vDSP Two-Dimensional Fast Fourier Transforms Reference

Performance > Carbon

2008-11-19

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Contents

vDSP Two-Dimensional Fast Fourier Transforms Reference

Framework: Accelerate/vecLib

Declared in vDSP.h

Overview

This document describes the C API for performing two-dimensional Fast Fourier Transforms on an input signal. It also describes the `FFTSetup` structure which you pass as a handle to the FFT functions.

Functions by Task

Computing In-Place Complex FFTs

`vDSP_fft2d_zip` (page 7)
Computes an in-place single-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

`vDSP_fft2d_zipD` (page 8)
Computes an in-place double-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

`vDSP_fft2d_zipt` (page 10)
Computes an in-place single-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

`vDSP_fft2d_ziptD` (page 11)
Computes an in-place double-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

Computing Out-of-Place Complex FFTs

`vDSP_fft2d_zop` (page 13)
Computes an out-of-place single-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

`vDSP_fft2d_zopD` (page 14)
Computes an out-of-place double-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

`vDSP_fft2d_zopt` (page 15)
: Computes an out-of-place single-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

`vDSP_fft2d_zoptD` (page 17)
: Computes an out-of-place double-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

Computing In-Place Real FFTs

`vDSP_fft2d_zrip` (page 18)
: Computes an in-place single-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

`vDSP_fft2d_zripD` (page 20)
: Computes an in-place double-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

`vDSP_fft2d_zript` (page 22)
: Computes an in-place single-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

`vDSP_fft2d_zriptD` (page 23)
: Computes an in-place double-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

Computing Out-of-Place Real FFTs

`vDSP_fft2d_zrop` (page 25)
: Computes an out-of-place single-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

`vDSP_fft2d_zropD` (page 27)
: Computes an out-of-place double-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

`vDSP_fft2d_zropt` (page 28)
: Computes an out-of-place single-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

`vDSP_fft2d_zroptD` (page 30)
: Computes an out-of-place double-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

Functions

vDSP_fft2d_zip

Computes an in-place single-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

```
void vDSP_fft2d_zip (FFTSetup setup,
DSPSplitComplex * ioData,
vDSP_Stride strideInRow,
vDSP_Stride strideInCol,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection direction);
```

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters `log2nInCol` and `log2nInRow` of the transform function.

ioData

A complex vector input.

strideInRow

Specifies a stride across each row of the matrix `signal`. Specifying 1 for `strideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

strideInCol

Specifies a column stride for the matrix, and should generally be allowed to default unless the matrix is a submatrix. Parameter `strideInCol` can be defaulted by specifying 0. The default column stride equals the row stride multiplied by the column count. Thus, if `strideInRow` is 1 and `strideInCol` is 0, every element of the input /output matrix is processed. If `strideInRow` is 2 and `strideInCol` is 0, every other element of each row is processed.

If not 0, parameter `strideInCol` represents the distance between each row of the matrix. If parameter `strideInCol` is 1024, for instance, complex element 512 of the matrix equates to element (1,0), element 1024 equates to element (2,0), and so forth.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 2 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for parameter `log2nInCol` and 6 for parameter `log2nInRow`. `log2nInRow` must be between 2 and 10, inclusive.

direction

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `direction`.

Discussion

This performs the operation

If F = 1 $C_{nm} = \text{FDFT2D}(C_{nm})$ n = {0, N-1} and m = {0, M-1}

If F = -1 $C_{nm} = \text{IDFT2D}(C_{nm}) \cdot MN$ n = {0, N-1} and m = {0, M-1}

$$\text{FDFT2D}(X_{nm}) = \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(-j2\pi pn)/N} \cdot e^{(-j2\pi qm)/M}$$

$$\text{IDFT2D}(X_{nm}) = \frac{1}{MN} \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(j2\pi pn)/N} \cdot e^{(j2\pi qm)/M}$$

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zipD

Computes an in-place double-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

```
void vDSP_fft2d_zipD (FFTSetupD setup,
DSPDoubleSplitComplex * ioData,
vDSP_Stride strideInRow,
vDSP_Stride strideInCol,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection direction);
```

Parameters

`setup`

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters `log2nInCol` and `log2nInRow` of the transform function.

`ioData`

A complex vector input.

`strideInRow`

Specifies a stride across each row of the matrix `signal`. Specifying 1 for `strideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

strideInCol

Specifies a column stride for the matrix, and should generally be allowed to default unless the matrix is a submatrix. Parameter `strideInCol` can be defaulted by specifying 0. The default column stride equals the row stride multiplied by the column count. Thus, if `strideInRow` is 1 and `strideInCol` is 0, every element of the input /output matrix is processed. If `strideInRow` is 2 and `strideInCol` is 0, every other element of each row is processed.

If not 0, parameter `strideInCol` represents the distance between each row of the matrix. If parameter `strideInCol` is 1024, for instance, complex element 512 of the matrix equates to element (1,0), element 1024 equates to element (2,0), and so forth.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 2 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for parameter `log2nInCol` and 6 for parameter `log2nInRow`. `log2nInRow` must be between 2 and 10, inclusive.

direction

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `direction`.

Discussion
This performs the operation

If F = 1 $C_{nm} = \text{FDFT2D}(C_{nm})$ n = {0, N-1} and m = {0, M-1}

If F = -1 $C_{nm} = \text{IDFT2D}(C_{nm}) \cdot MN$ n = {0, N-1} and m = {0, M-1}

$$\text{FDFT2D}(X_{nm}) = \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(-j2\pi pn)/N} \cdot e^{(-j2\pi qm)/M}$$

$$\text{IDFT2D}(X_{nm}) = \frac{1}{MN} \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(j2\pi pn)/N} \cdot e^{(j2\pi qm)/M}$$

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zipt

Computes an in-place single-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

```
void vDSP_fft2d_zipt (FFTSetup setup,
DSPSplitComplex * ioData,
vDSP_Stride strideInRow,
vDSP_Stride strideInCol,
DSPSplitComplex * bufferTemp,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection direction);
```

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters `log2nInCol` and `log2nInRow` of the transform function.

ioData

A complex vector input.

strideInRow

Specifies a stride across each row of the matrix `signal`. Specifying 1 for `strideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

strideInCol

Specifies a column stride for the matrix, and should generally be allowed to default unless the matrix is a submatrix. Parameter `strideInCol` can be defaulted by specifying 0. The default column stride equals the row stride multiplied by the column count. Thus, if `strideInRow` is 1 and `strideInCol` is 0, every element of the input /output matrix is processed. If `strideInRow` is 2 and `strideInCol` is 0, every other element of each row is processed.

If not 0, parameter `strideInCol` represents the distance between each row of the matrix. If parameter `strideInCol` is 1024, for instance, complex element 512 of the matrix equates to element (1,0), element 1024 equates to element (2,0), and so forth.

bufferTemp

A temporary matrix used for storing interim results. The size of temporary memory for each part (real and imaginary) is the lower value of 16 KB or 4*`n`, where `log2n = log2nInCol + log2nInRow`.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 2 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for parameter `log2nInCol` and 6 for parameter `log2nInRow`. `log2nInRow` must be between 2 and 10, inclusive.

direction

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `direction`.

Discussion
This performs the operation

If F = 1 $C_{nm} = \text{FDFT2D}(C_{nm})$ n = {0, N-1} and m = {0, M-1}

If F = -1 $C_{nm} = \text{IDFT2D}(C_{nm}) \cdot MN$ n = {0, N-1} and m = {0, M-1}

$$\text{FDFT2D}(X_{nm}) = \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(-j2\pi pn)/N} \cdot e^{(-j2\pi qm)/M}$$

$$\text{IDFT2D}(X_{nm}) = \frac{1}{MN} \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(j2\pi pn)/N} \cdot e^{(j2\pi qm)/M}$$

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_ziptD

Computes an in-place double-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

```
void vDSP_fft2d_ziptD (FFTSetupD setup,
DSPDoubleSplitComplex * ioData,
vDSP_Stride strideInRow,
vDSP_Stride strideInCol,
DSPDoubleSplitComplex * bufferTemp,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection direction);
```

Parameters

`setup`

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters `log2nInCol` and `log2nInRow` of the transform function.

`ioData`

A complex vector input.

`strideInRow`

Specifies a stride across each row of the matrix `signal`. Specifying 1 for `strideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

strideInCol

Specifies a column stride for the matrix, and should generally be allowed to default unless the matrix is a submatrix. Parameter `strideInCol` can be defaulted by specifying 0. The default column stride equals the row stride multiplied by the column count. Thus, if `strideInRow` is 1 and `strideInCol` is 0, every element of the input /output matrix is processed. If `strideInRow` is 2 and `strideInCol` is 0, every other element of each row is processed.

If not 0, parameter `strideInCol` represents the distance between each row of the matrix. If parameter `strideInCol` is 1024, for instance, complex element 512 of the matrix equates to element (1,0), element 1024 equates to element (2,0), and so forth.

bufferTemp

A temporary matrix used for storing interim results. The size of temporary memory for each part (real and imaginary) is the lower value of 16 KB or 4*`n`, where `log2n = log2nInCol + log2nInRow`.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 2 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for parameter `log2nInCol` and 6 for parameter `log2nInRow`. `log2nInRow` must be between 2 and 10, inclusive.

direction

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `direction`.

Discussion
This performs the operation

If F = 1 $C_{nm} = \text{FDFT2D}(C_{nm})$ n = {0, N-1} and m = {0, M-1}

If F = -1 $C_{nm} = \text{IDFT2D}(C_{nm}) \cdot MN$ n = {0, N-1} and m = {0, M-1}

$$\text{FDFT2D}(X_{nm}) = \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(-j2\pi pn)/N} \cdot e^{(-j2\pi qm)/M}$$

$$\text{IDFT2D}(X_{nm}) = \frac{1}{MN} \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(j2\pi pn)/N} \cdot e^{(j2\pi qm)/M}$$

See also functions "`vDSP_create_fftsetup`", "`vDSP_create_fftsetupD`", "`vDSP_destroy_fftsetup`", and "`vDSP_destroy_fftsetupD`".

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zop

Computes an out-of-place single-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

```
void vDSP_fft2d_zop (FFTSetup setup,
DSPSplitComplex * signal,
vDSP_Stride signalStrideInRow,
vDSP_Stride signalStrideInCol,
DSPSplitComplex * result,
vDSP_Stride strideResultInRow,
vDSP_Stride strideResultInCol,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection flag);
```

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters `log2nInCol` and `log2nInRow` of the transform function.

signal

A complex vector signal input.

signalStrideInRow

Specifies a stride across each row of matrix `a`. Specifying 1 for `signalStrideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

signalStrideInCol

If not 0, this parameter represents the distance between each row of the input /output matrix.

result

The complex vector signal output.

strideResultInRow

Specifies a row stride for output matrix `result` in the same way that `signalStrideInRow` specifies a stride for input the input /output matrix.

strideResultInCol

Specifies a column stride for output matrix `result` in the same way that `signalStrideInCol` specifies a stride for input the input /output matrix.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 2 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nInCol` and 6 for `log2nInRow`. `log2nInRow` must be between 2 and 10, inclusive.

flag

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `flag`.

Discussion

This performs the operation

If F = 1 $C_m = \text{FDFT}(A_m)$ If F = -1 $C_m = \text{IDFT}(A_m) \cdot N$ m = {0, N-1}

$$\text{FDFT}(X_m) = \sum_{n=0}^{N-1} X_n \cdot e^{(-j2\pi nm)/N} \qquad \text{IDFT}(X_m) = \frac{1}{N} \sum_{n=0}^{N-1} X_n \cdot e^{(j2\pi nm)/N}$$

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zopD

Computes an out-of-place double-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

```
void vDSP_fft2d_zopD (FFTSetupD setup,
DSPDoubleSplitComplex * signal,
vDSP_Stride signalStrideInRow,
vDSP_Stride signalStrideInCol,
DSPDoubleSplitComplex * result,
vDSP_Stride strideResultInRow,
vDSP_Stride strideResultInCol,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection flag);
```

Parameters

`setup`

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters `log2nInCol` and `log2nInRow` of the transform function.

`signal`

A complex vector signal input.

`signalStrideInRow`

Specifies a stride across each row of matrix `a`. Specifying 1 for `signalStrideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

`signalStrideInCol`

If not 0, this parameter represents the distance between each row of the input /output matrix.

`result`

The complex vector signal output.

`strideResultInRow`

Specifies a row stride for output matrix `result` in the same way that `signalStrideInRow` specifies a stride for input the input /output matrix.

strideResultInCol

Specifies a column stride for output matrix `result` in the same way that `signalStrideInCol` specifies a stride for input the input /output matrix.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 2 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nInCol` and 6 for `log2nInRow`. `log2nInRow` must be between 2 and 10, inclusive.

flag

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `flag`.

Discussion
This performs the operation

If F = 1 $C_m = \text{FDFT}(A_m)$ If F = -1 $C_m = \text{IDFT}(A_m) \cdot N$ m = {0, N-1}

$$\text{FDFT}(X_m) = \sum_{n=0}^{N-1} X_n \cdot e^{(-j2\pi nm)/N} \qquad \text{IDFT}(X_m) = \frac{1}{N} \sum_{n=0}^{N-1} X_n \cdot e^{(j2\pi nm)/N}$$

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zopt

Computes an out-of-place single-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

```
void vDSP_fft2d_zopt (FFTSetup setup,
DSPSplitComplex * signal,
vDSP_Stride signalStrideInRow,
vDSP_Stride signalStrideInCol,
DSPSplitComplex * result,
vDSP_Stride strideResultInRow,
vDSP_Stride strideResultInCol,
DSPSplitComplex * bufferTemp,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection flag);
```

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters `log2nInCol` and `log2nInRow` of the transform function.

signal

A complex vector signal input.

signalStrideInRow

Specifies a stride across each row of matrix `a`. Specifying 1 for `signalStrideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

signalStrideInCol

If not 0, this parameter represents the distance between each row of the input /output matrix. If parameter signalStrideInCol is 1024, for instance, element 512 equates to element (1,0) of matrix `a`, element 1024 equates to element (2,0), and so forth.

result

The complex vector signal output.

strideResultInRow

Specifies a row stride for output matrix `result` in the same way that `signalStrideInRow` specifies a stride for input the input /output matrix.

strideResultInCol

Specifies a column stride for output matrix `result` in the same way that `signalStrideInCol` specifies a stride for input the input /output matrix.

bufferTemp

A temporary matrix used for storing interim results. The size of temporary memory for each part (real and imaginary) is the lower value of 16 KiB or 4*`n`, where `log2n = log2nInCol + log2nInRow`.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 2 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nInCol` and 6 for `log2nInRow`. `log2nInRow` must be between 2 and 10, inclusive.

flag

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `flag`.

Discussion
This performs the operation

If F = 1 $C_m = \text{FDFT}(A_m)$ If F = -1 $C_m = \text{IDFT}(A_m) \bullet N$ m = {0, N-1}

$$\text{FDFT}(X_m) = \sum_{n=0}^{N-1} X_n \bullet e^{(-j2\pi nm)/N} \qquad \text{IDFT}(X_m) = \frac{1}{N} \sum_{n=0}^{N-1} X_n \bullet e^{(j2\pi nm)/N}$$

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zoptD

Computes an out-of-place double-precision complex discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

```
void vDSP_fft2d_zoptD (FFTSetupD setup,
DSPDoubleSplitComplex * signal,
vDSP_Stride signalStrideInRow,
vDSP_Stride signalStrideInCol,
DSPDoubleSplitComplex * result,
vDSP_Stride strideResultInRow,
vDSP_Stride strideResultInCol,
DSPDoubleSplitComplex * bufferTemp,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection flag);
```

Parameters

`setup`

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters `log2nInCol` and `log2nInRow` of the transform function.

`signal`

A complex vector signal input.

`signalStrideInRow`

Specifies a stride across each row of matrix `a`. Specifying 1 for `signalStrideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

`signalStrideInCol`

If not 0, this parameter represents the distance between each row of the input /output matrix. If parameter signalStrideInCol is 1024, for instance, element 512 equates to element (1,0) of matrix `a`, element 1024 equates to element (2,0), and so forth.

result

The complex vector signal output.

strideResultInRow

Specifies a row stride for output matrix `result` in the same way that `signalStrideInRow` specifies a stride for input the input /output matrix.

strideResultInCol

Specifies a column stride for output matrix `result` in the same way that `signalStrideInCol` specifies a stride for input the input /output matrix.

bufferTemp

A temporary matrix used for storing interim results. The size of temporary memory for each part (real and imaginary) is the lower value of 16 KB or 4*`n`, where `log2n = log2nInCol + log2nInRow`.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 2 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nInCol` and 6 for `log2nInRow`. `log2nInRow` must be between 2 and 10, inclusive.

flag

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `flag`.

Discussion
This performs the operation

$$\text{If F = 1} \quad C_m = \text{FDFT}(A_m) \qquad \text{If F = -1} \quad C_m = \text{IDFT}(A_m) \bullet N \qquad \text{m} = \{0, \text{N-1}\}$$

$$\text{FDFT}(X_m) = \sum_{n=0}^{N-1} X_n \bullet e^{(-j2\pi nm)/N} \qquad \text{IDFT}(X_m) = \frac{1}{N} \sum_{n=0}^{N-1} X_n \bullet e^{(j2\pi nm)/N}$$

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zrip

Computes an in-place single-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

```
void vDSP_fft2d_zrip (FFTSetup setup,
DSPSplitComplex * ioData,
vDSP_Stride strideInRow,
vDSP_Stride strideInCol,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection direction);
```

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters log2nInCol and `log2nInRow` of the transform function.

ioData

A complex vector input.

strideInRow

Specifies a stride across each row of the input matrix signal. Specifying 1 for `strideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

strideInCol

Specifies a column stride for the matrix, and should generally be allowed to default unless the matrix is a submatrix. Parameter `strideInCol` can be defaulted by specifying 0. The default column stride equals the row stride multiplied by the column count. Thus, if `strideInRow` is 1 and `strideInCol` is 0, every element of the input /output matrix is processed. If `strideInRow` is 2 and `strideInCol` is 0, every other element of each row is processed.

If not 0, `strideInCol` represents the distance between each row of the matrix. If `strideInCol` is 1024, for instance, complex element 512 of the matrix equates to element (1,0), element 1024 equates to element (2,0), and so forth.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 2 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nInCol` and 6 for `log2nInRow`. `log2nInRow` must be between 2 and 10, inclusive.

direction

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `direction`.

Discussion

Forward transforms read real input and write packed complex output. Inverse transforms read packed complex input and write real output. As a result of packing the frequency-domain data, spatial-domain data and its equivalent frequency-domain data have the same storage requirements.

If F = 1 $C_{nm} = \text{FDFT2D}(C_{nm}) \cdot 2$ n = {0, N-1} and m = {0, M-1}

If F = -1 $C_{nm} = \text{IDFT2D}(C_{nm}) \cdot MN$ n = {0, N-1} and m = {0, M-1}

$$\text{FDFT2D}(X_{nm}) = \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(-j2\pi pn)/N} \cdot e^{(-j2\pi qm)/M}$$

$$\text{IDFT2D}(X_{nm}) = \frac{1}{MN} \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(j2\pi pn)/N} \cdot e^{(j2\pi qm)/M}$$

Real data is stored in split complex form, with odd reals stored on the imaginary side of the split complex form and even reals in stored on the real side.

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zripD

Computes an in-place double-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

```
void vDSP_fft2d_zripD (FFTSetupD setup,
DSPDoubleSplitComplex * signal,
vDSP_Stride strideInRow,
vDSP_Stride strideInCol,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection flag);
```

Parameters

`setup`

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters log2nInCol and `log2nInRow` of the transform function.

`signal`

A complex vector signal input.

`strideInRow`

Specifies a stride across each row of the input matrix signal. Specifying 1 for `strideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

strideInCol

Specifies a column stride for the matrix, and should generally be allowed to default unless the matrix is a submatrix. Parameter `strideInCol` can be defaulted by specifying 0. The default column stride equals the row stride multiplied by the column count. Thus, if `strideInRow` is 1 and `strideInCol` is 0, every element of the input /output matrix is processed. If `strideInRow` is 2 and `strideInCol` is 0, every other element of each row is processed.

If not 0, `strideInCol` represents the distance between each row of the matrix. If `strideInCol` is 1024, for instance, complex element 512 of the matrix equates to element (1,0), element 1024 equates to element (2,0), and so forth.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 2 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nInCol` and 6 for `log2nInRow`. `log2nInRow` must be between 2 and 10, inclusive.

flag

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `flag`.

Discussion

Forward transforms read real input and write packed complex output. Inverse transforms read packed complex input and write real output. As a result of packing the frequency-domain data, spatial-domain data and its equivalent frequency-domain data have the same storage requirements.

If F = 1 $C_{nm} = \text{FDFT2D}(C_{nm}) \cdot 2$ n = {0, N-1} and m = {0, M-1}

If F = -1 $C_{nm} = \text{IDFT2D}(C_{nm}) \cdot MN$ n = {0, N-1} and m = {0, M-1}

$$\text{FDFT2D}(X_{nm}) = \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(-j2\pi pn)/N} \cdot e^{(-j2\pi qm)/M}$$

$$\text{IDFT2D}(X_{nm}) = \frac{1}{MN} \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(j2\pi pn)/N} \cdot e^{(j2\pi qm)/M}$$

Real data is stored in split complex form, with odd reals stored on the imaginary side of the split complex form and even reals in stored on the real side.

No Altivec/SSE support for double precision. The function always invokes scalar code.

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability

Available in Mac OS X v10.4 and later.

Declared In

`vDSP.h`

vDSP_fft2d_zript

Computes an in-place single-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

```
void vDSP_fft2d_zript (FFTSetup setup,
DSPSplitComplex * ioData,
vDSP_Stride strideInRow,
vDSP_Stride strideInCol,
DSPSplitComplex * bufferTemp,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection direction);
```

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters log2nInCol and `log2nInRow` of the transform function.

ioData

A complex vector input.

strideInRow

Specifies a stride across each row of the input matrix signal. Specifying 1 for `strideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

strideInCol

Specifies a column stride for the matrix, and should generally be allowed to default unless the matrix is a submatrix. Parameter `strideInCol` can be defaulted by specifying 0. The default column stride equals the row stride multiplied by the column count. Thus, if `strideInRow` is 1 and `strideInCol` is 0, every element of the input /output matrix is processed. If `strideInRow` is 2 and `strideInCol` is 0, every other element of each row is processed.

If not 0, `strideInCol` represents the distance between each row of the matrix. If `strideInCol` is 1024, for instance, complex element 512 of the matrix equates to element (1,0), element 1024 equates to element (2,0), and so forth.

bufferTemp

A temporary matrix used for storing interim results. The size of temporary memory required is discussed below.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 3 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nInCol` and 6 for `log2nInRow`. `log2nInRow` must be between 3 and 10, inclusive.

direction

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `direction`.

Discussion
Forward transforms read real input and write packed complex output. Inverse transforms read packed complex input and write real output. As a result of packing the frequency-domain data, spatial-domain data and its equivalent frequency-domain data have the same storage requirements.

If F = 1 $C_{nm} = \text{FDFT2D}(C_{nm}) \cdot 2$ n = {0, N-1} and m = {0, M-1}

If F = -1 $C_{nm} = \text{IDFT2D}(C_{nm}) \cdot MN$ n = {0, N-1} and m = {0, M-1}

$$\text{FDFT2D}(X_{nm}) = \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(-j2\pi pn)/N} \cdot e^{(-j2\pi qm)/M}$$

$$\text{IDFT2D}(X_{nm}) = \frac{1}{MN} \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(j2\pi pn)/N} \cdot e^{(j2\pi qm)/M}$$

Real data is stored in split complex form, with odd reals stored on the imaginary side of the split complex form and even reals in stored on the real side.

The space needed in `bufferTemp` is at most max(9*nr, nc/2) elements in each of `realp` and `imagp`. Here is an example of how to allocate the space:

```
int nr, nc, tempSize;
nr = 1<<log2InRow;
nc = 1<<log2InCol;
tempSize = max(9*nr, nc/2);
bufferTemp.realp = ( float* ) malloc (tempSize * sizeOf ( float ) );
bufferTemp.imagp = ( float* ) malloc (tempSize * sizeOf ( float ) );
```

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zriptD

Computes an in-place double-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

```
void vDSP_fft2d_zriptD (FFTSetupD setup,
DSPDoubleSplitComplex * signal,
vDSP_Stride strideInRow,
vDSP_Stride strideInCol,
DSPDoubleSplitComplex * bufferTemp,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection flag);
```

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the values supplied as parameters log2nInCol and `log2nInRow` of the transform function.

signal

A complex vector signal input.

strideInRow

Specifies a stride across each row of the input matrix signal. Specifying 1 for `strideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

strideInCol

Specifies a column stride for the matrix, and should generally be allowed to default unless the matrix is a submatrix. Parameter `strideInCol` can be defaulted by specifying 0. The default column stride equals the row stride multiplied by the column count. Thus, if `strideInRow` is 1 and `strideInCol` is 0, every element of the input /output matrix is processed. If `strideInRow` is 2 and `strideInCol` is 0, every other element of each row is processed.

If not 0, `strideInCol` represents the distance between each row of the matrix. If `strideInCol` is 1024, for instance, complex element 512 of the matrix equates to element (1,0), element 1024 equates to element (2,0), and so forth.

bufferTemp

A temporary matrix used for storing interim results. The size of temporary memory required is discussed below.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 3 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nInCol` and 6 for `log2nInRow`. `log2nInRow` must be between 3 and 10, inclusive.

flag

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `flag`.

Discussion

Forward transforms read real input and write packed complex output. Inverse transforms read packed complex input and write real output. As a result of packing the frequency-domain data, spatial-domain data and its equivalent frequency-domain data have the same storage requirements.

If F = 1 $C_{nm} = \text{FDFT2D}(C_{nm}) \cdot 2$ n = {0, N-1} and m = {0, M-1}

If F = -1 $C_{nm} = \text{IDFT2D}(C_{nm}) \cdot MN$ n = {0, N-1} and m = {0, M-1}

$$\text{FDFT2D}(X_{nm}) = \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(-j2\pi pn)/N} \cdot e^{(-j2\pi qm)/M}$$

$$\text{IDFT2D}(X_{nm}) = \frac{1}{MN} \sum_{p=0}^{N-1} \sum_{q=0}^{M-1} X_{pq} \cdot e^{(j2\pi pn)/N} \cdot e^{(j2\pi qm)/M}$$

Real data is stored in split complex form, with odd reals stored on the imaginary side of the split complex form and even reals in stored on the real side.

The space needed in `bufferTemp` is at most max(9*nr, nc/2) elements in each of `realp` and `imagp`. Here is an example of how to allocate the space:

```
int nr, nc, tempSize;
nr = 1<<log2InRow;
nc = 1<<log2InCol;
tempSize = max(9*nr, nc/2);
bufferTemp.realp = ( float* ) malloc (tempSize * sizeOf ( float ) );
bufferTemp.imagp = ( float* ) malloc (tempSize * sizeOf ( float ) );
```

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zrop

Computes an out-of-place single-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

```
void vDSP_fft2d_zrop (FFTSetup setup,
DSPSplitComplex * signal,
vDSP_Stride signalStrideInRow,
vDSP_Stride signalStrideInCol,
DSPSplitComplex * result,
vDSP_Stride strideResultInRow,
vDSP_Stride strideResultInCol,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection flag);
```

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the value supplied as parameter `log2n` or `log2m`, whichever is larger, of the transform function.

signal

A complex vector signal input.

signalStrideInRow

Specifies a stride across each row of matrix signal. Specifying 1 for `signalStrideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

signalStrideInCol

If not 0, represents the distance between each row of the input /output matrix. If parameter signalStrideInCol is 1024, for instance, element 512 equates to element (1,0) of matrix `a`, element 1024 equates to element (2,0), and so forth.

result

The complex vector signal output.

strideResultInRow

Specifies a row stride for output matrix `c` in the same way that `signalStrideInRow` specifies strides for input the matrix.

strideResultInCol

Specifies a column stride for output matrix `c` in the same way that `signalStrideInCol` specify strides for input the matrix.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 3 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nInCol` and 6 for `log2nInRow`. `log2nInRow` must be between 3 and 10, inclusive.

flag

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `flag`.

Discussion
Forward transforms read real input and write packed complex output. Inverse transforms read packed complex input and write real output. As a result of packing the frequency-domain data, spatial-domain data and its equivalent frequency-domain data have the same storage requirements.

Real data is stored in split complex form, with odd reals stored on the imaginary side of the split complex form and even reals in stored on the real side.

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zropD

Computes an out-of-place double-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse).

```
void vDSP_fft2d_zropD (FFTSetupD setup,
DSPDoubleSplitComplex * ioData,
vDSP_Stride Kr,
vDSP_Stride Kc,
DSPDoubleSplitComplex * ioData2,
vDSP_Stride Ir,
vDSP_Stride Ic,
vDSP_Length log2nc,
vDSP_Length log2nr,
FFTDirection flag);
```

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the value supplied as parameter `log2n` or `log2m`, whichever is larger, of the transform function.

ioData

A complex vector input.

Kr

Specifies a stride across each row of matrix signal. Specifying 1 for `Kr` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

Kc

If not 0, represents the distance between each row of the input /output matrix. If parameter `Kc` is 1024, for instance, element 512 equates to element (1,0) of matrix `a`, element 1024 equates to element (2,0), and so forth.

ioData2

The complex vector result.

Ir

Specifies a row stride for output matrix `ioData2` in the same way that `Kr` specifies strides for input the matrix.

Ic

Specifies a column stride for output matrix `ioData2` in the same way that `Kc` specify strides for input matrix `ioData`.

log2nc

The base 2 exponent of the number of columns to process for each row. `log2nc` must be between 3 and 10, inclusive.

log2nr

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nc` and 6 for `log2nr`. `log2nr` must be between 3 and 10, inclusive.

flag

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `flag`.

Discussion
Forward transforms read real input and write packed complex output. Inverse transforms read packed complex input and write real output. As a result of packing the frequency-domain data, spatial-domain data and its equivalent frequency-domain data have the same storage requirements.

Real data is stored in split complex form, with odd reals stored on the imaginary side of the split complex form and even reals in stored on the real side.

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zropt

Computes an out-of-place single-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

```
void vDSP_fft2d_zropt (FFTSetup setup,
DSPSplitComplex * signal,
vDSP_Stride signalStrideInRow,
vDSP_Stride signalStrideInCol,
DSPSplitComplex * result,
vDSP_Stride strideResultInRow,
vDSP_Stride strideResultInCol,
DSPSplitComplex * bufferTemp,
vDSP_Length log2nInCol,
vDSP_Length log2nInRow,
FFTDirection flag);
```

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the value supplied as parameter `log2n` or `log2m`, whichever is larger, of the transform function.

signal

A complex vector signal input.

signalStrideInRow

Specifies a stride across each row of matrix `signal`. Specifying 1 for `signalStrideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

signalStrideInCol

If not 0, represents the distance between each row of matrix `signal`. If parameter `signalStrideInCol` is 1024, for instance, element 512 equates to element (1,0) of matrix `signal`, element 1024 equates to element (2,0), and so forth.

result

The complex vector signal output.

strideResultInRow

Specifies a row stride for output matrix `result` in the same way that `signalStrideInRow` specifies strides for input matrix `result`.

strideResultInCol

Specifies a column stride for output matrix `c` in the same way that `signalStrideInCol` specify strides for input matrix `result`.

bufferTemp

A temporary matrix used for storing interim results. The size of temporary memory for each part (real and imaginary) can be calculated using the algorithm shown below.

log2nInCol

The base 2 exponent of the number of columns to process for each row. `log2nInCol` must be between 3 and 10, inclusive.

log2nInRow

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nInCol` and 6 for `log2nInRow`. `log2nInRow` must be between 3 and 10, inclusive.

flag

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `flag`.

Discussion
Here is the `bufferTemp` size algorithm:

```
int nr, nc, tempSize;
nr = 1<<log2InRow;
nc = 1<<log2InCol;
if ( ( (log2InCol-1) < 3 ) || ( log2InRow > 9)
{
tempSize = 9 * nr;
}
else
{
tempSize = 17 * nr
}
bufferTemp.realp = (float*) malloc (tempSize * sizeOf (float));
bufferTemp.imagp = (float*) malloc (tempSize * sizeOf (float));
```

Forward transforms read real input and write packed complex output. Inverse transforms read packed complex input and write real output. As a result of packing the frequency-domain data, spatial-domain data and its equivalent frequency-domain data have the same storage requirements.

Real data is stored in split complex form, with odd reals stored on the imaginary side of the split complex form and even reals in stored on the real side.

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

vDSP_fft2d_zroptD

Computes an out-of-place double-precision real discrete FFT, either from the spatial domain to the frequency domain (forward) or from the frequency domain to the spatial domain (inverse). A buffer is used for intermediate results.

```
void vDSP_fft2d_zroptD (FFTSetupD setup,
DSPDoubleSplitComplex * ioData,
vDSP_Stride Kr,
vDSP_Stride Kc,
DSPDoubleSplitComplex * ioData2,
vDSP_Stride Ir,
vDSP_Stride Ic,
DSPDoubleSplitComplex * temp,
vDSP_Length log2nc,
vDSP_Length log2nr,
FFTDirection flag);
```

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `vDSP_create_fftsetup` or `vDSP_create_fftsetupD`. The value supplied as parameter `log2n` of the setup function must equal or exceed the value supplied as parameter `log2n` or `log2m`, whichever is larger, of the transform function.

ioData

A complex vector input.

Kr

Specifies a stride across each row of matrix signal. Specifying 1 for `signalStrideInRow` processes every element across each row, specifying 2 processes every other element across each row, and so forth.

Kc

If not 0, represents the distance between each row of the input /output matrix. If parameter signalStrideInCol is 1024, for instance, element 512 equates to element (1,0) of matrix `a`, element 1024 equates to element (2,0), and so forth.

ioData2

The complex vector result.

Ir

Specifies a row stride for output matrix `ioData2` in the same way that `Kr` specifies strides for input the matrix.

Ic

Specifies a column stride for output matrix `ioData2` in the same way that `Kc` specify strides for input matrix `ioData`.

temp

A temporary matrix used for storing interim results. The size of temporary memory for each part (real and imaginary) is the lower value of 16 KB or 4*`n`, where `log2n = log2nInCol + log2nInRow`.

log2nc

The base 2 exponent of the number of columns to process for each row. `log2nc` must be between 3 and 10, inclusive.

log2nr

The base 2 exponent of the number of rows to process. For example, to process 64 rows of 128 columns, specify 7 for `log2nc` and 6 for `log2nr`. `log2nr` must be between 3 and 10, inclusive.

flag

A forward/inverse directional flag, and must specify `FFT_FORWARD` for a forward transform or `FFT_INVERSE` for an inverse transform.

Results are undefined for other values of `flag`.

Discussion
Forward transforms read real input and write packed complex output. Inverse transforms read packed complex input and write real output. As a result of packing the frequency-domain data, spatial-domain data and its equivalent frequency-domain data have the same storage requirements.

Real data is stored in split complex form, with odd reals stored on the imaginary side of the split complex form and even reals in stored on the real side.

See also functions `vDSP_create_fftsetup`, `vDSP_create_fftsetupD`, `vDSP_destroy_fftsetup`, and `vDSP_destroy_fftsetupD`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`vDSP.h`

Document Revision History

This document describes the C API for the two-dimensional Fast Fourier Transform functions in vecLib.

This table describes the changes to *vDSP Two-Dimensional Fast Fourier Transforms Reference*.

Date	Notes
2008-11-19	Improved function discussions and corrected formulae.
2007-06-15	New document that describes the C API for the vDSP functionality for two-dimensional Fast Fourier Transforms

Document Revision History

Index

V