

fft_cipx(), fft_cipdx(), fft_ciptx(), fft_ciptdx()

■ In-Place Interleaved Complex Fourier Transform

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

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

Synopsis

```

void fft_cipx(
    FFT_setup *setup,          /* weights array */
    COMPLEX *C,               /* input/output vector */
    long k,                   /* element stride for C, must be even */
    unsigned long log2N,      /* base 2 exponent of element count */
    long f,                   /* directional flag */
    long eflag                /* ESAL flag */
);

void fft_cipdx(
    FFT_setup *setup,          /* weights array */
    DOUBLE_COMPLEX *C,        /* input/output vector */
    long k,                   /* element stride for C, must be even */
    unsigned long log2N,      /* base 2 exponent of element count */
    long f,                   /* directional flag */
    long eflag                /* ESAL flag */
);

void fft_ciptx(
    FFT_setup *setup,          /* weights array */
    COMPLEX *C,               /* input/output vector */
    long k,                   /* element stride for C, must be even */
    COMPLEX *T,               /* temporary vector */
    unsigned long log2N,      /* base 2 exponent of element count */
    long f,                   /* directional flag */
    long eflag                /* ESAL flag */
);

```

```

void fft_ciptdx(
    FFT_setupd *setup,      /* weights array */
    DOUBLE_COMPLEX *C,     /* input/output vector */
    long k,                /* element stride for C, must be even */
    DOUBLE_COMPLEX *T,     /* temporary vector */
    unsigned long log2N,   /* base 2 exponent of element count */
    long f,                /* directional flag */
    long eflag);          /* ESAL flag */

```

Description

Functions `fft_cipx()`, `fft_cipdx()`, `fft_ciptx()`, and `fft_ciptdx()` compute an in-place complex discrete Fourier transform of vector C , either from the time domain to the frequency domain (forward) or from the frequency domain to the time domain (inverse).

Parameters

setup

Points to a structure initialized by a prior call to FFT weights array function `fft_setup()` or `fft_setupd()`. The argument n of the setup function must equal or exceed the argument n of the transform function.

k

Specifies an address stride through input/output vector C . Since vector C holds complex elements, k must be a multiple of 2. Thus, to process every element of C , specify 2 for parameter k ; to process every other element, specify 4. See “[Address Strides](#)” on page 4 for stride specification.

T

A temporary vector for storing interim results of `fft_ciptx()` and `fft_ciptdx()`. Must be large enough to hold the element count as specified by parameter N .

log2N

Base 2 exponent of the number of elements to process. For example, to process 1024 elements, specify 10 for parameter $log2N$. See also “[FFT Limitations](#)” on page 43.

f

A forward/inverse directional flag, and must specify one of the following:

`FFT_FORWARD`

Forward transform.

FFT_INVERSE

Inverse transform.

Results are undefined for other values of f .

See Also

[fft_setup\(\)](#), [fft_setupd\(\)](#), [fft_free\(\)](#), [fft_freed\(\)](#) on page 242,
“SAL Fast Fourier Transforms (FFTs)” on page 29.