vDSP Vector-To-Scalar Operations Reference

Performance > Carbon

2009-01-06

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

Simultaneously published in the United States and Canada.

Contents

vDSP Vector-To-Scalar Operations Reference

Framework: Accelerate/vecLib

Declared in vDSP.h

Overview

This document describes the C API for the vDSP functions that receive a vector as input and compute scalars as output. Examples of such operations include calculating the dot product of a vector, or finding the minimum or maximum of a vector.

Functions by Task

Calculating Dot Products

`vDSP_dotpr` (page 8)
Computes the dot or scalar product of vectors `A` and `B` and leaves the result in scalar `C`; single precision.

`vDSP_dotprD` (page 8)
Computes the dot or scalar product of vectors `A` and `B` and leaves the result in scalar `C`; double precision.

`vDSP_zdotpr` (page 33)
Calculates the complex dot product of complex vectors signal1 and signal2 and leaves the result in complex vector `result`; single precision.

`vDSP_zdotprD` (page 33)
Calculates the complex dot product of complex vectors signal1 and signal2 and leaves the result in complex vector `result`; double precision.

`vDSP_zidotpr` (page 34)
Calculates the conjugate dot product (or inner dot product) of complex vectors `signal1` and `signal2` and leave the result in complex vector `result`; single precision.

`vDSP_zidotprD` (page 34)
Calculates the conjugate dot product (or inner dot product) of complex vectors signal1 and signal2 and leave the result in complex vector `result`; double precision.

`vDSP_zrdotpr` (page 35)
Calculates the complex dot product of complex vector `A` and real vector `B` and leaves the result in complex vector `C`; single precision.

`vDSP_zrdotprD` (page 35)
Calculates the complex dot product of complex vector `A` and real vector `B` and leaves the result in complex vector `C`; double precision.

Finding Maximums

`vDSP_maxv` (page 12)
Vector maximum value; single precision.

`vDSP_maxvD` (page 12)
Vector maximum value; double precision.

`vDSP_maxvi` (page 13)
Vector maximum value with index; single precision.

`vDSP_maxviD` (page 14)
Vector maximum value with index; double precision.

`vDSP_maxmgv` (page 9)
Vector maximum magnitude; single precision.

`vDSP_maxmgvD` (page 9)
Vector maximum magnitude; double precision.

`vDSP_maxmgvi` (page 10)
Vector maximum magnitude with index; single precision.

`vDSP_maxmgviD` (page 11)
Vector maximum magnitude with index; double precision.

Finding Minimums

`vDSP_minv` (page 22)
Vector minimum value.

`vDSP_minvD` (page 22)
Vector minimum value; double precision.

`vDSP_minvi` (page 23)
Vector minimum value with index; single precision.

`vDSP_minviD` (page 24)
Vector minimum value with index; double precision.

`vDSP_minmgv` (page 19)
Vector minimum magnitude; single precision.

`vDSP_minmgvD` (page 19)
Vector minimum magnitude; double precision.

`vDSP_minmgvi` (page 20)
Vector minimum magnitude with index; single precision.

`vDSP_minmgviD` (page 21)
Vector minimum magnitude with index; double precision.

Calculating Means

`vDSP_meanv` (page 16)
Vector mean value; single precision.

Summing Vectors

Functions

vDSP_dotpr

Computes the dot or scalar product of vectors `A` and `B` and leaves the result in scalar `C`; single precision.

```
void vDSP_dotpr (const float A[],
vDSP_Stride I,
const float B[],
vDSP_Stride J,
float * C,
vDSP_Length N);
```

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI} \cdot B_{nJ} \qquad \text{n} = \{0, \text{N-1}\}$$

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

Declared In
`vDSP.h`

vDSP_dotprD

Computes the dot or scalar product of vectors `A` and `B` and leaves the result in scalar `C`; double precision.

```
void vDSP_dotprD (const double A[],
vDSP_Stride I,
const double B[],
vDSP_Stride J,
double * C,
vDSP_Length N);
```

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI} \cdot B_{nJ} \qquad \text{n} = \{0, \text{N-1}\}$$

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

Declared In
`vDSP.h`

vDSP_maxmgv

Vector maximum magnitude; single precision.

```
void vDSP_maxmgv (const float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector. If passed a vector with no elements, this function returns a value of 0 in `C`.

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

```
*C = 0;
for (n = 0; n < N; ++n)
    if (*C < abs(A[n*I]))
        *C = abs(A[n*I]);
```

Finds the element with the greatest magnitude in vector `A` and copies this value to scalar `C`.

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

Declared In
`vDSP.h`

vDSP_maxmgvD

Vector maximum magnitude; double precision.

```
void vDSP_maxmgvD (const double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A

Double-precision real input vector. If passed a vector with no elements, this function returns a value of 0 in `C`.

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

```
*C = 0;
for (n = 0; n < N; ++n)
    if (*C < abs(A[n*I]))
        *C = abs(A[n*I]);
```

Finds the element with the greatest magnitude in vector `A` and copies this value to scalar `C`.

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

Declared In
`vDSP.h`

vDSP_maxmgvi

Vector maximum magnitude with index; single precision.

```
void vDSP_maxmgvi (float * A,
vDSP_Stride I,
float * C,
vDSP_Length * IC,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector. If passed a vector with no elements, this function returns a value of 0 in `*C`, and the value returned in `IC` is undefined.

I

Stride for `A`

C

Output scalar

IC

Output scalar index

N

Count

Discussion
This performs the operation

```
*C = 0;
for (n = 0; n < N; ++n)
{
    if (*C < abs(A[n*I]))
    {
        *C = abs(A[n*I]);
        *IC = n*I;
    }
}
```

Copies the element with the greatest magnitude from real vector `A` to real scalar `C`, and writes its zero-based index to integer scalar `IC`. The index is the actual array index, not the pre-stride index. If vector `A` contains more than one instance of the maximum magnitude, `IC` contains the index of the first instance.

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

Declared In
`vDSP.h`

vDSP_maxmgviD

Vector maximum magnitude with index; double precision.

```
void vDSP_maxmgviD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length * IC,
vDSP_Length N);
```

Parameters

A

Double-precision real input vector. If passed a vector with no elements, this function returns a value of 0 in `*C`. The value returned in `IC` is undefined.

I

Stride for `A`

C

Output scalar

IC

Output scalar

N

Count

Discussion
This performs the operation

```
*C = 0;
for (n = 0; n < N; ++n)
{
    if (*C < abs(A[n*I]))
    {
        *C = abs(A[n*I]);
        *IC = n*I;
    }
}
```

Copies the element with the greatest magnitude from real vector `A` to real scalar `C`, and writes its zero-based index to integer scalar `IC`. The index is the actual array index, not the pre-stride index. If vector `A` contains more than one instance of the maximum magnitude, `IC` contains the index of the first instance.

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

Declared In
vDSP.h

vDSP_maxv

Vector maximum value; single precision.

```
void vDSP_maxv (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector.

I

Stride for A

C

Output scalar

N

Count

Discussion
This performs the operation

```
*C = -INFINITY;
for (n = 0; n < N; ++n)
    if (*C < A[n*I])
        *C = A[n*I];
```

Finds the element with the greatest value in vector A and copies this value to scalar C. If A has length of 0, this function returns -INFINITY.

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

Declared In
vDSP.h

vDSP_maxvD

Vector maximum value; double precision.

```
void vDSP_maxvD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A

Double-precision real input vector

I
Stride for `A`

C
Output scalar

N
Count

Discussion
This performs the operation

```
*C = -INFINITY;
for (n = 0; n < N; ++n)
    if (*C < A[n*I])
        *C = A[n*I];
```

Finds the element with the greatest value in vector `A` and copies this value to scalar `C`. If `A` has length of 0, this function returns `-INFINITY`.

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

Declared In
`vDSP.h`

vDSP_maxvi

Vector maximum value with index; single precision.

```
void vDSP_maxvi (float * A,
vDSP_Stride I,
float * C,
vDSP_Length * IC,
vDSP_Length N);
```

Parameters

A
Single-precision real input vector.

I
Stride for `A`

C
Output scalar

IC
Output scalar index

N
Count

Discussion
This performs the operation

```
*C = -INFINITY;
for (n = 0; n < N; ++n)
{
    if (*C < A[n * I])
```

```
    {
        *C = A[n * I];
        *IC = n * I;
    }
}
```

Copies the element with the greatest value from real vector `A` to real scalar `C`, and writes its zero-based index to integer scalar `IC`. The index is the actual array index, not the pre-stride index. If vector `A` contains more than one instance of the maximum value, `IC` contains the index of the first instance. If `A` is vector with no elements, this function returns a value of -infinity in `C` and `*IC` is undetermined.

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

Declared In
`vDSP.h`

vDSP_maxviD

Vector maximum value with index; double precision.

```
void vDSP_maxviD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length * IC,
vDSP_Length N);
```

Parameters

A
: Double-precision real input vector

I
: Stride for `A`

C
: Output scalar

IC
: Output scalar

N
: Count

Discussion
This performs the operation

```
*C = -INFINITY;
for (n = 0; n < N; ++n)
{
    if (*C < A[n * I])
    {
        *C = A[n * I];
        *IC = n * I;
    }
}
```

Copies the element with the greatest value from real vector `A` to real scalar `C`, and writes its zero-based index to integer scalar `IC`. The index is the actual array index, not the pre-stride index. If vector `A` contains more than one instance of the maximum value, `IC` contains the index of the first instance. If `A` is vector with no elements, this function returns a value of -infinity in `C` and `*IC` is undetermined.

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

Declared In
`vDSP.h`

vDSP_meamgv

Vector mean magnitude; single precision.

```
void vDSP_meamgv (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$C = \frac{1}{N} \sum_{n=0}^{N-1} \left| A_{ni} \right|$$

Finds the mean of the magnitudes of elements of vector `A` and stores this value in scalar `C`.

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

Declared In
`vDSP.h`

vDSP_meamgvD

Vector mean magnitude; double precision.

```
void vDSP_meamgvD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A

Double-precision real input vector

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$C = \frac{1}{N} \sum_{n=0}^{N-1} \left| A_{ni} \right|$$

Finds the mean of the magnitudes of elements of vector `A` and stores this value in scalar `C`.

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

Declared In
`vDSP.h`

vDSP_meanv

Vector mean value; single precision.

```
void vDSP_meanv (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector. If passed an array of length 0, this function returns a NaN.

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$C = \frac{1}{N} \sum_{n=0}^{N-1} A_{ni}$$

Finds the mean value of the elements of vector `A` and stores this value in scalar `C`.

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

Declared In
`vDSP.h`

vDSP_meanvD

Vector mean value; double precision.

```
void vDSP_meanvD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A

Double-precision real input vector

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$C = \frac{1}{N} \sum_{n=0}^{N-1} A_{ni}$$

Finds the mean value of the elements of vector `A` and stores this value in scalar `C`.

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

Declared In
`vDSP.h`

vDSP_measqv

Vector mean square value; single precision.

```
void vDSP_measqv (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector. If passed an array of length 0, this function returns a NaN.

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$C = \frac{1}{N} \sum_{n=0}^{N-1} A_{ni}^2$$

Finds the mean value of the squares of the elements of vector `A` and stores this value in scalar `C`.

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

Declared In
`vDSP.h`

vDSP_measqvD

Vector mean square value; double precision.

```
void vDSP_measqvD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A

Double-precision real input vector

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$C = \frac{1}{N} \sum_{n=0}^{N-1} A_{ni}^2$$

Finds the mean value of the squares of the elements of vector `A` and stores this value in scalar `C`.

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

Declared In
`vDSP.h`

vDSP_minmgv

Vector minimum magnitude; single precision.

```
void vDSP_minmgv (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector. If passed an array of length 0, this function returns +INF.

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$c = \left|A_0\right| \quad \text{If} \quad c > \left|A_{ni}\right| \quad \text{then} \quad c = \left|A_{ni}\right| \qquad \text{n} = \{1, \text{N-1}\}$$

Finds the element with the least magnitude in vector `A` and copies this value to scalar `C`.

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

Declared In
`vDSP.h`

vDSP_minmgvD

Vector minimum magnitude; double precision.

```
void vDSP_minmgvD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A

Double-precision real input vector

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$c = \left|A_0\right| \quad \text{If} \quad c > \left|A_{ni}\right| \quad \text{then} \quad c = \left|A_{ni}\right| \qquad \text{n} = \{1, \text{N-1}\}$$

Finds the element with the least magnitude in vector `A` and copies this value to scalar `C`.

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

Declared In
`vDSP.h`

vDSP_minmgvi

Vector minimum magnitude with index; single precision.

```
void vDSP_minmgvi (float * A,
vDSP_Stride I,
float * C,
vDSP_Length * IC,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector. If passed a zero vector, this function returns a value of +INF with an indeterminate index.

I

Stride for `A`

C

Output scalar

IC

Output scalar index

N

Count

Discussion
This performs the operation

$$c = |A_0| \quad d = 0 \qquad \text{If} \quad c > |A_{ni}| \quad \text{then} \quad c = |A_{ni}| \quad d = ni \qquad \text{n} = \{1, \text{N-1}\}$$

Copies the element with the least magnitude from real vector `A` to real scalar `C`, and writes its zero-based index to integer scalar `IC`. The index is the actual array index, not the pre-stride index. If vector `A` contains more than one instance of the least magnitude, `IC` contains the index of the first instance.

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

Declared In
`vDSP.h`

vDSP_minmgviD

Vector minimum magnitude with index; double precision.

```
void vDSP_minmgviD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length * IC,
vDSP_Length N);
```

Parameters

A
: Double-precision real input vector

I
: Stride for `A`

C
: Output scalar

IC
: Output scalar

N
: Count

Discussion
This performs the operation

$$c = |A_0| \quad d = 0 \qquad \text{If} \quad c > |A_{ni}| \quad \text{then} \quad c = |A_{ni}| \quad d = ni \qquad \text{n} = \{1, \text{N-1}\}$$

Copies the element with the least magnitude from real vector `A` to real scalar `C`, and writes its zero-based index to integer scalar `IC`. The index is the actual array index, not the pre-stride index. If vector `A` contains more than one instance of the least magnitude, `IC` contains the index of the first instance.

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

Declared In
`vDSP.h`

vDSP_minv

Vector minimum value.

```
void vDSP_minv (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector. If passed an array of length 0, this function returns +INF.

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$c = A_0 \quad \text{If} \quad c > A_{ni} \quad \text{then} \quad c = A_{ni} \qquad \text{n} = \{1, \text{N-1}\}$$

Finds the element with the least value in vector `A` and copies this value to scalar `C`.

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

Declared In
`vDSP.h`

vDSP_minvD

Vector minimum value; double precision.

```
void vDSP_minvD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A

Double-precision real input vector

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$c = A_0 \quad \text{If} \quad c > A_{ni} \quad \text{then} \quad c = A_{ni} \qquad \text{n} = \{1, \text{N-1}\}$$

Finds the element with the least value in vector `A` and copies this value to scalar `C`.

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

Declared In
`vDSP.h`

vDSP_minvi

Vector minimum value with index; single precision.

```
void vDSP_minvi (float * A,
vDSP_Stride I,
float * C,
vDSP_Length * IC,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector. If passed a zero vector, this function returns a value of +INF with an indeterminate index.

I

Stride for `A`

C

Output scalar

IC

Output scalar index

N

Count

Discussion
This performs the operation

$$\begin{matrix} c = A_0 \\ d = 0 \end{matrix} \quad \text{If} \quad c > A_{ni} \quad \text{then} \quad \begin{matrix} c = A_{ni} \\ d = ni \end{matrix} \qquad \text{n} = \{1, \text{N-1}\}$$

Copies the element with the least value from real vector `A` to real scalar `C`, and writes its zero-based index to integer scalar `IC`. The index is the actual array index, not the pre-stride index. If vector `A` contains more than one instance of the least value, `IC` contains the index of the first instance.

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

Declared In
`vDSP.h`

vDSP_minviD

Vector minimum value with index; double precision.

```
void vDSP_minviD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length * IC,
vDSP_Length N);
```

Parameters

A
: Double-precision real input vector

I
: Stride for `A`

C
: Output scalar

IC
: Output scalar

N
: Count

Discussion
This performs the operation

$$\begin{matrix} c = A_0 \\ d = 0 \end{matrix} \quad \text{If} \quad c > A_{ni} \quad \text{then} \quad \begin{matrix} c = A_{ni} \\ d = ni \end{matrix} \qquad \text{n} = \{1, \text{N-1}\}$$

Copies the element with the least value from real vector `A` to real scalar `C`, and writes its zero-based index to integer scalar `IC`. The index is the actual array index, not the pre-stride index. If vector `A` contains more than one instance of the least value, `IC` contains the index of the first instance.

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

Declared In
`vDSP.h`

vDSP_mvessq

Vector mean of signed squares; single precision.

```
void vDSP_mvessq (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector.

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$C = \frac{1}{N} \sum_{n=0}^{N-1} A_{ni} \cdot \left| A_{ni} \right|$$

Finds the mean value of the signed squares of the elements of vector `A` and stores this value in `*C`. If `A` is an array of length 0, this function returns a NaN. .

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

Declared In
`vDSP.h`

vDSP_mvessqD

Vector mean of signed squares; double precision.

```
void vDSP_mvessqD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A

Double-precision real input vector

I

Stride for `A`

C

Output scalar

N

Count

Discussion
This performs the operation

$$C = \frac{1}{N} \sum_{n=0}^{N-1} A_{ni} \cdot \left| A_{ni} \right|$$

Finds the mean value of the signed squares of the elements of vector `A` and stores this value in `*C`. If `A` is an array of length 0, this function returns a NaN.

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

Declared In
`vDSP.h`

vDSP_rmsqv

Vector root-mean-square; single precision.

```
void vDSP_rmsqv (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector. If passed an array of length 0, this function returns a NaN.

I

Stride for `A`

C

Single-precision real output scalar

N

Count

Discussion
This performs the operation

$$C = \sqrt{\frac{1}{N} \sum_{n=0}^{N-1} A_{nI}^2}$$

Calculates the root mean square of the elements of `A` and stores the result in `*C`

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

Declared In
vDSP.h

vDSP_rmsqvD

Vector root-mean-square; double precision.

```
void vDSP_rmsqvD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A

Double-precision real input vector

I

Stride for A

C

Double-precision real output scalar

N

Count

Discussion
This performs the operation

$$C = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1} A_{nI}^{2}}$$

Calculates the root mean square of the elements of A and stores the result in *C

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

Declared In
vDSP.h

vDSP_sve

Vector sum; single precision.

```
void vDSP_sve (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input vector.

I
Stride for A

C
Single-precision real output scalar

N
Count

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI}$$

Writes the sum of the elements of A into *C. If A is a vector with zero elements, this function returns 0 in *C.

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

Declared In
vDSP.h

vDSP_sveD

Vector sum; double precision.

```
void vDSP_sveD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A
Double-precision real input vector

I
Stride for A

C
Double-precision real output scalar

N
Count

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI}$$

Writes the sum of the elements of A into *C. If A is a vector with zero elements, this function returns 0 in *C.

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

Declared In
`vDSP.h`

vDSP_svemg

Vector sum of magnitudes; single precision.

```
void vDSP_svemg (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input scalar. If passed an value of 0, this function returns 0.

I

Stride for `A`

C

Single-precision real output scalar

N

Count

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} \left| A_{nI} \right|$$

Writes the sum of the magnitudes of the elements of `A` into `C`.

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

Declared In
`vDSP.h`

vDSP_svemgD

Vector sum of magnitudes; double precision.

```
void vDSP_svemgD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A

Double-precision real input scalar

I

Stride for `A`

C

Double-precision real output scalar

N

Count

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} \left| A_{nI} \right|$$

Writes the sum of the magnitudes of the elements of `A` into `C`.

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

Declared In
`vDSP.h`

vDSP_svesq

Vector sum of squares; single precision.

```
void vDSP_svesq (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input scalar. If passed an value of 0, this function returns 0.

I

Stride for `A`

C

Single-precision real output scalar

N

Count

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI}^2$$

Writes the sum of the squares of the elements of `A` into `C`.

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

Declared In
`vDSP.h`

vDSP_svesqD

Vector sum of squares; double precision.

```
void vDSP_svesqD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A
: Double-precision real input scalar

I
: Stride for `A`

C
: Double-precision real output scalar

N
: Count

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI}^2$$

Writes the sum of the squares of the elements of `A` into `C`.

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

Declared In
`vDSP.h`

vDSP_svs

Vector sum of signed squares; single precision.

```
void vDSP_svs (float * A,
vDSP_Stride I,
float * C,
vDSP_Length N);
```

Parameters

A

Single-precision real input scalar. If passed an value of 0, this function returns 0.

I

Stride for `A`

C

Single-precision real output scalar

N

Count

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI} \cdot \left| A_{nI} \right|$$

Writes the sum of the signed squares of the elements of `A` into `C`.

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

Declared In
`vDSP.h`

vDSP_svsD

Vector sum of signed squares; double precision.

```
void vDSP_svsD (double * A,
vDSP_Stride I,
double * C,
vDSP_Length N);
```

Parameters

A

Double-precision real input scalar

I

Stride for `A`

C

Double-precision real output scalar

N

Count

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI} \bullet \left| A_{nI} \right|$$

Writes the sum of the signed squares of the elements of `A` into `C`.

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

Declared In
`vDSP.h`

vDSP_zdotpr

Calculates the complex dot product of complex vectors signal1 and signal2 and leaves the result in complex vector `result`; single precision.

```
void vDSP_zdotpr (DSPSplitComplex * A,
vDSP_Stride I,
DSPSplitComplex * B,
vDSP_Stride J,
DSPSplitComplex * C,
vDSP_Length N);
```

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI} B_{nJ}$$

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

Declared In
`vDSP.h`

vDSP_zdotprD

Calculates the complex dot product of complex vectors signal1 and signal2 and leaves the result in complex vector `result`; double precision.

```
void vDSP_zdotprD (DSPDoubleSplitComplex * A,
vDSP_Stride I,
DSPDoubleSplitComplex * B,
vDSP_Stride J,
DSPDoubleSplitComplex * C,
vDSP_Length N);
```

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI} B_{nJ}$$

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

Declared In
`vDSP.h`

vDSP_zidotpr

Calculates the conjugate dot product (or inner dot product) of complex vectors `signal1` and `signal2` and leave the result in complex vector `result`; single precision.

```
void vDSP_zidotpr (DSPSplitComplex * A,
vDSP_Stride I,
DSPSplitComplex * B,
vDSP_Stride J,
DSPSplitComplex * C,
vDSP_Length N);
```

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI}^{*} B_{nJ}$$

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

Declared In
`vDSP.h`

vDSP_zidotprD

Calculates the conjugate dot product (or inner dot product) of complex vectors signal1 and signal2 and leave the result in complex vector `result`; double precision.

```
void vDSP_zidotprD (DSPDoubleSplitComplex * A,
vDSP_Stride I,
DSPDoubleSplitComplex * B,
vDSP_Stride J,
DSPDoubleSplitComplex * C,
vDSP_Length N);
```

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI}^{*} B_{nJ}$$

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

Declared In
`vDSP.h`

vDSP_zrdotpr

Calculates the complex dot product of complex vector `A` and real vector `B` and leaves the result in complex vector `C`; single precision.

```
void vDSP_zrdotpr (DSPSplitComplex * A,
vDSP_Stride I,
const float B[],
vDSP_Stride J,
DSPSplitComplex * C,
vDSP_Length N);
```

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI} B_{nJ}$$

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

Declared In
`vDSP.h`

vDSP_zrdotprD

Calculates the complex dot product of complex vector `A` and real vector `B` and leaves the result in complex vector `C`; double precision.

```
void vDSP_zrdotprD (DSPDoubleSplitComplex * A,
vDSP_Stride I,
const double B[],
vDSP_Stride J,
DSPDoubleSplitComplex * C,
vDSP_Length N);
```

Discussion
This performs the operation

$$C = \sum_{n=0}^{N-1} A_{nI} B_{nJ}$$

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

Declared In
`vDSP.h`

Document Revision History

This table describes the changes to *vDSP Vector-To-Scalar Operations Reference*.

Date	Notes
2009-01-06	Minor fixes to pseudocode and function parameter descriptions.
2008-11-19	Changed function parameter names to match formulae. Added pseudocode.

Document Revision History

Index

V