32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.60.1.10 NppStatus nppsNormL1GetBufferSize_16s64s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L1_16s64s_Sfs.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.60.1.11 NppStatus nppsNormL1GetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L1_32f.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.60.1.12 NppStatus nppsNormL1GetBufferSize_32fc64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L1_32fc64f.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.60.1.13 NppStatus nppsNormL1GetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L1_64f.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.60.1.14 NppStatus nppsNormL1GetBufferSize_64fc64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L1_64fc64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.61 L2 Norm

Functions

- NppStatus nppsNormL2GetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L2_32f.

- NppStatus nppsNorm_L2_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float vector L2 norm method

- NppStatus nppsNormL2GetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L2_64f.

- NppStatus nppsNorm_L2_64f (const Npp64f ∗pSrc, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float vector L2 norm method

- NppStatus nppsNormL2GetBufferSize_16s32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L2_16s32f.

- NppStatus nppsNorm_L2_16s32f (const Npp16s ∗pSrc, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer vector L2 norm method, return value is 32-bit float.

- NppStatus nppsNormL2GetBufferSize_32fc64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L2_32fc64f.

- NppStatus nppsNorm_L2_32fc64f (const Npp32fc ∗pSrc, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float complex vector L2 norm method, return value is 64-bit float.

- NppStatus nppsNormL2GetBufferSize_64fc64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L2_64fc64f.

- NppStatus nppsNorm_L2_64fc64f (const Npp64fc ∗pSrc, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float complex vector L2 norm method, return value is 64-bit float.

- NppStatus nppsNormL2GetBufferSize_16s32s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L2_16s32s_Sfs.

- NppStatus nppsNorm_L2_16s32s_Sfs (const Npp16s ∗pSrc, int nLength, Npp32s ∗pNorm, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer vector L2 norm method, return value is 32-bit signed integer.

- NppStatus nppsNormL2SqrGetBufferSize_16s64s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNorm_L2Sqr_16s64s_Sfs.

- NppStatus nppsNorm_L2Sqr_16s64s_Sfs (const Npp16s ∗pSrc, int nLength, Npp64s ∗pNorm, int nScaleFactor, Npp8u ∗pDeviceBuffer)

16-bit signed short integer vector L2 Square norm method, return value is 64-bit signed integer.

7.61.1 Function Documentation

7.61.1.1 NppStatus nppsNorm_L2_16s32f (const Npp16s ∗ *pSrc*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer vector L2 norm method, return value is 32-bit float.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL2GetBufferSize_16s32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.61.1.2 NppStatus nppsNorm_L2_16s32s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp32s ∗ *pNorm*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer vector L2 norm method, return value is 32-bit signed integer.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL2GetBufferSize_16s32s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.61.1.3 NppStatus nppsNorm_L2_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

32-bit float vector L2 norm method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL2GetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.61.1.4 NppStatus nppsNorm_L2_32fc64f (const Npp32fc ∗ *pSrc*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

32-bit float complex vector L2 norm method, return value is 64-bit float.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL2GetBufferSize_32fc64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.61.1.5 NppStatus nppsNorm_L2_64f (const Npp64f ∗ *pSrc*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float vector L2 norm method

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL2GetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.61.1.6 NppStatus nppsNorm_L2_64fc64f (const Npp64fc ∗ *pSrc*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float complex vector L2 norm method, return value is 64-bit float.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL2GetBufferSize_64fc64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.61.1.7 NppStatus nppsNorm_L2Sqr_16s64s_Sfs (const Npp16s ∗ *pSrc*, int *nLength*, Npp64s ∗ *pNorm*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer vector L2 Square norm method, return value is 64-bit signed integer.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormL2SqrGetBufferSize_16s64s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.61.1.8 NppStatus nppsNormL2GetBufferSize_16s32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L2_16s32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.61.1.9 NppStatus nppsNormL2GetBufferSize_16s32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L2_16s32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.61.1.10 NppStatus nppsNormL2GetBufferSize_32f (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L2_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.61.1.11 NppStatus nppsNormL2GetBufferSize_32fc64f (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L2_32fc64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.61.1.12 NppStatus nppsNormL2GetBufferSize_64f (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L2_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.61.1.13 NppStatus nppsNormL2GetBufferSize_64fc64f (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L2_64fc64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.61.1.14 NppStatus nppsNormL2SqrGetBufferSize_16s64s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNorm_L2Sqr_16s64s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.62 Infinity Norm Diff

Functions

- NppStatus nppsNormDiffInfGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_Inf_32f.

- NppStatus nppsNormDiff_Inf_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float C norm method on two vectors' difference

- NppStatus nppsNormDiffInfGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_Inf_64f.

- NppStatus nppsNormDiff_Inf_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float C norm method on two vectors' difference

- NppStatus nppsNormDiffInfGetBufferSize_16s32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_Inf_16s32f.

- NppStatus nppsNormDiff_Inf_16s32f (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer C norm method on two vectors' difference, return value is 32-bit float.

- NppStatus nppsNormDiffInfGetBufferSize_32fc32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_Inf_32fc32f.

- NppStatus nppsNormDiff_Inf_32fc32f (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float complex C norm method on two vectors' difference, return value is 32-bit float.

- NppStatus nppsNormDiffInfGetBufferSize_64fc64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_Inf_64fc64f.

- NppStatus nppsNormDiff_Inf_64fc64f (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float complex C norm method on two vectors' difference, return value is 64-bit float.

- NppStatus nppsNormDiffInfGetBufferSize_16s32s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_Inf_16s32s_Sfs.

- NppStatus nppsNormDiff_Inf_16s32s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp32s ∗pNorm, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer C norm method on two vectors' difference, return value is 32-bit signed integer.

7.62.1 Function Documentation

7.62.1.1 NppStatus nppsNormDiff_Inf_16s32f (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer C norm method on two vectors' difference, return value is 32-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffInfGetBufferSize_16s32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.62.1.2 NppStatus nppsNormDiff_Inf_16s32s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp32s ∗ *pNorm*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer C norm method on two vectors' difference, return value is 32-bit signed integer.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffInfGetBufferSize_16s32s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.62.1.3 NppStatus nppsNormDiff_Inf_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

32-bit float C norm method on two vectors' difference

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffInfGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.62.1.4 NppStatus nppsNormDiff_Inf_32fc32f (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

32-bit float complex C norm method on two vectors' difference, return value is 32-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffInfGetBufferSize_32fc32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.62.1.5 NppStatus nppsNormDiff_Inf_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float C norm method on two vectors' difference

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffInfGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.62.1.6 NppStatus nppsNormDiff_Inf_64fc64f (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float complex C norm method on two vectors' difference, return value is 64-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffInfGetBufferSize_64fc64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.62.1.7 NppStatus nppsNormDiffInfGetBufferSize_16s32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_Inf_16s32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.62.1.8 NppStatus nppsNormDiffInfGetBufferSize_16s32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_Inf_16s32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.62.1.9 NppStatus nppsNormDiffInfGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_Inf_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.62.1.10 NppStatus nppsNormDiffInfGetBufferSize_32fc32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_Inf_32fc32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.62.1.11 NppStatus nppsNormDiffInfGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_Inf_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.62.1.12 NppStatus nppsNormDiffInfGetBufferSize_64fc64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_Inf_64fc64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.63 L1 Norm Diff

Functions

- NppStatus nppsNormDiffL1GetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L1_32f.

- NppStatus nppsNormDiff_L1_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float L1 norm method on two vectors' difference

- NppStatus nppsNormDiffL1GetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L1_64f.

- NppStatus nppsNormDiff_L1_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float L1 norm method on two vectors' difference

- NppStatus nppsNormDiffL1GetBufferSize_16s32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L1_16s32f.

- NppStatus nppsNormDiff_L1_16s32f (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer L1 norm method on two vectors' difference, return value is 32-bit float.

- NppStatus nppsNormDiffL1GetBufferSize_32fc64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L1_32fc64f.

- NppStatus nppsNormDiff_L1_32fc64f (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float complex L1 norm method on two vectors' difference, return value is 64-bit float.

- NppStatus nppsNormDiffL1GetBufferSize_64fc64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L1_64fc64f.

- NppStatus nppsNormDiff_L1_64fc64f (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float complex L1 norm method on two vectors' difference, return value is 64-bit float.

- NppStatus nppsNormDiffL1GetBufferSize_16s32s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L1_16s32s_Sfs.

- NppStatus nppsNormDiff_L1_16s32s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp32s ∗pNorm, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer L1 norm method on two vectors' difference, return value is 32-bit signed integer.

- NppStatus nppsNormDiffL1GetBufferSize_16s64s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L1_16s64s_Sfs.

- NppStatus nppsNormDiff_L1_16s64s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp64s ∗pNorm, int nScaleFactor, Npp8u ∗pDeviceBuffer)

16-bit signed short integer L1 norm method on two vectors' difference, return value is 64-bit signed integer.

7.63.1 Function Documentation

7.63.1.1 NppStatus nppsNormDiff_L1_16s32f (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer L1 norm method on two vectors' difference, return value is 32-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the L1 norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL1GetBufferSize_16s32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.63.1.2 NppStatus nppsNormDiff_L1_16s32s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp32s ∗ *pNorm*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer L1 norm method on two vectors' difference, return value is 32-bit signed integer.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer..

nLength Signal Length.

pNorm Pointer to the norm result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL1GetBufferSize_16s32s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.63.1.3 NppStatus nppsNormDiff_L1_16s64s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp64s ∗ *pNorm*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer L1 norm method on two vectors' difference, return value is 64-bit signed integer.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL1GetBufferSize_16s64s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.63.1.4 NppStatus nppsNormDiff_L1_32f (const Npp32f $*$ *pSrc1*, const Npp32f $*$ *pSrc2*, int *nLength*, Npp32f $*$ *pNorm*, Npp8u $*$ *pDeviceBuffer*)

32-bit float L1 norm method on two vectors' difference

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL1GetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.63.1.5 NppStatus nppsNormDiff_L1_32fc64f (const Npp32fc $*$ *pSrc1*, const Npp32fc $*$ *pSrc2*, int *nLength*, Npp64f $*$ *pNorm*, Npp8u $*$ *pDeviceBuffer*)

32-bit float complex L1 norm method on two vectors' difference, return value is 64-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL1GetBufferSize_32fc64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.63.1.6 NppStatus nppsNormDiff_L1_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float L1 norm method on two vectors' difference

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL1GetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.63.1.7 NppStatus nppsNormDiff_L1_64fc64f (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float complex L1 norm method on two vectors' difference, return value is 64-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL1GetBufferSize_64fc64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.63.1.8 NppStatus nppsNormDiffL1GetBufferSize_16s32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L1_16s32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.63.1.9 NppStatus nppsNormDiffL1GetBufferSize_16s32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L1_16s32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer*.

Returns:

NPP_SUCCESS

7.63.1.10 NppStatus nppsNormDiffL1GetBufferSize_16s64s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L1_16s64s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer*.

Returns:

NPP_SUCCESS

7.63.1.11 NppStatus nppsNormDiffL1GetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L1_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer*.

Returns:

NPP_SUCCESS

7.63.1.12 NppStatus nppsNormDiffL1GetBufferSize_32fc64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L1_32fc64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer*.

Returns:

NPP_SUCCESS

7.63.1.13 NppStatus nppsNormDiffL1GetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L1_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.63.1.14 NppStatus nppsNormDiffL1GetBufferSize_64fc64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L1_64fc64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.64 L2 Norm Diff

Functions

- NppStatus nppsNormDiffL2GetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L2_32f.

- NppStatus nppsNormDiff_L2_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float L2 norm method on two vectors' difference

- NppStatus nppsNormDiffL2GetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L2_64f.

- NppStatus nppsNormDiff_L2_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float L2 norm method on two vectors' difference

- NppStatus nppsNormDiffL2GetBufferSize_16s32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L2_16s32f.

- NppStatus nppsNormDiff_L2_16s32f (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp32f ∗pNorm, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer L2 norm method on two vectors' difference, return value is 32-bit float.

- NppStatus nppsNormDiffL2GetBufferSize_32fc64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L2_32fc64f.

- NppStatus nppsNormDiff_L2_32fc64f (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  32-bit float complex L2 norm method on two vectors' difference, return value is 64-bit float.

- NppStatus nppsNormDiffL2GetBufferSize_64fc64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L2_64fc64f.

- NppStatus nppsNormDiff_L2_64fc64f (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, int nLength, Npp64f ∗pNorm, Npp8u ∗pDeviceBuffer)

  64-bit float complex L2 norm method on two vectors' difference, return value is 64-bit float.

- NppStatus nppsNormDiffL2GetBufferSize_16s32s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L2_16s32s_Sfs.

- NppStatus nppsNormDiff_L2_16s32s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp32s ∗pNorm, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer L2 norm method on two vectors' difference, return value is 32-bit signed integer.

- NppStatus nppsNormDiffL2SqrGetBufferSize_16s64s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsNormDiff_L2Sqr_16s64s_Sfs.

- NppStatus nppsNormDiff_L2Sqr_16s64s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp64s ∗pNorm, int nScaleFactor, Npp8u ∗pDeviceBuffer)

16-bit signed short integer L2 Square norm method on two vectors' difference, return value is 64-bit signed integer.

7.64.1 Function Documentation

7.64.1.1 NppStatus nppsNormDiff_L2_16s32f (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer L2 norm method on two vectors' difference, return value is 32-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL2GetBufferSize_16s32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.64.1.2 NppStatus nppsNormDiff_L2_16s32s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp32s ∗ *pNorm*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer L2 norm method on two vectors' difference, return value is 32-bit signed integer.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL2GetBufferSize_16s32s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.64.1.3 NppStatus nppsNormDiff_L2_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, int *nLength*, Npp32f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

32-bit float L2 norm method on two vectors' difference

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL2GetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.64.1.4 NppStatus nppsNormDiff_L2_32fc64f (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

32-bit float complex L2 norm method on two vectors' difference, return value is 64-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL2GetBufferSize_32fc64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.64.1.5 NppStatus nppsNormDiff_L2_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float L2 norm method on two vectors' difference

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL2GetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.64.1.6 NppStatus nppsNormDiff_L2_64fc64f (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pNorm*, Npp8u ∗ *pDeviceBuffer*)

64-bit float complex L2 norm method on two vectors' difference, return value is 64-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL2GetBufferSize_64fc64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.64.1.7 NppStatus nppsNormDiff_L2Sqr_16s64s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp64s ∗ *pNorm*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer L2 Square norm method on two vectors' difference, return value is 64-bit signed integer.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pNorm Pointer to the norm result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsNormDiffL2SqrGetBufferSize_16s64s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.64.1.8 NppStatus nppsNormDiffL2GetBufferSize_16s32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L2_16s32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.64.1.9 NppStatus nppsNormDiffL2GetBufferSize_16s32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L2_16s32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.64.1.10 NppStatus nppsNormDiffL2GetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L2_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.64.1.11 NppStatus nppsNormDiffL2GetBufferSize_32fc64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L2_32fc64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.64.1.12 NppStatus nppsNormDiffL2GetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L2_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.64.1.13 NppStatus nppsNormDiffL2GetBufferSize_64fc64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L2_64fc64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.64.1.14 NppStatus nppsNormDiffL2SqrGetBufferSize_16s64s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsNormDiff_L2Sqr_16s64s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65 Dot Product

Functions

- NppStatus nppsDotProdGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_32f.

- NppStatus nppsDotProd_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, int nLength, Npp32f ∗pDp, Npp8u ∗pDeviceBuffer)

  32-bit float dot product method, return value is 32-bit float.

- NppStatus nppsDotProdGetBufferSize_32fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_32fc.

- NppStatus nppsDotProd_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, int nLength, Npp32fc ∗pDp, Npp8u ∗pDeviceBuffer)

  32-bit float complex dot product method, return value is 32-bit float complex.

- NppStatus nppsDotProdGetBufferSize_32f32fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_32f32fc.

- NppStatus nppsDotProd_32f32fc (const Npp32f ∗pSrc1, const Npp32fc ∗pSrc2, int nLength, Npp32fc ∗pDp, Npp8u ∗pDeviceBuffer)

  32-bit float and 32-bit float complex dot product method, return value is 32-bit float complex.

- NppStatus nppsDotProdGetBufferSize_32f64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_32f64f.

- NppStatus nppsDotProd_32f64f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, int nLength, Npp64f ∗pDp, Npp8u ∗pDeviceBuffer)

  32-bit float dot product method, return value is 64-bit float.

- NppStatus nppsDotProdGetBufferSize_32fc64fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_32fc64fc.

- NppStatus nppsDotProd_32fc64fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, int nLength, Npp64fc ∗pDp, Npp8u ∗pDeviceBuffer)

  32-bit float complex dot product method, return value is 64-bit float complex.

- NppStatus nppsDotProdGetBufferSize_32f32fc64fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_32f32fc64fc.

- NppStatus nppsDotProd_32f32fc64fc (const Npp32f ∗pSrc1, const Npp32fc ∗pSrc2, int nLength, Npp64fc ∗pDp, Npp8u ∗pDeviceBuffer)

  32-bit float and 32-bit float complex dot product method, return value is 64-bit float complex.

- NppStatus nppsDotProdGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_64f.

- NppStatus nppsDotProd_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, int nLength, Npp64f ∗pDp, Npp8u ∗pDeviceBuffer)

64-bit float dot product method, return value is 64-bit float.

- NppStatus nppsDotProdGetBufferSize_64fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_64fc.

- NppStatus nppsDotProd_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, int nLength, Npp64fc ∗pDp, Npp8u ∗pDeviceBuffer)

  64-bit float complex dot product method, return value is 64-bit float complex.

- NppStatus nppsDotProdGetBufferSize_64f64fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_64f64fc.

- NppStatus nppsDotProd_64f64fc (const Npp64f ∗pSrc1, const Npp64fc ∗pSrc2, int nLength, Npp64fc ∗pDp, Npp8u ∗pDeviceBuffer)

  64-bit float and 64-bit float complex dot product method, return value is 64-bit float complex.

- NppStatus nppsDotProdGetBufferSize_16s64s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16s64s.

- NppStatus nppsDotProd_16s64s (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp64s ∗pDp, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer dot product method, return value is 64-bit signed integer.

- NppStatus nppsDotProdGetBufferSize_16sc64sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16sc64sc.

- NppStatus nppsDotProd_16sc64sc (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp64sc ∗pDp, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer complex dot product method, return value is 64-bit signed integer complex.

- NppStatus nppsDotProdGetBufferSize_16s16sc64sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16s16sc64sc.

- NppStatus nppsDotProd_16s16sc64sc (const Npp16s ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp64sc ∗pDp, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer and 16-bit signed short integer short dot product method, return value is 64-bit signed integer complex.

- NppStatus nppsDotProdGetBufferSize_16s32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16s32f.

- NppStatus nppsDotProd_16s32f (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp32f ∗pDp, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer dot product method, return value is 32-bit float.

- NppStatus nppsDotProdGetBufferSize_16sc32fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16sc32fc.

- NppStatus nppsDotProd_16sc32fc (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp32fc ∗pDp, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer complex dot product method, return value is 32-bit float complex.

- NppStatus nppsDotProdGetBufferSize_16s16sc32fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16s16sc32fc.

- NppStatus nppsDotProd_16s16sc32fc (const Npp16s ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp32fc ∗pDp, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer and 16-bit signed short integer complex dot product method, return value is 32-bit float complex.

- NppStatus nppsDotProdGetBufferSize_16s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16s_Sfs.

- NppStatus nppsDotProd_16s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp16s ∗pDp, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer dot product method, return value is 16-bit signed short integer.

- NppStatus nppsDotProdGetBufferSize_16sc_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16sc_Sfs.

- NppStatus nppsDotProd_16sc_Sfs (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp16sc ∗pDp, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer complex dot product method, return value is 16-bit signed short integer complex.

- NppStatus nppsDotProdGetBufferSize_32s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_32s_Sfs.

- NppStatus nppsDotProd_32s_Sfs (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, int nLength, Npp32s ∗pDp, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  32-bit signed integer dot product method, return value is 32-bit signed integer.

- NppStatus nppsDotProdGetBufferSize_32sc_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_32sc_Sfs.

- NppStatus nppsDotProd_32sc_Sfs (const Npp32sc ∗pSrc1, const Npp32sc ∗pSrc2, int nLength, Npp32sc ∗pDp, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  32-bit signed integer complex dot product method, return value is 32-bit signed integer complex.

- NppStatus nppsDotProdGetBufferSize_16s32s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16s32s_Sfs.

- NppStatus nppsDotProd_16s32s_Sfs (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp32s ∗pDp, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer dot product method, return value is 32-bit signed integer.

- NppStatus nppsDotProdGetBufferSize_16s16sc32sc_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16s16sc32sc_Sfs.

- NppStatus nppsDotProd_16s16sc32sc_Sfs (const Npp16s ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp32sc ∗pDp, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer and 16-bit signed short integer complex dot product method, return value is 32-bit signed integer complex.

- NppStatus nppsDotProdGetBufferSize_16s32s32s_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16s32s32s_Sfs.

- NppStatus nppsDotProd_16s32s32s_Sfs (const Npp16s ∗pSrc1, const Npp32s ∗pSrc2, int nLength, Npp32s ∗pDp, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer and 32-bit signed integer dot product method, return value is 32-bit signed integer.

- NppStatus nppsDotProdGetBufferSize_16s16sc_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16s16sc_Sfs.

- NppStatus nppsDotProd_16s16sc_Sfs (const Npp16s ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp16sc ∗pDp, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer and 16-bit signed short integer complex dot product method, return value is 16-bit signed short integer complex.

- NppStatus nppsDotProdGetBufferSize_16sc32sc_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_16sc32sc_Sfs.

- NppStatus nppsDotProd_16sc32sc_Sfs (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp32sc ∗pDp, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer complex dot product method, return value is 32-bit signed integer complex.

- NppStatus nppsDotProdGetBufferSize_32s32sc_Sfs (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsDotProd_32s32sc_Sfs.

- NppStatus nppsDotProd_32s32sc_Sfs (const Npp32s ∗pSrc1, const Npp32sc ∗pSrc2, int nLength, Npp32sc ∗pDp, int nScaleFactor, Npp8u ∗pDeviceBuffer)

  32-bit signed short integer and 32-bit signed short integer complex dot product method, return value is 32-bit signed integer complex.

7.65.1 Function Documentation

7.65.1.1 NppStatus nppsDotProd_16s16sc32fc (const Npp16s ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, int *nLength*, Npp32fc ∗ *pDp*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer and 16-bit signed short integer complex dot product method, return value is 32-bit float complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16s16sc32fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.2 NppStatus nppsDotProd_16s16sc32sc_Sfs (const Npp16s ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, int *nLength*, Npp32sc ∗ *pDp*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer and 16-bit signed short integer complex dot product method, return value is 32-bit signed integer complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16s16sc32sc_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.3 NppStatus nppsDotProd_16s16sc64sc (const Npp16s ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, int *nLength*, Npp64sc ∗ *pDp*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer and 16-bit signed short integer short dot product method, return value is 64-bit signed integer complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16s16sc64sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.4 NppStatus nppsDotProd_16s16sc_Sfs (const Npp16s * *pSrc1*, const Npp16sc * *pSrc2*, int *nLength*, Npp16sc * *pDp*, int *nScaleFactor*, Npp8u * *pDeviceBuffer*)

16-bit signed short integer and 16-bit signed short integer complex dot product method, return value is 16-bit signed short integer complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16s16sc_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.5 NppStatus nppsDotProd_16s32f (const Npp16s * *pSrc1*, const Npp16s * *pSrc2*, int *nLength*, Npp32f * *pDp*, Npp8u * *pDeviceBuffer*)

16-bit signed short integer dot product method, return value is 32-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16s32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.6 NppStatus nppsDotProd_16s32s32s_Sfs (const Npp16s * *pSrc1*, const Npp32s * *pSrc2*, int *nLength*, Npp32s * *pDp*, int *nScaleFactor*, Npp8u * *pDeviceBuffer*)

16-bit signed short integer and 32-bit signed integer dot product method, return value is 32-bit signed integer.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16s32s32s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.7 NppStatus nppsDotProd_16s32s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp32s ∗ *pDp*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer dot product method, return value is 32-bit signed integer.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16s32s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.8 NppStatus nppsDotProd_16s64s (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp64s ∗ *pDp*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer dot product method, return value is 64-bit signed integer.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16s64s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.9 NppStatus nppsDotProd_16s_Sfs (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp16s ∗ *pDp*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer dot product method, return value is 16-bit signed short integer.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.10 NppStatus nppsDotProd_16sc32fc (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, int *nLength*, Npp32fc ∗ *pDp*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer complex dot product method, return value is 32-bit float complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16sc32fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.11 NppStatus nppsDotProd_16sc32sc_Sfs (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, int *nLength*, Npp32sc ∗ *pDp*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer complex dot product method, return value is 32-bit signed integer complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16sc32sc_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.12 NppStatus nppsDotProd_16sc64sc (const Npp16sc * *pSrc1*, const Npp16sc * *pSrc2*, int *nLength*, Npp64sc * *pDp*, Npp8u * *pDeviceBuffer*)

16-bit signed short integer complex dot product method, return value is 64-bit signed integer complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16sc64sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.13 NppStatus nppsDotProd_16sc_Sfs (const Npp16sc * *pSrc1*, const Npp16sc * *pSrc2*, int *nLength*, Npp16sc * *pDp*, int *nScaleFactor*, Npp8u * *pDeviceBuffer*)

16-bit signed short integer complex dot product method, return value is 16-bit signed short integer complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_16sc_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.14 NppStatus nppsDotProd_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, int *nLength*, Npp32f ∗ *pDp*, Npp8u ∗ *pDeviceBuffer*)

32-bit float dot product method, return value is 32-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.15 NppStatus nppsDotProd_32f32fc (const Npp32f ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, int *nLength*, Npp32fc ∗ *pDp*, Npp8u ∗ *pDeviceBuffer*)

32-bit float and 32-bit float complex dot product method, return value is 32-bit float complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_32f32fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.16 NppStatus nppsDotProd_32f32fc64fc (const Npp32f ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, int *nLength*, Npp64fc ∗ *pDp*, Npp8u ∗ *pDeviceBuffer*)

32-bit float and 32-bit float complex dot product method, return value is 64-bit float complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_32f32fc64fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.17 NppStatus nppsDotProd_32f64f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDp*, Npp8u ∗ *pDeviceBuffer*)

32-bit float dot product method, return value is 64-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_32f64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.18 NppStatus nppsDotProd_32fc (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, int *nLength*, Npp32fc ∗ *pDp*, Npp8u ∗ *pDeviceBuffer*)

32-bit float complex dot product method, return value is 32-bit float complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_32fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.19 NppStatus nppsDotProd_32fc64fc (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, int *nLength*, Npp64fc ∗ *pDp*, Npp8u ∗ *pDeviceBuffer*)

32-bit float complex dot product method, return value is 64-bit float complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_32fc64fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.20 NppStatus nppsDotProd_32s32sc_Sfs (const Npp32s ∗ *pSrc1*, const Npp32sc ∗ *pSrc2*, int *nLength*, Npp32sc ∗ *pDp*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

32-bit signed short integer and 32-bit signed short integer complex dot product method, return value is 32-bit signed integer complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_32s32sc_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.21 NppStatus nppsDotProd_32s_Sfs (const Npp32s ∗ *pSrc1*, const Npp32s ∗ *pSrc2*, int *nLength*, Npp32s ∗ *pDp*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

32-bit signed integer dot product method, return value is 32-bit signed integer.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_32s_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.22 NppStatus nppsDotProd_32sc_Sfs (const Npp32sc ∗ *pSrc1*, const Npp32sc ∗ *pSrc2*, int *nLength*, Npp32sc ∗ *pDp*, int *nScaleFactor*, Npp8u ∗ *pDeviceBuffer*)

32-bit signed integer complex dot product method, return value is 32-bit signed integer complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

nScaleFactor Integer Result Scaling.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_32sc_Sfs to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.23 NppStatus nppsDotProd_64f (const Npp64f $*$ *pSrc1*, const Npp64f $*$ *pSrc2*, int *nLength*, Npp64f $*$ *pDp*, Npp8u $*$ *pDeviceBuffer*)

64-bit float dot product method, return value is 64-bit float.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.24 NppStatus nppsDotProd_64f64fc (const Npp64f $*$ *pSrc1*, const Npp64fc $*$ *pSrc2*, int *nLength*, Npp64fc $*$ *pDp*, Npp8u $*$ *pDeviceBuffer*)

64-bit float and 64-bit float complex dot product method, return value is 64-bit float complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_64f64fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.25 NppStatus nppsDotProd_64fc (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, int *nLength*, Npp64fc ∗ *pDp*, Npp8u ∗ *pDeviceBuffer*)

64-bit float complex dot product method, return value is 64-bit float complex.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDp Pointer to the dot product result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsDotProdGetBufferSize_64fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.65.1.26 NppStatus nppsDotProdGetBufferSize_16s16sc32fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16s16sc32fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.27 NppStatus nppsDotProdGetBufferSize_16s16sc32sc_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16s16sc32sc_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.28 NppStatus nppsDotProdGetBufferSize_16s16sc64sc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16s16sc64sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.29 NppStatus nppsDotProdGetBufferSize_16s16sc_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16s16sc_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.30 NppStatus nppsDotProdGetBufferSize_16s32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16s32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.31 NppStatus nppsDotProdGetBufferSize_16s32s32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16s32s32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.32 NppStatus nppsDotProdGetBufferSize_16s32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16s32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.33 NppStatus nppsDotProdGetBufferSize_16s64s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16s64s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.34 NppStatus nppsDotProdGetBufferSize_16s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.35 NppStatus nppsDotProdGetBufferSize_16sc32fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16sc32fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.36 NppStatus nppsDotProdGetBufferSize_16sc32sc_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16sc32sc_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.37 NppStatus nppsDotProdGetBufferSize_16sc64sc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16sc64sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.38 NppStatus nppsDotProdGetBufferSize_16sc_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_16sc_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.39 NppStatus nppsDotProdGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.40 NppStatus nppsDotProdGetBufferSize_32f32fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_32f32fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.41 NppStatus nppsDotProdGetBufferSize_32f32fc64fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_32f32fc64fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.42 NppStatus nppsDotProdGetBufferSize_32f64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_32f64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.43 NppStatus nppsDotProdGetBufferSize_32fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_32fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.44 NppStatus nppsDotProdGetBufferSize_32fc64fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_32fc64fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.45 NppStatus nppsDotProdGetBufferSize_32s32sc_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_32s32sc_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.46 NppStatus nppsDotProdGetBufferSize_32s_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_32s_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.47 NppStatus nppsDotProdGetBufferSize_32sc_Sfs (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_32sc_Sfs.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.48 NppStatus nppsDotProdGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.49 NppStatus nppsDotProdGetBufferSize_64f64fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_64f64fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.65.1.50 NppStatus nppsDotProdGetBufferSize_64fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsDotProd_64fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.66 Count In Range

Functions

- NppStatus nppsCountInRangeGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsCountInRange_32s.

- NppStatus nppsCountInRange_32s (const Npp32s ∗pSrc, int nLength, int ∗pCounts, Npp32s nLowerBound, Npp32s nUpperBound, Npp8u ∗pDeviceBuffer)

  Computes the number of elements whose values fall into the specified range on a 32-bit signed integer array.

7.66.1 Function Documentation

7.66.1.1 NppStatus nppsCountInRange_32s (const Npp32s ∗ *pSrc*, int *nLength*, int ∗ *pCounts*, Npp32s *nLowerBound*, Npp32s *nUpperBound*, Npp8u ∗ *pDeviceBuffer*)

Computes the number of elements whose values fall into the specified range on a 32-bit signed integer array.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pCounts Pointer to the number of elements.

nLowerBound Lower bound of the specified range.

nUpperBound Upper bound of the specified range.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsCountInRangeGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.66.1.2 NppStatus nppsCountInRangeGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsCountInRange_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.67 Count Zero Crossings

Functions

- NppStatus nppsZeroCrossingGetBufferSize_16s32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsZeroCrossing_16s32f.
- NppStatus nppsZeroCrossing_16s32f (const Npp16s ∗pSrc, int nLength, Npp32f ∗pValZC, NppsZCType tZCType, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer zero crossing method, return value is 32-bit floating point.
- NppStatus nppsZeroCrossingGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsZeroCrossing_32f.
- NppStatus nppsZeroCrossing_32f (const Npp32f ∗pSrc, int nLength, Npp32f ∗pValZC, NppsZCType tZCType, Npp8u ∗pDeviceBuffer)

  32-bit floating-point zero crossing method, return value is 32-bit floating point.

7.67.1 Function Documentation

7.67.1.1 NppStatus nppsZeroCrossing_16s32f (const Npp16s ∗ *pSrc*, int *nLength*, Npp32f ∗ *pValZC*, NppsZCType *tZCType*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer zero crossing method, return value is 32-bit floating point.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pValZC Pointer to the output result.

tZCType Type of the zero crossing measure: nppZCR, nppZCXor or nppZCC.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsZeroCrossingGetBufferSize_16s32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.67.1.2 NppStatus nppsZeroCrossing_32f (const Npp32f ∗ *pSrc*, int *nLength*, Npp32f ∗ *pValZC*, NppsZCType *tZCType*, Npp8u ∗ *pDeviceBuffer*)

32-bit floating-point zero crossing method, return value is 32-bit floating point.

Parameters:

pSrc Source Signal Pointer.

nLength Signal Length.

pValZC Pointer to the output result.

tZCType Type of the zero crossing measure: nppZCR, nppZCXor or nppZCC.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsZeroCrossingGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.67.1.3 NppStatus nppsZeroCrossingGetBufferSize_16s32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsZeroCrossing_16s32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.67.1.4 NppStatus nppsZeroCrossingGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsZeroCrossing_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68 MaximumError

Primitives for computing the maximum error between two signals.

Functions

- NppStatus nppsMaximumError_8u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  8-bit unsigned char maximum method.

- NppStatus nppsMaximumError_8s (const Npp8s ∗pSrc1, const Npp8s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  8-bit signed char maximum method.

- NppStatus nppsMaximumError_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit unsigned short integer maximum method.

- NppStatus nppsMaximumError_16s (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer maximum method.

- NppStatus nppsMaximumError_16sc (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit unsigned short complex integer maximum method.

- NppStatus nppsMaximumError_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit unsigned short integer maximum method.

- NppStatus nppsMaximumError_32s (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit signed short integer maximum method.

- NppStatus nppsMaximumError_32sc (const Npp32sc ∗pSrc1, const Npp32sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit unsigned short complex integer maximum method.

- NppStatus nppsMaximumError_64s (const Npp64s ∗pSrc1, const Npp64s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit signed short integer maximum method.

- NppStatus nppsMaximumError_64sc (const Npp64sc ∗pSrc1, const Npp64sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit unsigned short complex integer maximum method.

- NppStatus nppsMaximumError_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit floating point maximum method.

- NppStatus nppsMaximumError_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit floating point complex maximum method.
- NppStatus nppsMaximumError_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit floating point maximum method.
- NppStatus nppsMaximumError_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit floating point complex maximum method.
- NppStatus nppsMaximumErrorGetBufferSize_8u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_8u.
- NppStatus nppsMaximumErrorGetBufferSize_8s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_8s.
- NppStatus nppsMaximumErrorGetBufferSize_16u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_16u.
- NppStatus nppsMaximumErrorGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_16s.
- NppStatus nppsMaximumErrorGetBufferSize_16sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_16sc.
- NppStatus nppsMaximumErrorGetBufferSize_32u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_32u.
- NppStatus nppsMaximumErrorGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_32s.
- NppStatus nppsMaximumErrorGetBufferSize_32sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_32sc.
- NppStatus nppsMaximumErrorGetBufferSize_64s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_64s.
- NppStatus nppsMaximumErrorGetBufferSize_64sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_64sc.
- NppStatus nppsMaximumErrorGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_32f.
- NppStatus nppsMaximumErrorGetBufferSize_32fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_32fc.
- NppStatus nppsMaximumErrorGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumError_64f.

- NppStatus nppsMaximumErrorGetBufferSize_64fc (int nLength, int ∗hpBufferSize)
  Device-buffer size (in bytes) for nppsMaximumError_64fc.

7.68.1 Detailed Description

Primitives for computing the maximum error between two signals.

Given two signals $pSrc1$ and $pSrc2$ both with length N, the maximum error is defined as the largest absolute difference between the corresponding elements of two signals.

If the signal is in complex format, the absolute value of the complex number is used.

7.68.2 Function Documentation

7.68.2.1 NppStatus nppsMaximumError_16s (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.2 NppStatus nppsMaximumError_16sc (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

16-bit unsigned short complex integer maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_16sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.3 NppStatus nppsMaximumError_16u (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

16-bit unsigned short integer maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_16u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.4 NppStatus nppsMaximumError_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit floating point maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.5 NppStatus nppsMaximumError_32fc (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit floating point complex maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_32fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.6 NppStatus nppsMaximumError_32s (const Npp32s * *pSrc1*, const Npp32s * *pSrc2*, int *nLength*, Npp64f * *pDst*, Npp8u * *pDeviceBuffer*)

32-bit signed short integer maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.7 NppStatus nppsMaximumError_32sc (const Npp32sc * *pSrc1*, const Npp32sc * *pSrc2*, int *nLength*, Npp64f * *pDst*, Npp8u * *pDeviceBuffer*)

32-bit unsigned short complex integer maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_32sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.8 NppStatus nppsMaximumError_32u (const Npp32u * *pSrc1*, const Npp32u * *pSrc2*, int *nLength*, Npp64f * *pDst*, Npp8u * *pDeviceBuffer*)

32-bit unsigned short integer maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_32u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.9 NppStatus nppsMaximumError_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit floating point maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.10 NppStatus nppsMaximumError_64fc (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit floating point complex maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_64fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.11 NppStatus nppsMaximumError_64s (const Npp64s ∗ *pSrc1*, const Npp64s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit signed short integer maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_64s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.12 NppStatus nppsMaximumError_64sc (const Npp64sc ∗ *pSrc1*, const Npp64sc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit unsigned short complex integer maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_64sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.13 NppStatus nppsMaximumError_8s (const Npp8s ∗ *pSrc1*, const Npp8s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

8-bit signed char maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_8s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.14 NppStatus nppsMaximumError_8u (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

8-bit unsigned char maximum method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumErrorGetBufferSize_8u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.68.2.15 NppStatus nppsMaximumErrorGetBufferSize_16s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.16 NppStatus nppsMaximumErrorGetBufferSize_16sc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_16sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.17 NppStatus nppsMaximumErrorGetBufferSize_16u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_16u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.18 NppStatus nppsMaximumErrorGetBufferSize_32f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.19 NppStatus nppsMaximumErrorGetBufferSize_32fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_32fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.20 NppStatus nppsMaximumErrorGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.21 NppStatus nppsMaximumErrorGetBufferSize_32sc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_32sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.22 NppStatus nppsMaximumErrorGetBufferSize_32u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_32u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.23 NppStatus nppsMaximumErrorGetBufferSize_64f (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.24 NppStatus nppsMaximumErrorGetBufferSize_64fc (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_64fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.25 NppStatus nppsMaximumErrorGetBufferSize_64s (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_64s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.26 NppStatus nppsMaximumErrorGetBufferSize_64sc (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_64sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.27 NppStatus nppsMaximumErrorGetBufferSize_8s (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_8s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.68.2.28 NppStatus nppsMaximumErrorGetBufferSize_8u (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumError_8u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69 AverageError

Primitives for computing the Average error between two signals.

Functions

- NppStatus nppsAverageError_8u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  8-bit unsigned char Average method.

- NppStatus nppsAverageError_8s (const Npp8s ∗pSrc1, const Npp8s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  8-bit signed char Average method.

- NppStatus nppsAverageError_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit unsigned short integer Average method.

- NppStatus nppsAverageError_16s (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer Average method.

- NppStatus nppsAverageError_16sc (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit unsigned short complex integer Average method.

- NppStatus nppsAverageError_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit unsigned short integer Average method.

- NppStatus nppsAverageError_32s (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit signed short integer Average method.

- NppStatus nppsAverageError_32sc (const Npp32sc ∗pSrc1, const Npp32sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit unsigned short complex integer Average method.

- NppStatus nppsAverageError_64s (const Npp64s ∗pSrc1, const Npp64s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit signed short integer Average method.

- NppStatus nppsAverageError_64sc (const Npp64sc ∗pSrc1, const Npp64sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit unsigned short complex integer Average method.

- NppStatus nppsAverageError_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit floating point Average method.

- NppStatus nppsAverageError_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit floating point complex Average method.

- NppStatus nppsAverageError_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit floating point Average method.

- NppStatus nppsAverageError_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit floating point complex Average method.

- NppStatus nppsAverageErrorGetBufferSize_8u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_8u.

- NppStatus nppsAverageErrorGetBufferSize_8s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_8s.

- NppStatus nppsAverageErrorGetBufferSize_16u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_16u.

- NppStatus nppsAverageErrorGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_16s.

- NppStatus nppsAverageErrorGetBufferSize_16sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_16sc.

- NppStatus nppsAverageErrorGetBufferSize_32u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_32u.

- NppStatus nppsAverageErrorGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_32s.

- NppStatus nppsAverageErrorGetBufferSize_32sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_32sc.

- NppStatus nppsAverageErrorGetBufferSize_64s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_64s.

- NppStatus nppsAverageErrorGetBufferSize_64sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_64sc.

- NppStatus nppsAverageErrorGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_32f.

- NppStatus nppsAverageErrorGetBufferSize_32fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_32fc.

- NppStatus nppsAverageErrorGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageError_64f.

- NppStatus nppsAverageErrorGetBufferSize_64fc (int nLength, int *hpBufferSize)
  Device-buffer size (in bytes) for nppsAverageError_64fc.

7.69.1 Detailed Description

Primitives for computing the Average error between two signals.

Given two signals $pSrc1$ and $pSrc2$ both with length N, the average error is defined as

$$AverageError = \frac{1}{N}\sum_{n=0}^{N-1} |pSrc1(n) - pSrc2(n)|$$

If the signal is in complex format, the absolute value of the complex number is used.

7.69.2 Function Documentation

7.69.2.1 NppStatus nppsAverageError_16s (const Npp16s * *pSrc1*, const Npp16s * *pSrc2*, int *nLength*, Npp64f * *pDst*, Npp8u * *pDeviceBuffer*)

16-bit signed short integer Average method.

Parameters:

pSrc1 Source Signal Pointer.
pSrc2 Source Signal Pointer.
nLength Signal Length.
pDst Pointer to the error result.
pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.2 NppStatus nppsAverageError_16sc (const Npp16sc * *pSrc1*, const Npp16sc * *pSrc2*, int *nLength*, Npp64f * *pDst*, Npp8u * *pDeviceBuffer*)

16-bit unsigned short complex integer Average method.

Parameters:

pSrc1 Source Signal Pointer.
pSrc2 Source Signal Pointer.
nLength Signal Length.
pDst Pointer to the error result.
pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_16sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.3 NppStatus nppsAverageError_16u (const Npp16u * *pSrc1*, const Npp16u * *pSrc2*, int *nLength*, Npp64f * *pDst*, Npp8u * *pDeviceBuffer*)

16-bit unsigned short integer Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_16u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.4 NppStatus nppsAverageError_32f (const Npp32f * *pSrc1*, const Npp32f * *pSrc2*, int *nLength*, Npp64f * *pDst*, Npp8u * *pDeviceBuffer*)

32-bit floating point Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.5 NppStatus nppsAverageError_32fc (const Npp32fc * *pSrc1*, const Npp32fc * *pSrc2*, int *nLength*, Npp64f * *pDst*, Npp8u * *pDeviceBuffer*)

32-bit floating point complex Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_32fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.6 NppStatus nppsAverageError_32s (const Npp32s * *pSrc1*, const Npp32s * *pSrc2*, int *nLength*, Npp64f * *pDst*, Npp8u * *pDeviceBuffer*)

32-bit signed short integer Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.7 NppStatus nppsAverageError_32sc (const Npp32sc * *pSrc1*, const Npp32sc * *pSrc2*, int *nLength*, Npp64f * *pDst*, Npp8u * *pDeviceBuffer*)

32-bit unsigned short complex integer Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_32sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.8 NppStatus nppsAverageError_32u (const Npp32u * *pSrc1*, const Npp32u * *pSrc2*, int *nLength*, Npp64f * *pDst*, Npp8u * *pDeviceBuffer*)

32-bit unsigned short integer Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_32u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.9 NppStatus nppsAverageError_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit floating point Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.10 NppStatus nppsAverageError_64fc (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit floating point complex Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_64fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.11 NppStatus nppsAverageError_64s (const Npp64s ∗ *pSrc1*, const Npp64s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit signed short integer Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_64s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.12 NppStatus nppsAverageError_64sc (const Npp64sc ∗ *pSrc1*, const Npp64sc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit unsigned short complex integer Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_64sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.13 NppStatus nppsAverageError_8s (const Npp8s ∗ *pSrc1*, const Npp8s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

8-bit signed char Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_8s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.14 NppStatus nppsAverageError_8u (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

8-bit unsigned char Average method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageErrorGetBufferSize_8u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.69.2.15 NppStatus nppsAverageErrorGetBufferSize_16s (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.16 NppStatus nppsAverageErrorGetBufferSize_16sc (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_16sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.17 NppStatus nppsAverageErrorGetBufferSize_16u (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_16u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.18 NppStatus nppsAverageErrorGetBufferSize_32f (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.19 NppStatus nppsAverageErrorGetBufferSize_32fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_32fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.20 NppStatus nppsAverageErrorGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.21 NppStatus nppsAverageErrorGetBufferSize_32sc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_32sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.22 NppStatus nppsAverageErrorGetBufferSize_32u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_32u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.23 NppStatus nppsAverageErrorGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_64f.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.24 NppStatus nppsAverageErrorGetBufferSize_64fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_64fc.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.25 NppStatus nppsAverageErrorGetBufferSize_64s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_64s.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.26 NppStatus nppsAverageErrorGetBufferSize_64sc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_64sc.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.27 NppStatus nppsAverageErrorGetBufferSize_8s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_8s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.69.2.28 NppStatus nppsAverageErrorGetBufferSize_8u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageError_8u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70 MaximumRelativeError

Primitives for computing the MaximumRelative error between two signals.

Functions

- NppStatus nppsMaximumRelativeError_8u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  8-bit unsigned char MaximumRelative method.

- NppStatus nppsMaximumRelativeError_8s (const Npp8s ∗pSrc1, const Npp8s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  8-bit signed char MaximumRelative method.

- NppStatus nppsMaximumRelativeError_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit unsigned short integer MaximumRelative method.

- NppStatus nppsMaximumRelativeError_16s (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer MaximumRelative method.

- NppStatus nppsMaximumRelativeError_16sc (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit unsigned short complex integer MaximumRelative method.

- NppStatus nppsMaximumRelativeError_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit unsigned short integer MaximumRelative method.

- NppStatus nppsMaximumRelativeError_32s (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit signed short integer MaximumRelative method.

- NppStatus nppsMaximumRelativeError_32sc (const Npp32sc ∗pSrc1, const Npp32sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit unsigned short complex integer MaximumRelative method.

- NppStatus nppsMaximumRelativeError_64s (const Npp64s ∗pSrc1, const Npp64s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit signed short integer MaximumRelative method.

- NppStatus nppsMaximumRelativeError_64sc (const Npp64sc ∗pSrc1, const Npp64sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit unsigned short complex integer MaximumRelative method.

- NppStatus nppsMaximumRelativeError_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit floating point MaximumRelative method.

- NppStatus nppsMaximumRelativeError_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit floating point complex MaximumRelative method.

- NppStatus nppsMaximumRelativeError_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit floating point MaximumRelative method.

- NppStatus nppsMaximumRelativeError_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit floating point complex MaximumRelative method.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_8u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_8u.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_8s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_8s.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_16u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_16u.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_16s.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_16sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_16sc.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_32u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_32u.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_32s.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_32sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_32sc.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_64s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_64s.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_64sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_64sc.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_32f.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_32fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_32fc.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_64f.

- NppStatus nppsMaximumRelativeErrorGetBufferSize_64fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsMaximumRelativeError_64fc.

7.70.1 Detailed Description

Primitives for computing the MaximumRelative error between two signals.

Given two signals $pSrc1$ and $pSrc2$ both with length N, the maximum relative error is defined as

$$MaximumRelativeError = max \frac{|pSrc1(n) - pSrc2(n)|}{max(|pSrc1(n)|, |pSrc2(n)|)}$$

If the signal is in complex format, the absolute value of the complex number is used.

7.70.2 Function Documentation

7.70.2.1 NppStatus nppsMaximumRelativeError_16s (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.2 NppStatus nppsMaximumRelativeError_16sc (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

16-bit unsigned short complex integer MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_16sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.3 NppStatus nppsMaximumRelativeError_16u (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

16-bit unsigned short integer MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_16u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.4 NppStatus nppsMaximumRelativeError_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit floating point MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.5 NppStatus nppsMaximumRelativeError_32fc (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit floating point complex MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_32fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.6 NppStatus nppsMaximumRelativeError_32s (const Npp32s ∗ *pSrc1*, const Npp32s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit signed short integer MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.7 NppStatus nppsMaximumRelativeError_32sc (const Npp32sc ∗ *pSrc1*, const Npp32sc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit unsigned short complex integer MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_32sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.8 NppStatus nppsMaximumRelativeError_32u (const Npp32u ∗ *pSrc1*, const Npp32u ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit unsigned short integer MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_32u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.9 NppStatus nppsMaximumRelativeError_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit floating point MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.10 NppStatus nppsMaximumRelativeError_64fc (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit floating point complex MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_64fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.11 NppStatus nppsMaximumRelativeError_64s (const Npp64s ∗ *pSrc1*, const Npp64s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit signed short integer MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_64s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.12 NppStatus nppsMaximumRelativeError_64sc (const Npp64sc ∗ *pSrc1*, const Npp64sc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit unsigned short complex integer MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_64sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.13 NppStatus nppsMaximumRelativeError_8s (const Npp8s ∗ *pSrc1*, const Npp8s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

8-bit signed char MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_8s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.14 NppStatus nppsMaximumRelativeError_8u (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

8-bit unsigned char MaximumRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsMaximumRelativeErrorGetBufferSize_8u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.70.2.15 NppStatus nppsMaximumRelativeErrorGetBufferSize_16s (int *nLength*, int $*$ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.16 NppStatus nppsMaximumRelativeErrorGetBufferSize_16sc (int *nLength*, int $*$ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_16sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.17 NppStatus nppsMaximumRelativeErrorGetBufferSize_16u (int *nLength*, int $*$ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_16u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.18 NppStatus nppsMaximumRelativeErrorGetBufferSize_32f (int *nLength*, int $*$ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.19 NppStatus nppsMaximumRelativeErrorGetBufferSize_32fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_32fc.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.20 NppStatus nppsMaximumRelativeErrorGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_32s.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.21 NppStatus nppsMaximumRelativeErrorGetBufferSize_32sc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_32sc.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.22 NppStatus nppsMaximumRelativeErrorGetBufferSize_32u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_32u.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.23 NppStatus nppsMaximumRelativeErrorGetBufferSize_64f (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_64f.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.24 NppStatus nppsMaximumRelativeErrorGetBufferSize_64fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_64fc.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.25 NppStatus nppsMaximumRelativeErrorGetBufferSize_64s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_64s.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.26 NppStatus nppsMaximumRelativeErrorGetBufferSize_64sc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_64sc.

Parameters:

nLength Signal Length.
hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.27 NppStatus nppsMaximumRelativeErrorGetBufferSize_8s (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_8s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.70.2.28 NppStatus nppsMaximumRelativeErrorGetBufferSize_8u (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsMaximumRelativeError_8u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71 AverageRelativeError

Primitives for computing the AverageRelative error between two signals.

Functions

- NppStatus nppsAverageRelativeError_8u (const Npp8u ∗pSrc1, const Npp8u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  8-bit unsigned char AverageRelative method.

- NppStatus nppsAverageRelativeError_8s (const Npp8s ∗pSrc1, const Npp8s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  8-bit signed char AverageRelative method.

- NppStatus nppsAverageRelativeError_16u (const Npp16u ∗pSrc1, const Npp16u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit unsigned short integer AverageRelative method.

- NppStatus nppsAverageRelativeError_16s (const Npp16s ∗pSrc1, const Npp16s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit signed short integer AverageRelative method.

- NppStatus nppsAverageRelativeError_16sc (const Npp16sc ∗pSrc1, const Npp16sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  16-bit unsigned short complex integer AverageRelative method.

- NppStatus nppsAverageRelativeError_32u (const Npp32u ∗pSrc1, const Npp32u ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit unsigned short integer AverageRelative method.

- NppStatus nppsAverageRelativeError_32s (const Npp32s ∗pSrc1, const Npp32s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit signed short integer AverageRelative method.

- NppStatus nppsAverageRelativeError_32sc (const Npp32sc ∗pSrc1, const Npp32sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit unsigned short complex integer AverageRelative method.

- NppStatus nppsAverageRelativeError_64s (const Npp64s ∗pSrc1, const Npp64s ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit signed short integer AverageRelative method.

- NppStatus nppsAverageRelativeError_64sc (const Npp64sc ∗pSrc1, const Npp64sc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit unsigned short complex integer AverageRelative method.

- NppStatus nppsAverageRelativeError_32f (const Npp32f ∗pSrc1, const Npp32f ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit floating point AverageRelative method.

- NppStatus nppsAverageRelativeError_32fc (const Npp32fc ∗pSrc1, const Npp32fc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  32-bit floating point complex AverageRelative method.

- NppStatus nppsAverageRelativeError_64f (const Npp64f ∗pSrc1, const Npp64f ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit floating point AverageRelative method.

- NppStatus nppsAverageRelativeError_64fc (const Npp64fc ∗pSrc1, const Npp64fc ∗pSrc2, int nLength, Npp64f ∗pDst, Npp8u ∗pDeviceBuffer)

  64-bit floating point complex AverageRelative method.

- NppStatus nppsAverageRelativeErrorGetBufferSize_8u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_8u.

- NppStatus nppsAverageRelativeErrorGetBufferSize_8s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_8s.

- NppStatus nppsAverageRelativeErrorGetBufferSize_16u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_16u.

- NppStatus nppsAverageRelativeErrorGetBufferSize_16s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_16s.

- NppStatus nppsAverageRelativeErrorGetBufferSize_16sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_16sc.

- NppStatus nppsAverageRelativeErrorGetBufferSize_32u (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_32u.

- NppStatus nppsAverageRelativeErrorGetBufferSize_32s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_32s.

- NppStatus nppsAverageRelativeErrorGetBufferSize_32sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_32sc.

- NppStatus nppsAverageRelativeErrorGetBufferSize_64s (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_64s.

- NppStatus nppsAverageRelativeErrorGetBufferSize_64sc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_64sc.

- NppStatus nppsAverageRelativeErrorGetBufferSize_32f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_32f.

- NppStatus nppsAverageRelativeErrorGetBufferSize_32fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_32fc.

- NppStatus nppsAverageRelativeErrorGetBufferSize_64f (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_64f.

- NppStatus nppsAverageRelativeErrorGetBufferSize_64fc (int nLength, int ∗hpBufferSize)

  Device-buffer size (in bytes) for nppsAverageRelativeError_64fc.

7.71.1 Detailed Description

Primitives for computing the AverageRelative error between two signals.

Given two signals $pSrc1$ and $pSrc2$ both with length N, the average relative error is defined as

$$AverageRelativeError = \frac{1}{N}\sum_{n=0}^{N-1}\frac{|pSrc1(n) - pSrc2(n)|}{max(|pSrc1(n)|, |pSrc2(n)|)}$$

If the signal is in complex format, the absolute value of the complex number is used.

7.71.2 Function Documentation

7.71.2.1 NppStatus nppsAverageRelativeError_16s (const Npp16s ∗ *pSrc1*, const Npp16s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

16-bit signed short integer AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_16s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.2 NppStatus nppsAverageRelativeError_16sc (const Npp16sc ∗ *pSrc1*, const Npp16sc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

16-bit unsigned short complex integer AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_16sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.3 NppStatus nppsAverageRelativeError_16u (const Npp16u ∗ *pSrc1*, const Npp16u ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

16-bit unsigned short integer AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_16u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.4 NppStatus nppsAverageRelativeError_32f (const Npp32f ∗ *pSrc1*, const Npp32f ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit floating point AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_32f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.5 NppStatus nppsAverageRelativeError_32fc (const Npp32fc ∗ *pSrc1*, const Npp32fc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit floating point complex AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_32fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.6 NppStatus nppsAverageRelativeError_32s (const Npp32s ∗ *pSrc1*, const Npp32s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit signed short integer AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_32s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.7 NppStatus nppsAverageRelativeError_32sc (const Npp32sc ∗ *pSrc1*, const Npp32sc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit unsigned short complex integer AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_32sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.8 NppStatus nppsAverageRelativeError_32u (const Npp32u ∗ *pSrc1*, const Npp32u ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

32-bit unsigned short integer AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_32u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.9 NppStatus nppsAverageRelativeError_64f (const Npp64f ∗ *pSrc1*, const Npp64f ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit floating point AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_64f to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.10 NppStatus nppsAverageRelativeError_64fc (const Npp64fc ∗ *pSrc1*, const Npp64fc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit floating point complex AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_64fc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.11 NppStatus nppsAverageRelativeError_64s (const Npp64s ∗ *pSrc1*, const Npp64s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit signed short integer AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_64s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.12 NppStatus nppsAverageRelativeError_64sc (const Npp64sc ∗ *pSrc1*, const Npp64sc ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

64-bit unsigned short complex integer AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_64sc to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.13 NppStatus nppsAverageRelativeError_8s (const Npp8s ∗ *pSrc1*, const Npp8s ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

8-bit signed char AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_8s to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.14 NppStatus nppsAverageRelativeError_8u (const Npp8u ∗ *pSrc1*, const Npp8u ∗ *pSrc2*, int *nLength*, Npp64f ∗ *pDst*, Npp8u ∗ *pDeviceBuffer*)

8-bit unsigned char AverageRelative method.

Parameters:

pSrc1 Source Signal Pointer.

pSrc2 Source Signal Pointer.

nLength Signal Length.

pDst Pointer to the error result.

pDeviceBuffer Pointer to the required device memory allocation, Scratch Buffer and Host Pointer. Use nppsAverageRelativeErrorGetBufferSize_8u to determine the minium number of bytes required.

Returns:

Signal Data Related Error Codes, Length Related Error Codes.

7.71.2.15 NppStatus nppsAverageRelativeErrorGetBufferSize_16s (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_16s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.16 NppStatus nppsAverageRelativeErrorGetBufferSize_16sc (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_16sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.17 NppStatus nppsAverageRelativeErrorGetBufferSize_16u (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_16u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.18 NppStatus nppsAverageRelativeErrorGetBufferSize_32f (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_32f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.19 NppStatus nppsAverageRelativeErrorGetBufferSize_32fc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_32fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.20 NppStatus nppsAverageRelativeErrorGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_32s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.21 NppStatus nppsAverageRelativeErrorGetBufferSize_32sc (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_32sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.22 NppStatus nppsAverageRelativeErrorGetBufferSize_32u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_32u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.23 NppStatus nppsAverageRelativeErrorGetBufferSize_64f (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_64f.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.24 NppStatus nppsAverageRelativeErrorGetBufferSize_64fc (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_64fc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.25 NppStatus nppsAverageRelativeErrorGetBufferSize_64s (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_64s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.26 NppStatus nppsAverageRelativeErrorGetBufferSize_64sc (int *nLength*, int * *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_64sc.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.27 NppStatus nppsAverageRelativeErrorGetBufferSize_8s (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_8s.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.71.2.28 NppStatus nppsAverageRelativeErrorGetBufferSize_8u (int *nLength*, int ∗ *hpBufferSize*)

Device-buffer size (in bytes) for nppsAverageRelativeError_8u.

Parameters:

nLength Signal Length.

hpBufferSize Required buffer size. Important: hpBufferSize is a *host pointer.*

Returns:

NPP_SUCCESS

7.72 Filtering Functions

Functions that provide functionality of generating output signal based on the input signal like signal integral, etc.

Modules

- Integral

  Compute the indefinite interal of a given signal.

7.72.1 Detailed Description

Functions that provide functionality of generating output signal based on the input signal like signal integral, etc.

7.73 Integral

Compute the indefinite interal of a given signal.

Functions

- NppStatus nppsIntegralGetBufferSize_32s (int nLength, int ∗hpBufferSize)
- NppStatus nppsIntegral_32s (const Npp32s ∗pSrc, Npp32s ∗pDst, int nLength, Npp8u ∗pDeviceBuffer)

7.73.1 Detailed Description

Compute the indefinite interal of a given signal.

The i-th element is computed to be

$$s'_i = \sum_0^i s_j$$

7.73.2 Function Documentation

7.73.2.1 NppStatus nppsIntegral_32s (const Npp32s ∗ *pSrc*, Npp32s ∗ *pDst*, int *nLength*, Npp8u ∗ *pDeviceBuffer*)

7.73.2.2 NppStatus nppsIntegralGetBufferSize_32s (int *nLength*, int ∗ *hpBufferSize*)

Chapter 8

Data Structure Documentation

8.1 NPP_ALIGN_16 Struct Reference

Complex Number This struct represents a long long complex number.

```
#include <nppdefs.h>
```

Data Fields

- Npp64s re

  Real part.

- Npp64s im

  Imaginary part.

- Npp64f re

  Real part.

- Npp64f im

  Imaginary part.

8.1.1 Detailed Description

Complex Number This struct represents a long long complex number.

Complex Number This struct represents a double floating-point complex number.

8.1.2 Field Documentation

8.1.2.1 Npp64f NPP_ALIGN_16::im

Imaginary part.

8.1.2.2 Npp64s NPP_ALIGN_16::im

Imaginary part.

8.1.2.3 Npp64f NPP_ALIGN_16::re

Real part.

8.1.2.4 Npp64s NPP_ALIGN_16::re

Real part.

The documentation for this struct was generated from the following file:

- C:/src/sw/rel/gpgpu/toolkit/r9.0/NPP/npp/include/nppdefs.h

8.2 NPP_ALIGN_8 Struct Reference

Complex Number This struct represents an unsigned int complex number.

`#include <nppdefs.h>`

Data Fields

- Npp32u re

  Real part.
- Npp32u im

  Imaginary part.
- Npp32s re

  Real part.
- Npp32s im

  Imaginary part.
- Npp32f re

  Real part.
- Npp32f im

  Imaginary part.

8.2.1 Detailed Description

Complex Number This struct represents an unsigned int complex number.

Complex Number This struct represents a single floating-point complex number.

Complex Number This struct represents a signed int complex number.

8.2.2 Field Documentation

8.2.2.1 Npp32f NPP_ALIGN_8::im

Imaginary part.

8.2.2.2 Npp32s NPP_ALIGN_8::im

Imaginary part.

8.2.2.3 Npp32u NPP_ALIGN_8::im

Imaginary part.

8.2.2.4 Npp32f NPP_ALIGN_8::re

Real part.

8.2.2.5 Npp32s NPP_ALIGN_8::re

Real part.

8.2.2.6 Npp32u NPP_ALIGN_8::re

Real part.

The documentation for this struct was generated from the following file:

- C:/src/sw/rel/gpgpu/toolkit/r9.0/NPP/npp/include/nppdefs.h

8.3 NppiHaarBuffer Struct Reference

```
#include <nppdefs.h>
```

Data Fields

- int haarBufferSize
  size of the buffer
- Npp32s ∗ haarBuffer
  buffer

8.3.1 Field Documentation

8.3.1.1 Npp32s∗ NppiHaarBuffer::haarBuffer

buffer

8.3.1.2 int NppiHaarBuffer::haarBufferSize

size of the buffer

The documentation for this struct was generated from the following file:

- C:/src/sw/rel/gpgpu/toolkit/r9.0/NPP/npp/include/nppdefs.h

8.4 NppiHaarClassifier_32f Struct Reference

```
#include <nppdefs.h>
```

Data Fields

- int numClassifiers

  number of classifiers

- Npp32s ∗ classifiers

  packed classifier data 40 bytes each

- size_t classifierStep
- NppiSize classifierSize
- Npp32s ∗ counterDevice

8.4.1 Field Documentation

8.4.1.1 Npp32s∗ NppiHaarClassifier_32f::classifiers

packed classifier data 40 bytes each

8.4.1.2 NppiSize NppiHaarClassifier_32f::classifierSize

8.4.1.3 size_t NppiHaarClassifier_32f::classifierStep

8.4.1.4 Npp32s∗ NppiHaarClassifier_32f::counterDevice

8.4.1.5 int NppiHaarClassifier_32f::numClassifiers

number of classifiers

The documentation for this struct was generated from the following file:

- C:/src/sw/rel/gpgpu/toolkit/r9.0/NPP/npp/include/nppdefs.h

8.5 NppiHOGConfig Struct Reference

The NppiHOGConfig structure defines the configuration parameters for the HOG descriptor:.

```
#include <nppdefs.h>
```

Data Fields

- int cellSize

  square cell size (pixels).

- int histogramBlockSize

  square histogram block size (pixels).

- int nHistogramBins

  required number of histogram bins.

- NppiSize detectionWindowSize

  detection window size (pixels).

8.5.1 Detailed Description

The NppiHOGConfig structure defines the configuration parameters for the HOG descriptor:.

8.5.2 Field Documentation

8.5.2.1 int NppiHOGConfig::cellSize

square cell size (pixels).

8.5.2.2 NppiSize NppiHOGConfig::detectionWindowSize

detection window size (pixels).

8.5.2.3 int NppiHOGConfig::histogramBlockSize

square histogram block size (pixels).

8.5.2.4 int NppiHOGConfig::nHistogramBins

required number of histogram bins.

The documentation for this struct was generated from the following file:

- C:/src/sw/rel/gpgpu/toolkit/r9.0/NPP/npp/include/nppdefs.h

8.6 NppiPoint Struct Reference

2D Point

```
#include <nppdefs.h>
```

Data Fields

- int x

  x-coordinate.

- int y

  y-coordinate.

8.6.1 Detailed Description

2D Point

8.6.2 Field Documentation

8.6.2.1 int NppiPoint::x

x-coordinate.

8.6.2.2 int NppiPoint::y

y-coordinate.

The documentation for this struct was generated from the following file:

- C:/src/sw/rel/gpgpu/toolkit/r9.0/NPP/npp/include/nppdefs.h

8.7 NppiRect Struct Reference

2D Rectangle This struct contains position and size information of a rectangle in two space.

`#include <nppdefs.h>`

Data Fields

- int x

  x-coordinate of upper left corner (lowest memory address).
- int y

  y-coordinate of upper left corner (lowest memory address).
- int width

  Rectangle width.
- int height

  Rectangle height.

8.7.1 Detailed Description

2D Rectangle This struct contains position and size information of a rectangle in two space.

The rectangle's position is usually signified by the coordinate of its upper-left corner.

8.7.2 Field Documentation

8.7.2.1 int NppiRect::height

Rectangle height.

8.7.2.2 int NppiRect::width

Rectangle width.

8.7.2.3 int NppiRect::x

x-coordinate of upper left corner (lowest memory address).

8.7.2.4 int NppiRect::y

y-coordinate of upper left corner (lowest memory address).

The documentation for this struct was generated from the following file:

- C:/src/sw/rel/gpgpu/toolkit/r9.0/NPP/npp/include/nppdefs.h

8.8 NppiSize Struct Reference

2D Size This struct typically represents the size of a a rectangular region in two space.

```
#include <nppdefs.h>
```

Data Fields

- int width
  Rectangle width.
- int height
  Rectangle height.

8.8.1 Detailed Description

2D Size This struct typically represents the size of a a rectangular region in two space.

8.8.2 Field Documentation

8.8.2.1 int NppiSize::height

Rectangle height.

8.8.2.2 int NppiSize::width

Rectangle width.

The documentation for this struct was generated from the following file:

- C:/src/sw/rel/gpgpu/toolkit/r9.0/NPP/npp/include/nppdefs.h

8.9 NppLibraryVersion Struct Reference

```
#include <nppdefs.h>
```

Data Fields

- int major

  Major version number.

- int minor

  Minor version number.

- int build

  Build number.

8.9.1 Field Documentation

8.9.1.1 int NppLibraryVersion::build

Build number.

This reflects the nightly build this release was made from.

8.9.1.2 int NppLibraryVersion::major

Major version number.

8.9.1.3 int NppLibraryVersion::minor

Minor version number.

The documentation for this struct was generated from the following file:

- C:/src/sw/rel/gpgpu/toolkit/r9.0/NPP/npp/include/nppdefs.h

8.10 NppPointPolar Struct Reference

2D Polar Point

```
#include <nppdefs.h>
```

Data Fields

- Npp32f rho
- Npp32f theta

8.10.1 Detailed Description

2D Polar Point

8.10.2 Field Documentation

8.10.2.1 Npp32f NppPointPolar::rho

8.10.2.2 Npp32f NppPointPolar::theta

The documentation for this struct was generated from the following file:

- C:/src/sw/rel/gpgpu/toolkit/r9.0/NPP/npp/include/nppdefs.h

Index