

# *mediaLib™ Quick Reference*

Release 2.1



**Sun Microsystems, Inc.**

901 San Antonio Road

Palo Alto, CA 94303-4900 USA

800/681-8845

[www.sun.com/sparc](http://www.sun.com/sparc)

Part Number: 802-7801-07

October 2001



# *mediaLib™ Quick Reference*

**Release 2.1**

**October 2001**

**Sun Microsystems, Inc.**

901 San Antonio Road

Palo Alto, CA 94303-4900 USA

800/681-8845

[www.sun.com/sparc](http://www.sun.com/sparc)

Part Number: 802-7801-07



Copyright © 2001 Sun Microsystems, Inc. All Rights Reserved.

THE INFORMATION CONTAINED IN THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT ANY EXPRESS REPRESENTATIONS OR WARRANTIES. IN ADDITION, SUN MICROSYSTEMS, INC. DISCLAIMS ALL IMPLIED REPRESENTATIONS AND WARRANTIES, INCLUDING ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

This document contains proprietary information of Sun Microsystems, Inc. or under license from third parties. No part of this document may be reproduced in any form or by any means or transferred to any third party without the prior written consent of Sun Microsystems, Inc.

Sun, Sun Microsystems, and the Sun logo are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the United States and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The information contained in this document is not designed or intended for use in on-line control of aircraft, air traffic, aircraft navigation or aircraft communications; or in the design, construction, operation or maintenance of any nuclear facility. Sun disclaims any express or implied warranty of fitness for such uses.

Printed in the United States of America.



# Contents

---

## Usage Notes 1

## Imaging Functions and Syntax 2

Imaging: Arithmetic 2

Imaging: Fourier Transform 8

Imaging: Geometric 8

Imaging: Image Composition 13

Imaging: Image Conversion Functions —Data Format Conversion 13

Imaging: Image Conversion Functions —General Color Space Conversion 14

Imaging: Image Conversion Functions —RGB to Monochrome Color Space Conversion 14

Imaging: Image Conversion Functions —RGB/HSL Color Space Conversion 15

Imaging: Image Conversion Functions —RGB/HSV Color Space Conversion 15

Imaging: Image Conversion Functions —RGB/XYZ Color Space Conversion 15

Imaging: Image Conversion Functions —R'G'B'/Y'CBCR Color Space Conversion 16

Imaging: Image Conversion Functions —True Color to Indexed Color Conversion	16
Imaging: Image Copying and Clearing	17
Imaging: Image Edge Detection	18
Imaging: Image Initialization Functions —Get Functions	18
Imaging: Image Initialization Functions —Image Structure Creation and Deletion	20
Imaging: Image Statistics	21
Imaging: Linear Spatial	22
Imaging: Logical	25
Imaging: Morphological	27
Imaging: Nonlinear Spatial	28
Imaging: Radiometric	30

## **Linear Algebra Functions and Syntax 32**

Linear Algebra: Data Arrangement Functions	32
Linear Algebra: Matrix Functions	33
Linear Algebra: Vector Functions	34

<b>Signal and Audio Functions and Syntax</b>	<b>37</b>
Signal and Audio: Channel Merge and Split	37
Signal and Audio: Cepstral Analysis	37
Signal and Audio: Convolution and Correlation	38
Signal and Audio: Data Type Conversion	39
Signal and Audio: Dynamic Time Warping	39
Signal and Audio: Encoding and Decoding	42
Signal and Audio: Fast Fourier Transform (FFT)	43
Signal and Audio: Finite Impulse Response (FIR) Filtering	50
Signal and Audio: Hard Limiting	51
Signal and Audio: Infinite Impulse Response (IIR) Filtering	52
Signal and Audio: LMS Adaptive Filtering	53
Signal and Audio: Linear Predictive Coding	53
Signal and Audio: Multiply	56
Signal and Audio: Pre-emphasizing	57
Signal and Audio: Requantization	58
Signal and Audio: Resampling	58

Signal and Audio: Signal Generation	59
Signal and Audio: Windowing	60
<b>Video Functions and Syntax</b>	<b>63</b>
Video: Alpha-blending Functions	63
Video: Band-combine Functions	64
Video: Band-shuffling Functions	65
Video: Inter-picture Processing Functions —DCT	66
Video: Inter-picture Processing Functions —IDCT	67
Video: Inter-picture Processing Functions —Motion Compensation	67
Video: Inter-picture Processing Functions —Motion Estimation	70
Video: Intra-picture Processing Functions —Color Conversion	71
Video: Intra-picture Processing Functions —DCT	76
Video: Intra-picture Processing Functions —IDCT	76
Video: JPEG File Interchange Format (JFIF) Processing Functions	76
Video: Resizing Functions	79
Video: Wavelet Color Space Conversion Functions	79
Video: Wavelet Sign-magnitude Conversion Functions	80

Video: Wavelet Transformation Functions—Forward	81
Video: Wavelet Transformation Functions—Inverse	81
Video: YUV Subsampling Functions	82
<b>Graphics Functions and Syntax</b>	<b>83</b>
Graphics: Circle and Ellipse Functions	83
Graphics: Fill Functions	84
Graphics: Point, Line, and Arc Functions	85
Graphics: Polygon Functions	90
<b>Volume Imaging Functions and Syntax</b>	<b>99</b>
Volume Imaging: Maximum Intensity Searching Functions	99
Volume Imaging: Ray Casting Functions	100
Volume Imaging: Window-level Operation Functions	100
<b>Reader Comments</b>	<b>101</b>

# mediaLib™ Quick Reference

---

## USAGE NOTES

Functions are grouped by category and are listed in alphabetic order within the category.

Each function is displayed in bold typeface, followed by the syntax in regular typeface.

The syntax for all functions is preceded by “mlib\_status ” unless otherwise indicated in a footnote.

Some APIs are in a condensed form. "*datatype* \*" is used to represent the data types of the input and output data arrays. The actual data types of input and output data arrays in each individual function should match the input and output data type indicated in the function name.

## IMAGING FUNCTIONS AND SYNTAX

---

### *Imaging: Arithmetic*

---

```
mlib_ImageAbs(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageAbs_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageAbs_Fp_Inp(mlib_image *srcdst)
```

```
mlib_ImageAbs_Inp(mlib_image *srcdst)
```

```
mlib_ImageAdd(mlib_image *dst, mlib_image *src1, mlib_image *src2)
```

```
mlib_ImageAdd_Fp(mlib_image *dst, mlib_image *src1, mlib_image *src2)
```

```
mlib_ImageAdd_Fp_Inp(mlib_image *srcldst, mlib_image *src2)
```

```
mlib_ImageAdd_Inp(mlib_image *srcldst, mlib_image *src2)
```

```
mlib_ImageAve(mlib_image *dst, mlib_image *src1, mlib_image *src2)
```

```
mlib_ImageAve_Fp(mlib_image *dst, mlib_image *src1, mlib_image *src2)
```

```
mlib_ImageAve_Fp_Inp(mlib_image *srcldst, mlib_image *src2)
```

```
mlib_ImageAve_Inp(mlib_image *srcldst, mlib_image *src2)
```

```
mlib_ImageBlend(mlib_image *dst, mlib_image *src1, mlib_image *src2, mlib_image *alpha)
```

```
mlib_ImageBlend_Fp(mlib_image *dst, mlib_image *src1, mlib_image *src2, mlib_image *alpha)
```

```
mlib_ImageBlend1_Fp_Inp(mlib_image *srcldst, mlib_image *src2, mlib_image *alpha)
```

---

## *Imaging: Arithmetic (Continued)*

---

```
mlib_ImageBlend1_Inp(mlib_image *src1dst, mlib_image *src2, mlib_image *alpha)
mlib_ImageBlend2_Fp_Inp(mlib_image *src2dst, mlib_image *src1, mlib_image *alpha)
mlib_ImageBlend2_Inp(mlib_image *src2dst, mlib_image *src1, mlib_image *alpha)
mlib_ImageConstAdd(mlib_image *dst, mlib_image *src, mlib_s32 *c)
mlib_ImageConstAdd_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *c)
mlib_ImageConstAdd_Fp_Inp(mlib_image *srcdst, mlib_d64 *c)
mlib_ImageConstAdd_Inp(mlib_image *srcdst, mlib_s32 *c)
mlib_ImageConstDiv(mlib_image *dst, mlib_image *src, mlib_d64 *c)
mlib_ImageConstDiv_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *c)
mlib_ImageConstDiv_Fp_Inp(mlib_image *srcdst, mlib_d64 *c)
mlib_ImageConstDiv_Inp(mlib_image *srcdst, mlib_d64 *c)
mlib_ImageConstDivShift(mlib_image *dst, mlib_image *src, mlib_s32 *c, mlib_s32 shift)
mlib_ImageConstDivShift_Inp(mlib_image *srcdst, mlib_s32 *c, mlib_s32 shift)
mlib_ImageConstMul(mlib_image *dst, mlib_image *src, mlib_d64 *c)
mlib_ImageConstMul_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *c)
mlib_ImageConstMul_Fp_Inp(mlib_image *srcdst, mlib_d64 *c)
mlib_ImageConstMul_Inp(mlib_image *srcdst, mlib_d64 *c)
```

---

### *Imaging: Arithmetic (Continued)*

---

**mlib\_ImageConstMulShift**(mlib\_image \*dst, mlib\_image \*src, mlib\_s32 \*c, mlib\_s32 shift)

**mlib\_ImageConstMulShift\_Inp**(mlib\_image \*srcdst, mlib\_s32 \*c, mlib\_s32 shift)

**mlib\_ImageConstSub**(mlib\_image \*dst, mlib\_image \*src, mlib\_s32 \*c)

**mlib\_ImageConstSub\_Fp**(mlib\_image \*dst, mlib\_image \*src, mlib\_d64 \*c)

**mlib\_ImageConstSub\_Fp\_Inp**(mlib\_image \*srcdst, mlib\_d64 \*c)

**mlib\_ImageConstSub\_Inp**(mlib\_image \*srcdst, mlib\_s32 \*c)

**mlib\_ImageDivAlpha**(mlib\_image \*dst, mlib\_image \*src, mlib\_s32 cmask)

**mlib\_ImageDivAlpha\_Fp**(mlib\_image \*dst, mlib\_image \*src, mlib\_s32 cmask)

**mlib\_ImageDivAlpha\_Fp\_Inp**(mlib\_image \*srcdst, mlib\_s32 cmask)

**mlib\_ImageDivAlpha\_Inp**(mlib\_image \*srcdst, mlib\_s32 cmask)

**mlib\_ImageDivConstShift**(mlib\_image \*dst, mlib\_image \*src, mlib\_s32 \*c, mlib\_s32 shift)

**mlib\_ImageDivConstShift\_Inp**(mlib\_image \*srcdst, mlib\_s32 \*c, mlib\_s32 shift)

**mlib\_ImageDiv\_Fp**(mlib\_image \*dst, mlib\_image \*src1, mlib\_image \*src2)

**mlib\_ImageDiv1\_Fp\_Inp**(mlib\_image \*src1dst, mlib\_image \*src2)

**mlib\_ImageDiv2\_Fp\_Inp**(mlib\_image \*src2dst, mlib\_image \*src1)

**mlib\_ImageDivShift**(mlib\_image \*dst, mlib\_image \*src1, mlib\_image \*src2, mlib\_s32 shift)

**mlib\_ImageDivShift1\_Inp**(mlib\_image \*src1dst, mlib\_image \*src2, mlib\_s32 shift)

---

## *Imaging: Arithmetic (Continued)*

---

`mlib_ImageDivShift2_Inp(mlib_image *src2dst, mlib_image *src1, mlib_s32 shift)`

`mlib_ImageExp(mlib_image *dst, mlib_image *src)`

`mlib_ImageExp_Fp(mlib_image *dst, mlib_image *src)`

`mlib_ImageExp_Fp_Inp(mlib_image *srcdst)`

`mlib_ImageExp_Inp(mlib_image *srcdst)`

`mlib_ImageInvert(mlib_image *dst, mlib_image *src)`

`mlib_ImageInvert_Fp(mlib_image *dst, mlib_image *src)`

`mlib_ImageInvert_Fp_Inp(mlib_image *srcdst)`

`mlib_ImageInvert_Inp(mlib_image *srcdst)`

`mlib_ImageLog(mlib_image *dst, mlib_image *src)`

`mlib_ImageLog_Fp(mlib_image *dst, mlib_image *src)`

`mlib_ImageLog_Fp_Inp(mlib_image *srcdst)`

`mlib_ImageLog_Inp(mlib_image *srcdst)`

`mlib_ImageMax(mlib_image *dst, mlib_image *src1, mlib_image *src2)`

`mlib_ImageMax_Fp(mlib_image *dst, mlib_image *src1, mlib_image *src2)`

`mlib_ImageMax_Fp_Inp(mlib_image *src1dst, mlib_image *src2)`

`mlib_ImageMax_Inp(mlib_image *src1dst, mlib_image *src2)`

---

*Imaging: Arithmetic (Continued)*

---

```
mllib_ImageMin(mllib_image *dst, mllib_image *src1, mllib_image *src2)
```

```
mllib_ImageMin_Fp(mllib_image *dst, mllib_image *src1, mllib_image *src2)
```

```
mllib_ImageMin_Fp_Inp(mllib_image *src1dst, mllib_image *src2)
```

```
mllib_ImageMin_Inp(mllib_image *src1dst, mllib_image *src2)
```

```
mllib_ImageMul_Fp(mllib_image *dst, mllib_image *src1, mllib_image *src2)
```

```
mllib_ImageMul_Fp_Inp(mllib_image *src1dst, mllib_image *src2)
```

```
mllib_ImageMulAlpha(mllib_image *dst, mllib_image *src, mllib_s32 cmask)
```

```
mllib_ImageMulAlpha_Fp(mllib_image *dst, mllib_image *src, mllib_s32 cmask)
```

```
mllib_ImageMulAlpha_Fp_Inp(mllib_image *srcdst, mllib_s32 cmask)
```

```
mllib_ImageMulAlpha_Inp(mllib_image *srcdst, mllib_s32 cmask)
```

```
mllib_ImageMulShift(mllib_image *dst, mllib_image *src1, mllib_image *src2, mllib_s32 shift)
```

```
mllib_ImageMulShift_Inp(mllib_image *src1dst, mllib_image *src2, mllib_s32 shift)
```

```
mllib_ImageScalarBlend(mllib_image *dst, mllib_image *src1, mllib_image *src2, mllib_s32 *alpha)
```

```
mllib_ImageScalarBlend_Fp(mllib_image *dst, mllib_image *src1, mllib_image *src2, mllib_d64 *alpha)
```

```
mllib_ImageScalarBlend_Fp_Inp(mllib_image *src1dst, mllib_image *src2, mllib_d64 *alpha)
```

```
mllib_ImageScalarBlend_Inp(mllib_image *src1dst, mllib_image *src2, mllib_s32 *alpha)
```

```
mllib_ImageScale(mllib_image *dst, mllib_image *src, mllib_s32 *alpha, mllib_s32 *beta, mllib_s32 shift)
```

---

## *Imaging: Arithmetic (Continued)*

---

```
mlib_ImageScale_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *alpha, mlib_d64 *beta)
mlib_ImageScale_Fp_Inp(mlib_image *srcdst, mlib_d64 *alpha, mlib_d64 *beta)
mlib_ImageScale_Inp(mlib_image *srcdst, mlib_s32 *alpha, mlib_s32 *beta, mlib_s32 shift)
mlib_ImageScale2(mlib_image *dst, mlib_image *src, mlib_d64 *alpha, mlib_d64 *beta)
mlib_ImageScale2_Inp(mlib_image *srcdst, mlib_d64 *alpha, mlib_d64 *beta)
mlib_ImageSqr_Fp(mlib_image *dst, mlib_image *src)
mlib_ImageSqr_Fp_Inp(mlib_image *srcdst)
mlib_ImageSqrShift(mlib_image *dst, mlib_image *src, mlib_s32 shift)
mlib_ImageSqrShift_Inp(mlib_image *srcdst, mlib_s32 shift)
mlib_ImageSub(mlib_image *dst, mlib_image *src1, mlib_image *src2)
mlib_ImageSub_Fp(mlib_image *dst, mlib_image *src1, mlib_image *src2)
mlib_ImageSub1_Fp_Inp(mlib_image *src1dst, mlib_image *src2)
mlib_ImageSub1_Inp(mlib_image *src1dst, mlib_image *src2)
mlib_ImageSub2_Fp_Inp(mlib_image *src2dst, mlib_image *src1)
mlib_ImageSub2_Inp(mlib_image *src2dst, mlib_image *src1)
```

---

---

### *Imaging: Fourier Transform*

---

```
mlib_ImageFourierTransform(mlib_image *dst, mlib_image *src, mlib_fourier_mode mode)
```

---

---

### *Imaging: Geometric*

---

```
mlib_ImageAffine(mlib_image *dst, mlib_image *src, mlib_d64 *mtx, mlib_filter filter, mlib_edge edge)
```

```
mlib_ImageAffine_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *mtx, mlib_filter filter, mlib_edge edge)
```

```
mlib_ImageAffineIndex(mlib_image *dst, mlib_image *src, mlib_d64 *mtx, mlib_filter filter, mlib_edge edge,  
    void *colormap)
```

```
mlib_ImageAffineTable(mlib_image *dst, mlib_image *src, mlib_d64 *mtx, void *interp_table, mlib_edge edge)
```

```
mlib_ImageAffineTable_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *mtx, void *interp_table, mlib_edge edge)
```

```
mlib_ImageAffineTransform(mlib_image *dst, mlib_image *src, mlib_d64 *mtx, mlib_filter filter, mlib_edge edge)
```

```
mlib_ImageAffineTransform_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *mtx, mlib_filter filter, mlib_edge edge)
```

```
mlib_ImageAffineTransformIndex(mlib_image *dst, mlib_image *src, mlib_d64 *mtx, mlib_filter filter, mlib_edge edge,  
    void *colormap)
```

```
mlib_ImageFilteredSubsample(mlib_image *dst, mlib_image *src, mlib_s32 scaleX, mlib_s32 scaleY, mlib_s32 transX,  
    mlib_s32 transY, mlib_d64 *hKernel, mlib_d64 *vKernel, mlib_s32 hSize, mlib_s32 vSize, mlib_s32 hParity, mlib_s32  
    vParity, mlib_edge edge)
```

---

## *Imaging: Geometric (Continued)*

---

```
mlib_ImageFilteredSubsample_Fp(mlib_image *dst, mlib_image *src, mlib_s32 scaleX, mlib_s32 scaleY, mlib_s32 transX,  
    mlib_s32 transY, mlib_d64 *hKernel, mlib_d64 *vKernel, mlib_s32 hSize, mlib_s32 vSize, mlib_s32 hParity, mlib_s32  
    vParity, mlib_edge edge)
```

```
mlib_ImageFlipAntiDiag(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageFlipAntiDiag_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageFlipMainDiag(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageFlipMainDiag_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageFlipX(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageFlipX_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageFlipY(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageFlipY_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageGridWarp(mlib_image *dst, mlib_image *src, mlib_f32 *xWarpPos, mlib_f32 *yWarpPos, mlib_d64 postShiftX,  
    mlib_d64 postShiftY, mlib_s32 xStart, mlib_s32 xStep, mlib_s32 xNumCells, mlib_s32 yStart, mlib_s32 yStep,  
    mlib_s32 yNumCells, mlib_filter filter, mlib_edge edge)
```

```
mlib_ImageGridWarp_Fp(mlib_image *dst, mlib_image *src, mlib_f32 *xWarpPos, mlib_f32 *yWarpPos, mlib_d64 postShiftX,  
    mlib_d64 postShiftY, mlib_s32 xStart, mlib_s32 xStep, mlib_s32 xNumCells, mlib_s32 yStart, mlib_s32 yStep,  
    mlib_s32 yNumCells, mlib_filter filter, mlib_edge edge)
```

---

### *Imaging: Geometric (Continued)*

---

```
mllib_ImageGridWarpTable(mllib_image *dst, mllib_image *src, mllib_f32 *xWarpPos, mllib_f32 *yWarpPos, mllib_d64 postShiftX,
    mllib_d64 postShiftY, mllib_s32 xStart, mllib_s32 xStep, mllib_s32 xNumCells, mllib_s32 yStart, mllib_s32 yStep, mllib_s32
    yNumCells, void *table, mllib_edge edge)
```

```
mllib_ImageGridWarpTable_Fp(mllib_image *dst, mllib_image *src, mllib_f32 *xWarpPos, mllib_f32 *yWarpPos, mllib_d64
    postShiftX, mllib_d64 postShiftY, mllib_s32 xStart, mllib_s32 xStep, mllib_s32 xNumCells, mllib_s32 yStart, mllib_s32
    yStep, mllib_s32 yNumCells, void *table, mllib_edge edge)
```

```
mllib_ImageInterpTableCreate(mllib_type type, mllib_s32 width, mllib_s32 height, mllib_s32 leftPadding, mllib_s32 topPadding,
    mllib_s32 subsampleBitsH, mllib_s32 subsampleBitsV, mllib_s32 precisionBits, void *dataH, void *dataV)1
```

```
mllib_ImageInterpTableDelete(void *interp_table)2
```

```
mllib_ImagePolynomialWarp(mllib_image *dst, mllib_image *src, mllib_d64 *xCoeffs, mllib_d64 *yCoeffs, mllib_s32 n,
    mllib_d64 preShiftX, mllib_d64 preShiftY, mllib_d64 postShiftX, mllib_d64 postShiftY, mllib_d64 preScaleX,
    mllib_d64 preScaleY, mllib_d64 postScaleX, mllib_d64 postScaleY, mllib_filter filter, mllib_edge edge)
```

```
mllib_ImagePolynomialWarp_Fp(mllib_image *dst, mllib_image *src, mllib_d64 *xCoeffs, mllib_d64 *yCoeffs, mllib_s32 n,
    mllib_d64 preShiftX, mllib_d64 preShiftY, mllib_d64 postShiftX, mllib_d64 postShiftY, mllib_d64 preScaleX,
    mllib_d64 preScaleY, mllib_d64 postScaleX, mllib_d64 postScaleY, mllib_filter filter, mllib_edge edge)
```

```
mllib_ImagePolynomialWarpTable(mllib_image *dst, mllib_image *src, mllib_d64 *xCoeffs, mllib_d64 *yCoeffs, mllib_s32 n,
    mllib_d64 preShiftX, mllib_d64 preShiftY, mllib_d64 postShiftX, mllib_d64 postShiftY, mllib_d64 preScaleX, mllib_d64
    preScaleY, mllib_d64 postScaleX, mllib_d64 postScaleY, void *interp_table, mllib_edge edge)
```

---

## *Imaging: Geometric (Continued)*

---

**mllib\_ImagePolynomialWarpTable\_Fp**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 \*xCoeffs, mllib\_d64 \*yCoeffs, mllib\_s32 n, mllib\_d64 preShiftX, mllib\_d64 preShiftY, mllib\_d64 postShiftX, mllib\_d64 postShiftY, mllib\_d64 preScaleX, mllib\_d64 preScaleY, mllib\_d64 postScaleX, mllib\_d64 postScaleY, void \*interp\_table, mllib\_edge edge)

**mllib\_ImageRotate**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 angle, mllib\_d64 xcenter, mllib\_d64 ycenter, mllib\_filter filter, mllib\_edge edge)

**mllib\_ImageRotate180**(mllib\_image \*dst, mllib\_image \*src)

**mllib\_ImageRotate180\_Fp**(mllib\_image \*dst, mllib\_image \*src)

**mllib\_ImageRotate270**(mllib\_image \*dst, mllib\_image \*src)

**mllib\_ImageRotate270\_Fp**(mllib\_image \*dst, mllib\_image \*src)

**mllib\_ImageRotate90**(mllib\_image \*dst, mllib\_image \*src)

**mllib\_ImageRotate90\_Fp**(mllib\_image \*dst, mllib\_image \*src)

**mllib\_ImageRotate\_Fp**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 angle, mllib\_d64 xcenter, mllib\_d64 ycenter, mllib\_filter filter, mllib\_edge edge)

**mllib\_ImageRotateIndex**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 angle, mllib\_d64 xcenter, mllib\_d64 ycenter, mllib\_filter filter, mllib\_edge edge, void \*colormap)

**mllib\_ImageSubsampleBinaryToGray**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 xscale, mllib\_d64 yscale, mllib\_u8 \*lutGray)

**mllib\_ImageZoom**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 zoomx, mllib\_d64 zoomy, mllib\_filter filter, mllib\_edge edge)

**mllib\_ImageZoom\_Fp**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 zoomx, mllib\_d64 zoomy, mllib\_filter filter, mllib\_edge edge)

---

*Imaging: Geometric (Continued)*

---

**mllib\_ImageZoomIn2X**(mllib\_image \*dst, mllib\_image \*src, mllib\_filter filter, mllib\_edge edge)

**mllib\_ImageZoomIn2X\_Fp**(mllib\_image \*dst, mllib\_image \*src, mllib\_filter filter, mllib\_edge edge)

**mllib\_ImageZoomIn2XIndex**(mllib\_image \*dst, mllib\_image \*src, mllib\_filter filter, mllib\_edge edge, void \*colormap)

**mllib\_ImageZoomIndex**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 zoomx, mllib\_d64 zoomy, mllib\_filter filter, mllib\_edge edge, void \*colormap)

**mllib\_ImageZoomOut2X**(mllib\_image \*dst, mllib\_image \*src, mllib\_filter filter, mllib\_edge edge)

**mllib\_ImageZoomOut2X\_Fp**(mllib\_image \*dst, mllib\_image \*src, mllib\_filter filter, mllib\_edge edge)

**mllib\_ImageZoomOut2XIndex**(mllib\_image \*dst, mllib\_image \*src, mllib\_filter filter, mllib\_edge edge, void \*colormap)

**mllib\_ImageZoomTranslate**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 zoomx, mllib\_d64 zoomy, mllib\_d64 tx, mllib\_d64 ty, mllib\_filter filter, mllib\_edge edge)

**mllib\_ImageZoomTranslate\_Fp**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 zoomx, mllib\_d64 zoomy, mllib\_d64 tx, mllib\_d64 ty, mllib\_filter filter, mllib\_edge edge)

**mllib\_ImageZoomTranslateTable**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 zoomx, mllib\_d64 zoomy, mllib\_d64 tx, mllib\_d64 ty, void \*interp\_table, mllib\_edge edge)

**mllib\_ImageZoomTranslateTable\_Fp**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 zoomx, mllib\_d64 zoomy, mllib\_d64 tx, mllib\_d64 ty, void \*interp\_table, mllib\_edge edge)

**mllib\_ImageZoomTranslateToGray**(mllib\_image \*dst, mllib\_image \*src, mllib\_d64 zoomx, mllib\_d64 zoomy, mllib\_d64 tx, mllib\_d64 ty, mllib\_filter filter, mllib\_edge edge, mllib\_s32 \*ghigh, mllib\_s32 \*glow)

---

1. This function requires the return type “void\*.”
2. This function requires the return type “void.”

---

### *Imaging: Image Composition*

---

```
mlib_ImageBlend_BSRC1_BSRC2(mlib_image dst, mlib_image src1, mlib_image src2, mlib_s32 cmask)
```

```
mlib_ImageBlend_BSRC1_BSRC2_Inp(mlib_image src1dst, mlib_image src2, mlib_s32 cmask)
```

```
mlib_ImageComposite(mlib_image dst, mlib_image src1, mlib_image src2, mlib_blend bsrc1, mlib_blend bsrc2,  
    mlib_s32 cmask)
```

```
mlib_ImageComposite_Inp(mlib_image src1dst, mlib_image src2, mlib_blend bsrc1, mlib_blend bsrc2, mlib_s32 cmask)
```

---

---

### *Imaging: Image Conversion Functions —Data Format Conversion*

---

```
mlib_ImageChannelCopy(mlib_image *dst, mlib_image *src, mlib_s32 cmask)
```

```
mlib_ImageChannelExtract(mlib_image *dst, mlib_image *src, mlib_s32 cmask)
```

```
mlib_ImageChannelInsert(mlib_image *dst, mlib_image *src, mlib_s32 cmask)
```

```
mlib_ImageChannelMerge(mlib_image *dst, mlib_image **srcs)
```

```
mlib_ImageChannelSplit(mlib_image **dsts, mlib_image *src)
```

```
mlib_ImageDataTypeConvert(mlib_image *dst, mlib_image *src)
```

---

***Imaging: Image Conversion Functions —Data Format Conversion (Continued)***

---

```
mlib_ImageReformat(void **dstData, void **srcData, mlib_s32 numBands, mlib_s32 xSize, mlib_s32 ySize,  
    mlib_type dstDataType, mlib_s32 *dstBandoffsets, mlib_s32 dstScanlinestride, mlib_s32 dstPixelstride,  
    mlib_type srcDataType, mlib_s32 *srcBandoffsets, mlib_s32 srcScanlinestride, mlib_s32 srcPixelstride)
```

---

---

***Imaging: Image Conversion Functions —General Color Space Conversion***

---

```
mlib_ImageColorConvert1(mlib_image *dst, mlib_image *src, mlib_d64 *cmat)  
mlib_ImageColorConvert1_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *cmat)  
mlib_ImageColorConvert2(mlib_image *dst, mlib_image *src, mlib_d64 *cmat, mlib_d64 *offset)  
mlib_ImageColorConvert2_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *cmat, mlib_d64 *offset)
```

---

---

***Imaging: Image Conversion Functions —RGB to Monochrome Color Space Conversion***

---

```
mlib_ImageColorRGB2CIEMono(mlib_image *dst, mlib_image *src)  
mlib_ImageColorRGB2CIEMono_Fp(mlib_image *dst, mlib_image *src)  
mlib_ImageColorRGB2Mono(mlib_image *dst, mlib_image *src, mlib_d64 *weight)  
mlib_ImageColorRGB2Mono_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *weight)
```

---

---

### *Imaging: Image Conversion Functions —RGB/HSL Color Space Conversion*

---

`mlib_ImageColorHSL2RGB(mlib_image *dst, mlib_image *src)`

`mlib_ImageColorHSL2RGB_Fp(mlib_image *dst, mlib_image *src)`

`mlib_ImageColorRGB2HSL(mlib_image *dst, mlib_image *src)`

`mlib_ImageColorRGB2HSL_Fp(mlib_image *dst, mlib_image *src)`

---

### *Imaging: Image Conversion Functions —RGB/HSV Color Space Conversion*

---

`mlib_ImageColorHSV2RGB(mlib_image *dst, mlib_image *src)`

`mlib_ImageColorHSV2RGB_Fp(mlib_image *dst, mlib_image *src)`

`mlib_ImageColorRGB2HSV(mlib_image *dst, mlib_image *src)`

`mlib_ImageColorRGB2HSV_Fp(mlib_image *dst, mlib_image *src)`

---

### *Imaging: Image Conversion Functions —RGB/XYZ Color Space Conversion*

---

`mlib_ImageColorRGB2XYZ(mlib_image *dst, mlib_image *src)`

`mlib_ImageColorRGB2XYZ_Fp(mlib_image *dst, mlib_image *src)`

`mlib_ImageColorXYZ2RGB(mlib_image *dst, mlib_image *src)`

`mlib_ImageColorXYZ2RGB_Fp(mlib_image *dst, mlib_image *src)`

---

---

### *Imaging: Image Conversion Functions —R'G'B'/Y'C<sub>B</sub>C<sub>R</sub> Color Space Conversion*

---

```
mlib_ImageColorRGB2YCC(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageColorRGB2YCC_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageColorYCC2RGB(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageColorYCC2RGB_Fp(mlib_image *dst, mlib_image *src)
```

---

### *Imaging: Image Conversion Functions —True Color to Indexed Color Conversion*

---

```
mlib_ImageColorErrorDiffusion3x3(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 scale, void *colormap)
```

```
mlib_ImageColorOrderedDither8x8(mlib_image *dst, mlib_image *src, mlib_s32 *dmask, mlib_s32 scale, void *colormap)
```

```
mlib_ImageColorTrue2Index(mlib_image *dst, mlib_image *src, void *colormap)
```

```
mlib_ImageColorTrue2IndexFree(void *colormap)1
```

```
mlib_ImageColorTrue2IndexInit(void **colormap, mlib_s32 bits, mlib_type intype, mlib_type outtype, mlib_s32 channels,
    mlib_s32 entries, mlib_s32 offset, void **table)
```

---

1. This function requires the return type “void.”

---

## *Imaging: Image Copying and Clearing*

---

**mlib\_ImageClear**(mlib\_image \*img, mlib\_s32 \*color)

**mlib\_ImageClear\_Fp**(mlib\_image \*img, mlib\_d64 \*color)

**mlib\_ImageClearEdge**(mlib\_image \*img, mlib\_s32 dx, mlib\_s32 dy, mlib\_s32 \*color)

**mlib\_ImageClearEdge\_Fp**(mlib\_image \*img, mlib\_s32 dx, mlib\_s32 dy, mlib\_d64 \*color)

**mlib\_ImageCopy**(mlib\_image \*dst, mlib\_image \*src)

**mlib\_ImageCopyArea**(mlib\_image \*img, mlib\_s32 x, mlib\_s32 y, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dx, mlib\_s32 dy)

**mlib\_ImageCopyMask**(mlib\_image \*dst, mlib\_image \*src, mlib\_image \*mask, mlib\_s32 \*thresh)

**mlib\_ImageCopyMask\_Fp**(mlib\_image \*dst, mlib\_image \*src, mlib\_image \*mask, mlib\_d64 \*thresh)

**mlib\_ImageCopySubimage**(mlib\_image \*dst, mlib\_image \*src, mlib\_s32 xd, mlib\_s32 yd, mlib\_s32 xs, mlib\_s32 ys,  
mlib\_s32 w, mlib\_s32 h)

---

---

### *Imaging: Image Edge Detection*

---

```
mlib_ImageGradient3x3(mlib_image *dst, mlib_image *src, mlib_d64 *hmask, mlib_d64 *vmask, mlib_s32 cmask,  
    mlib_edge edge)
```

```
mlib_ImageGradient3x3_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *hmask, mlib_d64 *vmask, mlib_s32 cmask,  
    mlib_edge edge)
```

```
mlib_ImageGradientMxN(mlib_image *dst, mlib_image *src, mlib_d64 *hmask, mlib_d64 *vmask, mlib_s32 m, mlib_s32 n,  
    mlib_s32 dm, mlib_s32 dn, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageGradientMxN_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *hmask, mlib_d64 *vmask, mlib_s32 m, mlib_s32 n,  
    mlib_s32 dm, mlib_s32 dn, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageSobel(mlib_image *dst, mlib_image *src, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageSobel_Fp(mlib_image *dst, mlib_image *src, mlib_s32 cmask, mlib_edge edge)
```

---

### *Imaging: Image Initialization Functions —Get Functions*

---

```
mlib_ImageGetBitOffset(mlib_image *img)1
```

```
mlib_ImageGetBorders(mlib_image *img)2
```

```
mlib_ImageGetChannels(mlib_image *img)1
```

```
mlib_ImageGetData(mlib_image *img)3
```

---

## *Imaging: Image Initialization Functions —Get Functions (Continued)*

---

**mllib\_ImageGetFlags**(mllib\_image \*img)<sup>1</sup>

**mllib\_ImageGetHeight**(mllib\_image \*img)<sup>1</sup>

**mllib\_ImageGetStride**(mllib\_image \*img)<sup>1</sup>

**mllib\_ImageGetType**(mllib\_image \*img)<sup>4</sup>

**mllib\_ImageGetWidth**(mllib\_image \*img)<sup>1</sup>

**mllib\_ImageIsNotAligned2**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageIsNotAligned4**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageIsNotAligned8**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageIsNotAligned64**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageIsNotHeight2X**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageIsNotHeight4X**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageIsNotHeight8X**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageIsNotOneDvector**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageIsNotStride8X**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageIsNotWidth2X**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageIsNotWidth4X**(mllib\_image \*img)<sup>5</sup>

---

### *Imaging: Image Initialization Functions —Get Functions (Continued)*

---

**mllib\_ImageIsNotWidth8X**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageIsUserAllocated**(mllib\_image \*img)<sup>5</sup>

**mllib\_ImageTestFlags**(mllib\_image \*img, mllib\_s32 flags)<sup>5</sup>

---

1. This function requires the return type “mllib\_s32.”
2. This function requires the return type “mllib\_s32\*.”
3. This function requires the return type “void\*.”
4. This function requires the return type “mllib\_type.”
5. This function requires the return type “int.”

---

### *Imaging: Image Initialization Functions —Image Structure Creation and Deletion*

---

**mllib\_ImageCreate**(mllib\_type type, mllib\_s32 channels, mllib\_s32 width, mllib\_s32 height)<sup>1</sup>

**mllib\_ImageCreateStruct**(mllib\_type type, mllib\_s32 channels, mllib\_s32 width, mllib\_s32 height, mllib\_s32 stride, void \*data)<sup>1</sup>

**mllib\_ImageCreateSubimage**(mllib\_image \*img, mllib\_s32 x, mllib\_s32 y, mllib\_s32 w, mllib\_s32 h)<sup>1</sup>

**mllib\_ImageDelete**(mllib\_image \*img)<sup>2</sup>

**mllib\_ImageSetBorders**(mllib\_image \*img, mllib\_s32 minX, mllib\_s32 minY, mllib\_s32 maxX, mllib\_s32 maxY)

---

1. This function requires the return type “mllib\_image\*.”
2. This function requires the return type “void.”

---

## *Imaging: Image Statistics*

---

**mllib\_ImageAutoCorrel**(mllib\_d64 \*correl, mllib\_image \*img, mllib\_s32 dx, mllib\_s32 dy)

**mllib\_ImageAutoCorrel\_Fp**(mllib\_d64 \*correl, mllib\_image \*img, mllib\_s32 dx, mllib\_s32 dy)

**mllib\_ImageCrossCorrel**(mllib\_d64 \*correl, mllib\_image \*img1, mllib\_image \*img2)

**mllib\_ImageCrossCorrel\_Fp**(mllib\_d64 \*correl, mllib\_image \*img1, mllib\_image \*img2)

**mllib\_ImageExtrema2**(mllib\_s32 \*min, mllib\_s32 \*max, mllib\_image \*img, mllib\_s32 xStart, mllib\_s32 yStart, mllib\_s32 xPeriod, mllib\_s32 yPeriod)

**mllib\_ImageExtrema2\_Fp**(mllib\_d64 \*min, mllib\_d64 \*max, mllib\_image \*img, mllib\_s32 xStart, mllib\_s32 yStart, mllib\_s32 xPeriod, mllib\_s32 yPeriod)

**mllib\_ImageHistogram**(mllib\_s32 \*\*histo, mllib\_image \*img)

**mllib\_ImageHistogram2**(mllib\_s32 \*\*histo, mllib\_image \*img, mllib\_s32 \*numBins, mllib\_s32 \*lowValue, mllib\_s32 \*highValue, mllib\_s32 xStart, mllib\_s32 yStart, mllib\_s32 xPeriod, mllib\_s32 yPeriod)

**mllib\_ImageMaximum**(mllib\_s32 \*max, mllib\_image \*img)

**mllib\_ImageMaximum\_Fp**(mllib\_d64 \*max, mllib\_image \*img)

**mllib\_ImageMean**(mllib\_d64 \*mean, mllib\_image \*img)

**mllib\_ImageMean\_Fp**(mllib\_d64 \*mean, mllib\_image \*img)

**mllib\_ImageMinimum**(mllib\_s32 \*min, mllib\_image \*img)

---

### *Imaging: Image Statistics (Continued)*

---

```
mlib_ImageMinimum_Fp(mlib_d64 *min, mlib_image *img)
mlib_ImageMoment2(mlib_d64 *moment, mlib_image *img)
mlib_ImageMoment2_Fp(mlib_d64 *moment, mlib_image *img)
mlib_ImageStdDev(mlib_d64 *sdev, mlib_image *img, mlib_d64 *mean)
mlib_ImageStdDev_Fp(mlib_d64 *sdev, mlib_image *img, mlib_d64 *mean)
mlib_ImageXProj(mlib_d64 *xproj, mlib_image *img)
mlib_ImageXProj_Fp(mlib_d64 *xproj, mlib_image *img)
mlib_ImageYProj(mlib_d64 *yproj, mlib_image *img)
mlib_ImageYProj_Fp(mlib_d64 *yproj, mlib_image *img)
```

---

---

### *Imaging: Linear Spatial*

---

```
mlib_ImageConv2x2(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 scale, mlib_s32 cmask, mlib_edge edge)
mlib_ImageConv2x2_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *kernel, mlib_s32 cmask, mlib_edge edge)
mlib_ImageConv2x2Index(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 scale, mlib_edge edge,
    void *colormap)
mlib_ImageConv3x3(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 scale, mlib_s32 cmask, mlib_edge edge)
mlib_ImageConv3x3_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *kernel, mlib_s32 cmask, mlib_edge edge)
```

---

## *Imaging: Linear Spatial (Continued)*

---

```
mlib_ImageConv3x3Index(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 scale, mlib_edge edge,  
void *colormap)
```

```
mlib_ImageConv4x4(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 scale, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageConv4x4_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *kernel, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageConv4x4Index(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 scale, mlib_edge edge,  
void *colormap)
```

```
mlib_ImageConv5x5(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 scale, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageConv5x5_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *kernel, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageConv5x5Index(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 scale, mlib_edge edge,  
void *colormap)
```

```
mlib_ImageConv7x7(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 scale, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageConv7x7_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *kernel, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageConv7x7Index(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 scale, mlib_edge edge,  
void *colormap)
```

```
mlib_ImageConvKernelConvert(mlib_s32 *ikernel, mlib_s32 *iscale, mlib_d64 *fkernel, mlib_s32 m, mlib_s32 n,  
mlib_type type)
```

```
mlib_ImageConvMxN(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 m, mlib_s32 n, mlib_s32 dm, mlib_s32 dn,  
mlib_s32 scale, mlib_s32 cmask, mlib_edge edge)
```

---

### *Imaging: Linear Spatial (Continued)*

---

```
mlib_ImageConvMxN_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *kernel, mlib_s32 m, mlib_s32 n, mlib_s32 dm,  
    mlib_s32 dn, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageConvMxNIndex(mlib_image *dst, mlib_image *src, mlib_s32 *kernel, mlib_s32 m, mlib_s32 n, mlib_s32 dm,  
    mlib_s32 dn, mlib_s32 scale, mlib_edge edge, void *colormap)
```

```
mlib_ImageConvolveMxN(mlib_image *dst, mlib_image *src, mlib_d64 *kernel, mlib_s32 m, mlib_s32 n, mlib_s32 dm,  
    mlib_s32 dn, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageConvolveMxN_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *kernel, mlib_s32 m, mlib_s32 n, mlib_s32 dm,  
    mlib_s32 dn, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageSConv3x3(mlib_image *dst, mlib_image *src, mlib_s32 *hkernel, mlib_s32 *vkernel, mlib_s32 scale,  
    mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageSConv3x3_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *hkernel, mlib_d64 *vkernel, mlib_s32 cmask,  
    mlib_edge edge)
```

```
mlib_ImageSConv5x5(mlib_image *dst, mlib_image *src, mlib_s32 *hkernel, mlib_s32 *vkernel, mlib_s32 scale,  
    mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageSConv5x5_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *hkernel, mlib_d64 *vkernel, mlib_s32 cmask,  
    mlib_edge edge)
```

---

## *Imaging: Linear Spatial (Continued)*

---

**mlib\_ImageSConv7x7**(mlib\_image \*dst, mlib\_image \*src, mlib\_s32 \*hkernel, mlib\_s32 \*vkernel, mlib\_s32 scale, mlib\_s32 cmask, mlib\_edge edge)

**mlib\_ImageSConv7x7\_Fp**(mlib\_image \*dst, mlib\_image \*src, mlib\_d64 \*hkernel, mlib\_d64 \*vkernel, mlib\_s32 cmask, mlib\_edge edge)

**mlib\_ImageSConvKernelConvert**(mlib\_s32 \*ihkernel, mlib\_s32 \*ivkernel, mlib\_s32 \*iscale, mlib\_d64 \*fhkernel, mlib\_d64 \*fvkernel, mlib\_s32 m, mlib\_s32 n, mlib\_type type)

---

---

## *Imaging: Logical*

---

**mlib\_ImageAnd**(mlib\_image \*dst, mlib\_image \*src1, mlib\_image \*src2)

**mlib\_ImageAnd\_Inp**(mlib\_image \*src1dst, mlib\_image \*src2)

**mlib\_ImageAndNot**(mlib\_image \*dst, mlib\_image \*src1, mlib\_image \*src2)

**mlib\_ImageAndNot1\_Inp**(mlib\_image \*src1dst, mlib\_image \*src2)

**mlib\_ImageAndNot2\_Inp**(mlib\_image \*src2dst, mlib\_image \*src1)

**mlib\_ImageConstAnd**(mlib\_image \*dst, mlib\_image \*src, mlib\_s32 \*c)

**mlib\_ImageConstAnd\_Inp**(mlib\_image \*srcdst, mlib\_s32 \*c)

**mlib\_ImageConstAndNot**(mlib\_image \*dst, mlib\_image \*src, mlib\_s32 \*c)

**mlib\_ImageConstAndNot\_Inp**(mlib\_image \*srcdst, mlib\_s32 \*c)

---

---

*Imaging: Logical (Continued)*

---

```
mlib_ImageConstNotAnd(mlib_image *dst, mlib_image *src, mlib_s32 *c)
mlib_ImageConstNotAnd_Inp(mlib_image *srcdst, mlib_s32 *c)
mlib_ImageConstNotOr(mlib_image *dst, mlib_image *src, mlib_s32 *c)
mlib_ImageConstNotOr_Inp(mlib_image *srcdst, mlib_s32 *c)
mlib_ImageConstNotXor(mlib_image *dst, mlib_image *src, mlib_s32 *c)
mlib_ImageConstNotXor_Inp(mlib_image *srcdst, mlib_s32 *c)
mlib_ImageConstOr(mlib_image *dst, mlib_image *src, mlib_s32 *c)
mlib_ImageConstOr_Inp(mlib_image *srcdst, mlib_s32 *c)
mlib_ImageConstOrNot(mlib_image *dst, mlib_image *src, mlib_s32 *c)
mlib_ImageConstOrNot_Inp(mlib_image *srcdst, mlib_s32 *c)
mlib_ImageConstXor(mlib_image *dst, mlib_image *src, mlib_s32 *c)
mlib_ImageConstXor_Inp(mlib_image *srcdst, mlib_s32 *c)
mlib_ImageNot(mlib_image *dst, mlib_image *src)
mlib_ImageNot_Inp(mlib_image *srcdst)
mlib_ImageNotAnd(mlib_image *dst, mlib_image *src1, mlib_image *src2)
mlib_ImageNotAnd_Inp(mlib_image *src1dst, mlib_image *src2)
mlib_ImageNotOr(mlib_image *dst, mlib_image *src1, mlib_image *src2)
```

---

## *Imaging: Logical (Continued)*

---

```
mllib_ImageNotOr_Inp(mllib_image *src1dst, mllib_image *src2)
mllib_ImageNotXor(mllib_image *dst, mllib_image *src1, mllib_image *src2)
mllib_ImageNotXor_Inp(mllib_image *src1dst, mllib_image *src2)
mllib_ImageOr(mllib_image *dst, mllib_image *src1, mllib_image *src2)
mllib_ImageOr_Inp(mllib_image *src1dst, mllib_image *src2)
mllib_ImageOrNot(mllib_image *dst, mllib_image *src1, mllib_image *src2)
mllib_ImageOrNot1_Inp(mllib_image *src1dst, mllib_image *src2)
mllib_ImageOrNot2_Inp(mllib_image *src2dst, mllib_image *src1)
mllib_ImageXor(mllib_image *dst, mllib_image *src1, mllib_image *src2)
mllib_ImageXor_Inp(mllib_image *src1dst, mllib_image *src2)
```

---

---

## *Imaging: Morphological*

---

```
mllib_ImageDilate4(mllib_image *dst, mllib_image *src)
mllib_ImageDilate4_Fp(mllib_image *dst, mllib_image *src)
mllib_ImageDilate8(mllib_image *dst, mllib_image *src)
mllib_ImageDilate8_Fp(mllib_image *dst, mllib_image *src)
mllib_ImageErode4(mllib_image *dst, mllib_image *src)
```

---

*Imaging: Morphological (Continued)*

---

```
mlib_ImageErode4_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageErode8(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageErode8_Fp(mlib_image *dst, mlib_image *src)
```

---

*Imaging: Nonlinear Spatial*

---

```
mlib_ImageMaxFilter3x3(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageMaxFilter3x3_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageMaxFilter5x5(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageMaxFilter5x5_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageMaxFilter7x7(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageMaxFilter7x7_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageMedianFilter3x3(mlib_image *dst, mlib_image *src, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageMedianFilter3x3_Fp(mlib_image *dst, mlib_image *src, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageMedianFilter3x3_US(mlib_image *dst, mlib_image *src, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge,  
    mlib_s32 bits)
```

```
mlib_ImageMedianFilter5x5(mlib_image *dst, mlib_image *src, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageMedianFilter5x5_Fp(mlib_image *dst, mlib_image *src, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge)
```

---

## *Imaging: Nonlinear Spatial (Continued)*

---

```
mlib_ImageMedianFilter5x5_US(mlib_image *dst, mlib_image *src, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge, mlib_s32 bits)
```

```
mlib_ImageMedianFilter7x7(mlib_image *dst, mlib_image *src, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageMedianFilter7x7_Fp(mlib_image *dst, mlib_image *src, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageMedianFilter7x7_US(mlib_image *dst, mlib_image *src, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge, mlib_s32 bits)
```

```
mlib_ImageMedianFilterMxN(mlib_image *dst, mlib_image *src, mlib_s32 m, mlib_s32 n, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageMedianFilterMxN_Fp(mlib_image *dst, mlib_image *src, mlib_s32 m, mlib_s32 n, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge)
```

```
mlib_ImageMedianFilterMxN_US(mlib_image *dst, mlib_image *src, mlib_s32 m, mlib_s32 n, mlib_median_mask mmask, mlib_s32 cmask, mlib_edge edge, mlib_s32 bits)
```

```
mlib_ImageMinFilter3x3(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageMinFilter3x3_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageMinFilter5x5(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageMinFilter5x5_Fp(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageMinFilter7x7(mlib_image *dst, mlib_image *src)
```

```
mlib_ImageMinFilter7x7_Fp(mlib_image *dst, mlib_image *src)
```

---

### *Imaging: Nonlinear Spatial (Continued)*

---

```
mlib_ImageRankFilter3x3(mlib_image *dst, mlib_image *src, mlib_s32 rank)
mlib_ImageRankFilter3x3_Fp(mlib_image *dst, mlib_image *src, mlib_s32 rank)
mlib_ImageRankFilter3x3_US(mlib_image *dst, mlib_image *src, mlib_s32 rank, mlib_s32 bits)
mlib_ImageRankFilter5x5(mlib_image *dst, mlib_image *src, mlib_s32 rank)
mlib_ImageRankFilter5x5_Fp(mlib_image *dst, mlib_image *src, mlib_s32 rank)
mlib_ImageRankFilter5x5_US(mlib_image *dst, mlib_image *src, mlib_s32 rank, mlib_s32 bits)
mlib_ImageRankFilter7x7(mlib_image *dst, mlib_image *src, mlib_s32 rank)
mlib_ImageRankFilter7x7_Fp(mlib_image *dst, mlib_image *src, mlib_s32 rank)
mlib_ImageRankFilter7x7_US(mlib_image *dst, mlib_image *src, mlib_s32 rank, mlib_s32 bits)
mlib_ImageRankFilterMxN(mlib_image *dst, mlib_image *src, mlib_s32 m, mlib_s32 n, mlib_s32 rank)
mlib_ImageRankFilterMxN_Fp(mlib_image *dst, mlib_image *src, mlib_s32 m, mlib_s32 n, mlib_s32 rank)
mlib_ImageRankFilterMxN_US(mlib_image *dst, mlib_image *src, mlib_s32 m, mlib_s32 n, mlib_s32 rank, mlib_s32 bits)
```

---

### *Imaging: Radiometric*

---

```
mlib_ImageLookUp(mlib_image *dst, mlib_image *src, void **table)
mlib_ImageLookUp2(mlib_image *dst, mlib_image *src, void **table, mlib_s32 *offsets, mlib_s32 channels)
mlib_ImageLookUpMask(mlib_image *dst, mlib_image *src, void **table, mlib_s32 channels, mlib_s32 cmask)
```

---

## *Imaging: Radiometric (Continued)*

---

`mlib_ImageLookUp_Inp(mlib_image *srcdst, void **table)`

`mlib_ImageThresh1(mlib_image *dst, mlib_image *src, mlib_s32 *thresh, mlib_s32 *ghigh, mlib_s32 *glow)`

`mlib_ImageThresh1_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *thresh, mlib_d64 *ghigh, mlib_d64 *glow)`

`mlib_ImageThresh1_Fp_Inp(mlib_image *srcdst, mlib_d64 *thresh, mlib_d64 *ghigh, mlib_d64 *glow)`

`mlib_ImageThresh1_Inp(mlib_image *srcdst, mlib_s32 *thresh, mlib_s32 *ghigh, mlib_s32 *glow)`

`mlib_ImageThresh2(mlib_image *dst, mlib_image *src, mlib_s32 *thresh, mlib_s32 *glow)`

`mlib_ImageThresh2_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *thresh, mlib_d64 *glow)`

`mlib_ImageThresh2_Fp_Inp(mlib_image *srcdst, mlib_d64 *thresh, mlib_d64 *glow)`

`mlib_ImageThresh2_Inp(mlib_image *srcdst, mlib_s32 *thresh, mlib_s32 *glow)`

`mlib_ImageThresh3(mlib_image *dst, mlib_image *src, mlib_s32 *thresh, mlib_s32 *ghigh)`

`mlib_ImageThresh3_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *thresh, mlib_d64 *ghigh)`

`mlib_ImageThresh3_Fp_Inp(mlib_image *srcdst, mlib_d64 *thresh, mlib_d64 *ghigh)`

`mlib_ImageThresh3_Inp(mlib_image *srcdst, mlib_s32 *thresh, mlib_s32 *ghigh)`

`mlib_ImageThresh4(mlib_image *dst, mlib_image *src, mlib_s32 *thigh, mlib_s32 *tlow, mlib_s32 *ghigh, mlib_s32 *glow)`

`mlib_ImageThresh4_Fp(mlib_image *dst, mlib_image *src, mlib_d64 *thigh, mlib_d64 *tlow, mlib_d64 *ghigh,  
mlib_d64 *glow)`

`mlib_ImageThresh4_Fp_Inp(mlib_image *srcdst, mlib_d64 *thigh, mlib_d64 *tlow, mlib_d64 *ghigh, mlib_d64 *glow)`

---

### *Imaging: Radiometric (Continued)*

`mlib_ImageThresh4_Inp`(`mlib_image *srcdst`, `mlib_s32 *thigh`, `mlib_s32 *tlow`, `mlib_s32 *ghigh`, `mlib_s32 *glow`)

`mlib_ImageThresh5`(`mlib_image *dst`, `mlib_image *src`, `mlib_s32 *thigh`, `mlib_s32 *tlow`, `mlib_s32 *gray`)

`mlib_ImageThresh5_Fp`(`mlib_image *dst`, `mlib_image *src`, `mlib_d64 *thigh`, `mlib_d64 *tlow`, `mlib_d64 *gray`)

`mlib_ImageThresh5_Fp_Inp`(`mlib_image *srcdst`, `mlib_d64 *thigh`, `mlib_d64 *tlow`, `mlib_d64 *gray`)

`mlib_ImageThresh5_Inp`(`mlib_image *srcdst`, `mlib_s32 *thigh`, `mlib_s32 *tlow`, `mlib_s32 *gray`)

---

## LINEAR ALGEBRA FUNCTIONS AND SYNTAX

---

### *Linear Algebra: Data Arrangement Functions*

`mlib_VectorConvert_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod]`(`datatype *z`,  
`datatype *x`, `mlib_s32 n`)

`mlib_VectorCopy_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]`(`datatype *z`, `datatype *x`, `mlib_s32 n`)

`mlib_VectorMerge_[U8C|S8C|S16C|S32C]_[U8|S8|S16|S32]`(`datatype *z`, `datatype *r`, `datatype *i`, `mlib_s32 n`)

`mlib_Vectorsplit_[U8|S8|S16|S32]_[U8C|S8C|S16C|S32C]`(`datatype *r`, `datatype *i`, `datatype *x`, `mlib_s32 n`)

---

## *Linear Algebra: Matrix Functions*

```
mlib_MatrixAdd_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *y, mlib_s32 m, mlib_s32 n)
mlib_MatrixAdd_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
    datatype *y, mlib_s32 m, mlib_s32 n)
mlib_MatrixAddS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *c, mlib_s32 m, mlib_s32 n)
mlib_MatrixAddS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
    datatype *c, mlib_s32 m, mlib_s32 n)
mlib_MatrixMul_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
    datatype *y, mlib_s32 m, mlib_s32 l, mlib_s32 n)
mlib_MatrixMulS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *c, mlib_s32 m, mlib_s32 n)
mlib_MatrixMulS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
    datatype *c, mlib_s32 m, mlib_s32 n)
mlib_MatrixScale_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *a, datatype *b, mlib_s32 m,
    mlib_s32 n)
mlib_MatrixScale_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z,
    datatype *x, datatype *a, datatype *b, mlib_s32 m, mlib_s32 n)
mlib_MatrixSub_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *y, mlib_s32 m, mlib_s32 n)
```

---

### *Linear Algebra: Matrix Functions (Continued)*

---

```
mllib_MatrixSub_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z,
  datatype *x,datatype *y, mlib_s32 m, mlib_s32 n)
```

```
mllib_MatrixSubS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *c, mlib_s32 m, mlib_s32 n)
```

```
mllib_MatrixSubS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
  datatype *c, mlib_s32 m, mlib_s32 n)
```

```
mllib_MatrixTranspose_[U8|U8C|S8|S8C|S16|S16C|S32|S32C](datatype *xz, mlib_s32 mn)
```

```
mllib_MatrixTranspose_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C](datatype *z, datatype *x,
  mlib_s32 m, mlib_s32 n)
```

```
mllib_MatrixUnit_[U8|U8C|S8|S8C|S16|S16C|S32|S32C](datatype *z, mlib_s32 n)
```

---

### *Linear Algebra: Vector Functions*

---

```
mllib_VectorAdd_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *y, mlib_s32 n)
```

```
mllib_VectorAdd_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
  datatype *y, mlib_s32 n)
```

```
mllib_VectorAddS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *c, mlib_s32 n)
```

```
mllib_VectorAddS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
  datatype *c, mlib_s32 n)
```

## *Linear Algebra: Vector Functions (Continued)*

```
mllib_VectorAng_[U8C|S8C|S16C|S32C](mllib_d64 *a, datatype *x, mllib_s32 n)
mllib_VectorConj_[S8C|S16C|S32C]_Sat(datatype *xz, mllib_s32 n)
mllib_VectorConj_[S8C|S16C|S32C]_[S8C|S16C|S32C]_Sat(datatype *z, datatype *x, mllib_s32 n)
mllib_VectorConjRev_[S8C|S16C|S32C]_[S8C|S16C|S32C]_Sat(datatype *z, datatype *x, mllib_s32 n)
mllib_VectorConjSymExt_[S8C|S16C|S32C]_[S8C|S16C|S32C]_Sat(datatype *z, datatype *x, mllib_s32 n)
mllib_VectorDistance_[U8|S8|S16|S32]_Sat(mllib_d64 *z, datatype *x, datatype *y, mllib_s32 n)
mllib_VectorDotProd_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_Sat(mllib_d64 *z, datatype *x, datatype *y, mllib_s32 n)
mllib_VectorMag_[U8C|S8C|S16C|S32C](mllib_d64 *m, datatype *x, mllib_s32 n)
mllib_VectorMul_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *y, mllib_s32 n)
mllib_VectorMul_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
    datatype *y, mllib_s32 n)
mllib_VectorMulM_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
    datatype *y, mllib_s32 m, mllib_s32 n)
mllib_VectorMulS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *c, mllib_s32 n)
mllib_VectorMulS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
    datatype *c, mllib_s32 n)
mllib_VectorMulSAdd_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *y, datatype *c, mllib_s32 n)
```

---

**Linear Algebra: Vector Functions (Continued)**


---

```

mlib_VectorMulSAdd_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z,
    datatype *x, datatype *y, datatype *c, mlib_s32 n)
mlib_VectorNorm_[U8|S8|S16|S32]_Sat(mlib_d64 *z, datatype *x, mlib_s32 n)
mlib_VectorScale_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *a, datatype *b, mlib_s32 n)
mlib_VectorScale_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z,
    datatype *x, datatype *a, datatype *b, mlib_s32 n)
mlib_VectorSet_[U8|U8C|S8|S8C|S16|S16C|S32|S32C](datatype *z, datatype *c, mlib_s32 n)
mlib_VectorSub_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *y, mlib_s32 n)
mlib_VectorSub_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
    datatype *y, mlib_s32 n)
mlib_VectorSubS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *xz, datatype *c, mlib_s32 n)
mlib_VectorSubS_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[U8|U8C|S8|S8C|S16|S16C|S32|S32C]_[Sat|Mod](datatype *z, datatype *x,
    datatype *c, mlib_s32 n)
mlib_VectorSumAbs_[U8|S8|S16|S32]_Sat(mlib_d64 *z, datatype *x, mlib_s32 n)
mlib_VectorSumAbsDiff_[U8|S8|S16|S32]_Sat(mlib_d64 *z, datatype *x, datatype *y, mlib_s32 n)
mlib_VectorZero_[U8|U8C|S8|S8C|S16|S16C|S32|S32C](datatype *z, mlib_s32 n)

```

---

## SIGNAL AND AUDIO FUNCTIONS AND SYNTAX

---

### *Signal and Audio: Channel Merge and Split*

`mlib_SignalMerge_F32S_F32(mlib_f32 *dst, mlib_f32 *ch0, mlib_f32 *ch1, mlib_s32 n)`

`mlib_SignalMerge_S16S_S16(mlib_s16 *dst, mlib_s16 *ch0, mlib_s16 *ch1, mlib_s32 n)`

`mlib_SignalSplit_S16_S16S(mlib_s16 *ch0, mlib_s16 *ch1, mlib_s16 *src, mlib_s32 n)`

---

### *Signal and Audio: Cepstral Analysis*

`mlib_SignalCepstral_F32(mlib_f32 *cepst, mlib_f32 *signal, void *state)`

`mlib_SignalCepstral_S16(mlib_s16 *cepst, mlib_s32 cscale, mlib_s16 *signal, void *state)`

`mlib_SignalCepstral_S16_Adp(mlib_s16 *cepst, mlib_s32 *cscale, mlib_s16 *signal, void *state)`

`mlib_SignalCepstralFree_F32(void *state)1`

`mlib_SignalCepstralFree_S16(void *state)1`

`mlib_SignalCepstralInit_F32(void **state, mlib_s32 order)`

`mlib_SignalCepstralInit_S16(void **state, mlib_s32 order)`

`mlib_SignalMelCepstral_F32(mlib_f32 *cepst, mlib_f32 *signal, void *state)`

`mlib_SignalMelCepstral_S16(mlib_s16 *cepst, mlib_s32 cscale, mlib_s16 *signal, void *state)`

---

---

### *Signal and Audio: Cepstral Analysis (Continued)*

---

`mlib_SignalMelCepstral_S16_Adp`(mlib\_s16 \*cepst, mlib\_s32 \*cscale, mlib\_s16 \*signal, void \*state)

`mlib_SignalMelCepstralFree_F32`(void \*state)<sup>1</sup>

`mlib_SignalMelCepstralFree_S16`(void \*state)<sup>1</sup>

`mlib_SignalMelCepstralInit_F32`(void \*\*state, mlib\_s32 nlinear, mlib\_s32 nmel, mlib\_f32 melbgn, mlib\_f32 melend, mlib\_f32 meldiv, mlib\_s32 order)

`mlib_SignalMelCepstralInit_S16`(void \*\*state, mlib\_s32 nlinear, mlib\_s32 nmel, mlib\_f32 melbgn, mlib\_f32 melend, mlib\_f32 meldiv, mlib\_s32 order)

---

1. This function requires the return type “void.”

---

### *Signal and Audio: Convolution and Correlation*

---

`mlib_SignalAutoCorrel_[F32|F32S]`(mlib\_d64 \*correl, mlib\_f32 \*src, mlib\_s32 disp, mlib\_s32 n)

`mlib_SignalAutoCorrel_[S16|S16S]`(mlib\_d64 \*correl, mlib\_s16 \*src, mlib\_s32 disp, mlib\_s32 n)

`mlib_SignalConv_[F32|F32S]_[F32|F32S]`(mlib\_f32 \*dst, mlib\_f32 \*src1, mlib\_f32 \*src2, mlib\_s32 m, mlib\_s32 n)

`mlib_SignalConv_[S16|S16S]_[S16|S16S]_Sat`(mlib\_s16 \*dst, mlib\_s16 \*src1, mlib\_s16 \*src2, mlib\_s32 m, mlib\_s32 n)

`mlib_SignalCrossCorrel_[F32|F32S]`(mlib\_d64 \*correl, mlib\_f32 \*src1, mlib\_f32 \*src2, mlib\_s32 n)

`mlib_SignalCrossCorrel_[S16|S16S]`(mlib\_d64 \*correl, mlib\_s16 \*src1, mlib\_s16 \*src2, mlib\_s32 n)

---

---

## *Signal and Audio: Data Type Conversion*

---

```
mlib_signalConvertShift_[F32|F32S]_[U8|U8S|S8|S8S|S16|S16S|S32|S32S](datatype *dst, datatype *src, mlib_s32 shift,
    mlib_s32 n)
mlib_signalConvertShift_[U8|U8S|S8|S8S|S16|S16S|S32|S32S]_[U8|U8S|S8|S8S|S16|S16S|S32|S32S]_sat(void *dst, void *src,
    mlib_s32 shift, mlib_s32 n)
```

---

---

## *Signal and Audio: Dynamic Time Warping*

---

```
mlib_signalDTWKScalar_F32(mlib_d64 *dist, mlib_f32 *dobs, mlib_s32 lobs, void *state)
mlib_signalDTWKScalar_S16(mlib_d64 *dist, mlib_s16 *dobs, mlib_s32 lobs, mlib_s32 sob, void *state)
mlib_signalDTWKScalarFree_F32(void *state)1
mlib_signalDTWKScalarFree_S16(void *state)1
mlib_signalDTWKScalarInit_F32(void **state, mlib_f32 *dref, mlib_s32 lref, mlib_s32 kbest, mlib_s32 delta, mlib_s32
    local, mlib_s32 slope)
mlib_signalDTWKScalarInit_S16(void **state, mlib_s16 *dref, mlib_s32 lref, mlib_s32 kbest, mlib_s32 sref, mlib_s32
    delta, mlib_s32 local, mlib_s32 slope)
mlib_signalDTWKScalarPath_F32(mlib_s32 *path, mlib_s32 *lpath, mlib_s32 kpath, void *state)
mlib_signalDTWKScalarPath_S16(mlib_s32 *path, mlib_s32 *lpath, mlib_s32 kpath, void *state)
```

---

*Signal and Audio: Dynamic Time Warping (Continued)*

---

```
mllib_SignalDTWKVector_F32(mllib_d64 *dist, mllib_f32 **dobs, mllib_s32 lobs, void *state)
```

```
mllib_SignalDTWKVector_S16(mllib_d64 *dist, mllib_s16 **dobs, mllib_s32 lobs, mllib_s32 sob, void *state)
```

```
mllib_SignalDTWKVectorFree_F32(void *state)1
```

```
mllib_SignalDTWKVectorFree_S16(void *state)1
```

```
mllib_SignalDTWKVectorInit_F32(void **state, mllib_f32 **dref, mllib_s32 lref, mllib_s32 ndata, mllib_s32 kbest, mllib_s32  
dtype, mllib_s32 delta, mllib_s32 local, mllib_s32 slope)
```

```
mllib_SignalDTWKVectorInit_S16(void **state, mllib_s16 **dref, mllib_s32 lref, mllib_s32 ndata, mllib_s32 kbest,  
mllib_s32 dtype, mllib_s32 sref, mllib_s32 delta, mllib_s32 local, mllib_s32 slope)
```

```
mllib_SignalDTWKVectorPath_F32(mllib_s32 *path, mllib_s32 *lpath, mllib_s32 kpath, void *state)
```

```
mllib_SignalDTWKVectorPath_S16(mllib_s32 *path, mllib_s32 *lpath, mllib_s32 kpath, void *state)
```

```
mllib_SignalDTWScalar_F32(mllib_d64 *dist, mllib_f32 *dobs, mllib_s32 lobs, void *state)
```

```
mllib_SignalDTWScalar_S16(mllib_d64 *dist, mllib_s16 *dobs, mllib_s32 lobs, mllib_s32 sob, void *state)
```

```
mllib_SignalDTWScalarFree_F32(void *state)1
```

```
mllib_SignalDTWScalarFree_S16(void *state)1
```

```
mllib_SignalDTWScalarInit_F32(void **state, mllib_f32 *dref, mllib_s32 lref, mllib_s32 delta, mllib_s32 local, mllib_s32  
slope)
```

```
mllib_SignalDTWScalarInit_S16(void **state, mllib_s16 *dref, mllib_s32 lref, mllib_s32 sref, mllib_s32 delta, mllib_s32 local,  
mllib_s32 slope)
```

---

## *Signal and Audio: Dynamic Time Warping (Continued)*

---

**mllib\_SignalDTWScalarPath\_F32**(mllib\_d64 \*dist, mllib\_s32 \*path, mllib\_s32 \*lpath, mllib\_f32 \*pdobs, mllib\_s32 lobs, void \*state)

**mllib\_SignalDTWScalarPath\_S16**(mllib\_d64 \*dist, mllib\_s32 \*path, mllib\_s32 \*lpath, mllib\_s16 \*pdobs, mllib\_s32 lobs, mllib\_s32 sob, void \*state)

**mllib\_SignalDTWVector\_F32**(mllib\_d64 \*dist, mllib\_f32 \*\*dobs, mllib\_s32 lobs, void \*state)

**mllib\_SignalDTWVector\_S16**(mllib\_d64 \*dist, mllib\_s16 \*\*dobs, mllib\_s32 lobs, mllib\_s32 sob, void \*state)

**mllib\_SignalDTWVectorFree\_F32**(void \*state)<sup>1</sup>

**mllib\_SignalDTWVectorFree\_S16**(void \*state)<sup>1</sup>

**mllib\_SignalDTWVectorInit\_F32**(void \*\*state, mllib\_f32 \*\*dref, mllib\_s32 lref, mllib\_s32 ndata, mllib\_s32 dtype, mllib\_s32 delta, mllib\_s32 local, mllib\_s32 slope)

**mllib\_SignalDTWVectorInit\_S16**(void \*\*state, mllib\_s16 \*\*dref, mllib\_s32 lref, mllib\_s32 ndata, mllib\_s32 dtype, mllib\_s32 sref, mllib\_s32 delta, mllib\_s32 local, mllib\_s32 slope)

**mllib\_SignalDTWVectorPath\_F32**(mllib\_d64 \*dist, mllib\_s32 \*path, mllib\_s32 \*lpath, mllib\_f32 \*\*pdobs, mllib\_s32 lobs, void \*state)

**mllib\_SignalDTWVectorPath\_S16**(mllib\_d64 \*dist, mllib\_s32 \*path, mllib\_s32 \*lpath, mllib\_s16 \*\*pdobs, mllib\_s32 lobs, mllib\_s32 sob, void \*state)

---

1. This function requires the return type "void."

---

*Signal and Audio: Encoding and Decoding*

---

```
mlib_SignalADPCM2Bits2Linear(mlib_s16 *pcm, mlib_u8 *adpcm, void *state, mlib_s32 n)
mlib_SignalADPCM3Bits2Linear(mlib_s16 *pcm, mlib_u8 *adpcm, void *state, mlib_s32 n)
mlib_SignalADPCM4Bits2Linear(mlib_s16 *pcm, mlib_u8 *adpcm, void *state, mlib_s32 n)
mlib_SignalADPCM5Bits2Linear(mlib_s16 *pcm, mlib_u8 *adpcm, void *state, mlib_s32 n)
mlib_SignalADPCMFree(void *state)1
mlib_SignalADPCMInit(void **state)
mlib_SignalALaw2Linear(mlib_s16 *pcm, mlib_u8 *acode, mlib_s32 n)
mlib_SignalALaw2uLaw(mlib_u8 *ucode, mlib_u8 *acode, mlib_s32 n)
mlib_SignalIMDCT_D64(mlib_d64 *data)
mlib_SignalIMDCT_F32(mlib_f32 *data)
mlib_SignalIMDCTSPLIT_D64(mlib_d64 *data)
mlib_SignalIMDCTSPLIT_F32(mlib_f32 *data)
mlib_SignalLinear2ADPCM2Bits(mlib_su8 *adpcm, mlib_s16 *pcm, void *state, mlib_s32 n)
mlib_SignalLinear2ADPCM3Bits(mlib_u8 *adpcm, mlib_s16 *pcm, void *state, mlib_s32 n)
mlib_SignalLinear2ADPCM4Bits(mlib_u8 *adpcm, mlib_s16 *pcm, void *state, mlib_s32 n)
mlib_SignalLinear2ADPCM5Bits(mlib_u8 *adpcm, mlib_s16 *pcm, void *state, mlib_s32 n)
```

---

## *Signal and Audio: Encoding and Decoding (Continued)*

---

`mlib_SignalLinear2ALaw(mlib_u8 *acode, mlib_s16 *pcm, mlib_s32 n)`

`mlib_SignalLinear2uLaw(mlib_u8 *ucode, mlib_s16 *pcm, mlib_s32 n)`

`mlib_SignaluLaw2ALaw(mlib_u8 *acode, mlib_u8 *ucode, mlib_s32 n)`

`mlib_SignaluLaw2Linear(mlib_s16 *pcm, mlib_u8 *ucode, mlib_s32 n)`

---

1. This function requires the return type “void.”

---

## *Signal and Audio: Fast Fourier Transform (FFT)*

---

`mlib_SignalFFT_1_D64(mlib_d64 *srcdstr, mlib_d64 *srcdsti, mlib_s32 order)`

`mlib_SignalFFT_1_D64_D64(mlib_d64 *dstr, mlib_d64 *dsti, mlib_d64 *srcr, mlib_d64 *srci, mlib_s32 order)`

`mlib_SignalFFT_1_D64C(mlib_d64 *srcdstc, mlib_s32 order)`

`mlib_SignalFFT_1_D64C_D64(mlib_d64 *dstc, mlib_d64 *srcr, mlib_s32 order)`

`mlib_SignalFFT_1_D64C_D64C(mlib_d64 *dstc, mlib_d64 *srcc, mlib_s32 order)`

`mlib_SignalFFT_2_D64(mlib_d64 *srcdstr, mlib_d64 *srcdsti, mlib_s32 order)`

`mlib_SignalFFT_2_D64_D64(mlib_d64 *dstr, mlib_d64 *dsti, mlib_d64 *srcr, mlib_d64 *srci, mlib_s32 order)`

`mlib_SignalFFT_2_D64C(mlib_d64 *srcdstc, mlib_s32 order)`

`mlib_SignalFFT_2_D64C_D64(mlib_d64 *dstc, mlib_d64 *srcr, mlib_s32 order)`

`mlib_SignalFFT_2_D64C_D64C(mlib_d64 *dstc, mlib_d64 *srcc, mlib_s32 order)`

---

---

*Signal and Audio: Fast Fourier Transform (FFT) (Continued)*

---

```
mlib_SignalFFT_2_S16(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s32 order)
mlib_SignalFFT_2_S16_S16(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srcr, mlib_s16 *srci, mlib_s32 order)
mlib_SignalFFT_2_S16C(mlib_s16 *srcdstc, mlib_s32 order)
mlib_SignalFFT_2_S16C_S16(mlib_s16 *dstc, mlib_s16 *srcr, mlib_s32 order)
mlib_SignalFFT_2_S16C_S16C(mlib_s16 *dstc, mlib_s16 *srcc, mlib_s32 order)
mlib_SignalFFT_3_D64(mlib_d64 *srcdstr, mlib_d64 *srcdsti, mlib_s32 order)
mlib_SignalFFT_3_D64_D64(mlib_d64 *dstr, mlib_d64 *dsti, mlib_d64 *srcr, mlib_d64 *srci, mlib_s32 order)
mlib_SignalFFT_3_D64C(mlib_d64 *srcdstc, mlib_s32 order)
mlib_SignalFFT_3_D64C_D64(mlib_d64 *dstc, mlib_d64 *srcr, mlib_s32 order)
mlib_SignalFFT_3_D64C_D64C(mlib_d64 *dstc, mlib_d64 *srcc, mlib_s32 order)
mlib_SignalFFT_4_S16(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s32 order, mlib_s32 *scale)
mlib_SignalFFT_4_S16_S16(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srcr, mlib_s16 *srci, mlib_s32 order,
    mlib_s32 *scale)
mlib_SignalFFT_4_S16C(mlib_s16 *srcdstc, mlib_s32 order, mlib_s32 *scale)
mlib_SignalFFT_4_S16C_S16(mlib_s16 *dstc, mlib_s16 *srcr, mlib_s32 order, mlib_s32 *scale)
mlib_SignalFFT_4_S16C_S16C(mlib_s16 *dstc, mlib_s16 *srcc, mlib_s32 order, mlib_s32 *scale)
mlib_SignalFFT_[1|3]_S16_Mod(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s32 order)
```

---

## *Signal and Audio: Fast Fourier Transform (FFT) (Continued)*

---

`mlib_SignalFFT_[1|3]_S16_S16_Mod(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srcr, mlib_s16 *srci, mlib_s32 order)`

`mlib_SignalFFT_[1|3]_S16C_Mod(mlib_s16 *srcdstc, mlib_s32 order)`

`mlib_SignalFFT_[1|3]_S16C_S16_Mod(mlib_s16 *dstc, mlib_s16 *srcr, mlib_s32 order)`

`mlib_SignalFFT_[1|3]_S16C_S16C_Mod(mlib_s16 *dstc, mlib_s16 *srcr, mlib_s32 order)`

`mlib_SignalFFTW_2_S16(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s16 *window, mlib_s32 order)`

`mlib_SignalFFTW_2_S16_S16(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srcr, mlib_s16 *srci, mlib_s16 *window,  
mlib_s32 order)`

`mlib_SignalFFTW_2_S16C(mlib_s16 *srcdstc, mlib_s16 *window, mlib_s32 order)`

`mlib_SignalFFTW_2_S16C_S16(mlib_s16 *dstc, mlib_s16 *srcr, mlib_s16 *window, mlib_s32 order)`

`mlib_SignalFFTW_2_S16C_S16C(mlib_s16 *dstc, mlib_s16 *srcr, mlib_s16 *window, mlib_s32 order)`

`mlib_SignalFFTW_4_S16(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s16 *window, mlib_s32 order, mlib_s32 *scale)`

`mlib_SignalFFTW_4_S16_S16(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srcr, mlib_s16 *srci, mlib_s16 *window,  
mlib_s32 order, mlib_s32 *scale)`

`mlib_SignalFFTW_4_S16C(mlib_s16 *srcdstc, mlib_s16 *window, mlib_s32 order, mlib_s32 *scale)`

`mlib_SignalFFTW_4_S16C_S16(mlib_s16 *dstc, mlib_s16 *srcr, mlib_s16 *window, mlib_s32 order, mlib_s32 *scale)`

`mlib_SignalFFTW_4_S16C_S16C(mlib_s16 *dstc, mlib_s16 *srcr, mlib_s16 *window, mlib_s32 order, mlib_s32 *scale)`

`mlib_SignalFFTW_[1|2|3]_F32(mlib_f32 *srcdstr, mlib_f32 *srcdsti, mlib_f32 *window, mlib_s32 order)`

---

*Signal and Audio: Fast Fourier Transform (FFT) (Continued)*

---

```
mlib_SignalFFTW_[1|2|3]_F32_F32(mlib_f32 *dstr, mlib_f32 *dsti, mlib_f32 *srcr, mlib_f32 *srci, mlib_f32 *window,
    mlib_s32 order)
mlib_SignalFFTW_[1|2|3]_F32C(mlib_f32 *srcdstc, mlib_f32 *window, mlib_s32 order)
mlib_SignalFFTW_[1|2|3]_F32C_F32(mlib_f32 *dstc, mlib_f32 *srcr, mlib_f32 *window, mlib_s32 order)
mlib_SignalFFTW_[1|2|3]_F32C_F32C(mlib_f32 *dstc, mlib_f32 *srcc, mlib_f32 *window, mlib_s32 order)
mlib_SignalFFTW_[1|3]_S16_Mod(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s16 *window, mlib_s32 order)
mlib_SignalFFTW_[1|3]_S16_S16_Mod(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srcr, mlib_s16 *srci, mlib_s16 *window,
    mlib_s32 order)
mlib_SignalFFTW_[1|3]_S16C_Mod(mlib_s16 *srcdstc, mlib_s16 *window, mlib_s32 order)
mlib_SignalFFTW_[1|3]_S16C_S16_Mod(mlib_s16 *dstc, mlib_s16 *srcr, mlib_s16 *window, mlib_s32 order)
mlib_SignalFFTW_[1|3]_S16C_S16C_Mod(mlib_s16 *dstc, mlib_s16 *srcc, mlib_s16 *window, mlib_s32 order)
mlib_SignalIFFT_1_D64(mlib_d64 *srcdstr, mlib_d64 *srcdsti, mlib_s32 order)
mlib_SignalIFFT_1_D64_D64(mlib_d64 *dstr, mlib_d64 *dsti, mlib_d64 *srcr, mlib_d64 *srci, mlib_s32 order)
mlib_SignalIFFT_1_D64_D64C(mlib_d64 *dstr, mlib_d64 *srcc, mlib_s32 order)
mlib_SignalIFFT_1_D64C(mlib_d64 *srcdstc, mlib_s32 order)
mlib_SignalIFFT_1_D64C_D64C(mlib_d64 *dstc, mlib_d64 *srcc, mlib_s32 order)
mlib_SignalIFFT_1_S16(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s32 order)
```

---

## *Signal and Audio: Fast Fourier Transform (FFT) (Continued)*

---

```
mlib_SignalIFFT_1_S16_S16(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srccr, mlib_s16 *srcci, mlib_s32 order)
mlib_SignalIFFT_1_S16_S16C(mlib_s16 *dstr, mlib_s16 *srcc, mlib_s32 order)
mlib_SignalIFFT_1_S16C(mlib_s16 *srcdstc, mlib_s32 order)
mlib_SignalIFFT_1_S16C_S16C(mlib_s16 *dstc, mlib_s16 *srcc, mlib_s32 order)
mlib_SignalIFFT_2_D64(mlib_d64 *srcdstr, mlib_d64 *srcdsti, mlib_s32 order)
mlib_SignalIFFT_2_D64_D64(mlib_d64 *dstr, mlib_d64 *dsti, mlib_d64 *srccr, mlib_d64 *srcci, mlib_s32 order)
mlib_SignalIFFT_2_D64_D64C(mlib_d64 *dstr, mlib_d64 *srcc, mlib_s32 order)
mlib_SignalIFFT_2_D64C(mlib_d64 *srcdstc, mlib_s32 order)
mlib_SignalIFFT_2_D64C_D64C(mlib_d64 *dstc, mlib_d64 *srcc, mlib_s32 order)
mlib_SignalIFFT_3_D64(mlib_d64 *srcdstr, mlib_d64 *srcdsti, mlib_s32 order)
mlib_SignalIFFT_3_D64_D64(mlib_d64 *dstr, mlib_d64 *dsti, mlib_d64 *srccr, mlib_d64 *srcci, mlib_s32 order)
mlib_SignalIFFT_3_D64_D64C(mlib_d64 *dstr, mlib_d64 *srcc, mlib_s32 order)
mlib_SignalIFFT_3_D64C(mlib_d64 *srcdstc, mlib_s32 order)
mlib_SignalIFFT_3_D64C_D64C(mlib_d64 *dstc, mlib_d64 *srcc, mlib_s32 order)
mlib_SignalIFFT_4_S16(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s32 order, mlib_s32 *scale)
mlib_SignalIFFT_4_S16_S16(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srccr, mlib_s16 *srcci, mlib_s32 order,
    mlib_s32 *scale)
```

---

*Signal and Audio: Fast Fourier Transform (FFT) (Continued)*

---

```
mlib_SignalIFFT_4_S16_S16C(mlib_s16 *dstr, mlib_s16 *srcc, mlib_s32 order, mlib_s32 *scale)
```

```
mlib_SignalIFFT_4_S16C(mlib_s16 *srcdstc, mlib_s32 order, mlib_s32 *scale)
```

```
mlib_SignalIFFT_4_S16C_S16C(mlib_s16 *dstc, mlib_s16 *srcc, mlib_s32 order, mlib_s32 *scale)
```

```
mlib_SignalIFFT_[1|2|3]_F32(mlib_f32 *srcdstr, mlib_f32 *srcdsti, mlib_s32 order)
```

```
mlib_SignalIFFT_[1|2|3]_F32_F32(mlib_f32 *dstr, mlib_f32 *dsti, mlib_f32 *srcr, mlib_f32 *srci, mlib_s32 order)
```

```
mlib_SignalIFFT_[1|2|3]_F32_F32C(mlib_f32 *dstr, mlib_f32 *srcc, mlib_s32 order)
```

```
mlib_SignalIFFT_[1|2|3]_F32C(mlib_f32 *srcdstc, mlib_s32 order)
```

```
mlib_SignalIFFT_[1|2|3]_F32C_F32C(mlib_f32 *dstc, mlib_f32 *srcc, mlib_s32 order)
```

```
mlib_SignalIFFT_[2|3]_S16_Mod(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s32 order)
```

```
mlib_SignalIFFT_[2|3]_S16_S16_Mod(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srcr, mlib_s16 *srci, mlib_s32 order)
```

```
mlib_SignalIFFT_[2|3]_S16_S16C_Mod(mlib_s16 *dstr, mlib_s16 *srcc, mlib_s32 order)
```

```
mlib_SignalIFFT_[2|3]_S16C_Mod(mlib_s16 *srcdstc, mlib_s32 order)
```

```
mlib_SignalIFFT_[2|3]_S16C_Mod(mlib_s16 *srcdstc, mlib_s32 order)
```

```
mlib_SignalIFFT_[2|3]_S16C_S16C_Mod(mlib_s16 *dstc, mlib_s16 *srcc, mlib_s32 order)
```

```
mlib_SignalIFFT_[2|3]_S16C_S16C_Mod(mlib_s16 *dstc, mlib_s16 *srcc, mlib_s32 order)
```

```
mlib_SignalIFFTW_1_S16(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s16 *window, mlib_s32 order)
```

## *Signal and Audio: Fast Fourier Transform (FFT) (Continued)*

```
mlib_SignalIFFTW_1_S16_S16(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srcc, mlib_s16 *srcc, mlib_s16 *window,
    mlib_s32 order)
mlib_SignalIFFTW_1_S16_S16C(mlib_s16 *dstr, mlib_s16 *srcc, mlib_s16 *window, mlib_s32 order)
mlib_SignalIFFTW_1_S16C(mlib_s16 *srcdstc, mlib_s16 *window, mlib_s32 order)
mlib_SignalIFFTW_1_S16C_S16C(mlib_s16 *dstc, mlib_s16 *srcc, mlib_s16 *window, mlib_s32 order)
mlib_SignalIFFTW_4_S16(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s16 *window, mlib_s32 order, mlib_s32 *scale)
mlib_SignalIFFTW_4_S16_S16(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srcc, mlib_s16 *srcc, mlib_s16 *window,
    mlib_s32 order, mlib_s32 *scale)
mlib_SignalIFFTW_4_S16_S16C(mlib_s16 *dstr, mlib_s16 *srcc, mlib_s16 *window, mlib_s32 order, mlib_s32 *scale)
mlib_SignalIFFTW_4_S16C(mlib_s16 *srcdstc, mlib_s16 *window, mlib_s32 order, mlib_s32 *scale)
mlib_SignalIFFTW_4_S16C_S16C(mlib_s16 *dstc, mlib_s16 *srcc, mlib_s16 *window, mlib_s32 order, mlib_s32 *scale)
mlib_SignalIFFTW_[1|2|3]_F32(mlib_f32 *srcdstr, mlib_f32 *srcdsti, mlib_f32 *window, mlib_s32 order)
mlib_SignalIFFTW_[1|2|3]_F32_F32(mlib_f32 *dstr, mlib_f32 *dsti, mlib_f32 *srcc, mlib_f32 *srcc, mlib_f32 *window,
    mlib_s32 order)
mlib_SignalIFFTW_[1|2|3]_F32_F32C(mlib_f32 *dstr, mlib_f32 *srcc, mlib_f32 *window, mlib_s32 order)
mlib_SignalIFFTW_[1|2|3]_F32C(mlib_f32 *srcdstc, mlib_f32 *window, mlib_s32 order)
mlib_SignalIFFTW_[1|2|3]_F32C_F32C(mlib_f32 *dstc, mlib_f32 *srcc, mlib_f32 *window, mlib_s32 order)
mlib_SignalIFFTW_[2|3]_S16_Mod(mlib_s16 *srcdstr, mlib_s16 *srcdsti, mlib_s16 *window, mlib_s32 order)
```

---

### *Signal and Audio: Fast Fourier Transform (FFT) (Continued)*

---

```
mlib_SignalIFFTW_[2|3]_S16_S16_Mod(mlib_s16 *dstr, mlib_s16 *dsti, mlib_s16 *srcr, mlib_s16 *srci, mlib_s16 *window,
    mlib_s32 order)
mlib_SignalIFFTW_[2|3]_S16_S16C_Mod(mlib_s16 *dstc, mlib_s16 *srcc, mlib_s16 *window, mlib_s32 order)
mlib_SignalIFFTW_[2|3]_S16C_Mod(mlib_s16 *srcdstc, mlib_s16 *window, mlib_s32 order)
mlib_SignalIFFTW_[2|3]_S16C_S16C_Mod(mlib_s16 *dstc, mlib_s16 *srcc, mlib_s16 *window, mlib_s32 order)
```

---

### *Signal and Audio: Finite Impulse Response (FIR) Filtering*

---

```
mlib_SignalFIR_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, void *filter, mlib_s32 n)
mlib_SignalFIR_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src, void *filter, mlib_s32 n)
mlib_SignalFIRFree_[F32|F32S]_[F32|F32S](void *filter)1
mlib_SignalFIRFree_[S16|S16S]_[S16|S16S](void *filter)1
mlib_SignalFIRInit_[F32|F32S]_[F32|F32S](void **filter, mlib_f32 *flt, mlib_s32 tap)
mlib_SignalFIRInit_[S16|S16S]_[S16|S16S](void **filter, mlib_f32 *flt, mlib_s32 tap)
mlib_SignalReSampleFIR_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, void *state, mlib_s32 n)
mlib_SignalReSampleFIR_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src, void *state, mlib_s32 n)
mlib_SignalReSampleFIRFree_[F32|F32S]_[F32|F32S](void *state)1
```

---

## *Signal and Audio: Finite Impulse Response (FIR) Filtering (Continued)*

---

**mlib\_SignalReSampleFIRFree\_**[S16|S16S]\_[S16|S16S](void \*state)<sup>1</sup>

**mlib\_SignalReSampleFIRInit\_**[F32|F32S]\_[F32|F32S](void \*\*state, mlib\_f32 \*flt, mlib\_s32 tap, mlib\_s32 factor, mlib\_s32 phase)

**mlib\_SignalReSampleFIRInit\_**[S16|S16S]\_[S16|S16S](void \*\*state, mlib\_f32 \*flt, mlib\_s32 tap, mlib\_s32 ufactor, mlib\_s32 uphase, mlib\_s32 dfactor, mlib\_s32 dphase)

---

1. This function requires the return type “void.”

---

## *Signal and Audio: Hard Limiting*

---

**mlib\_SignalLimit\_**[F32|F32S](mlib\_f32 \*srcdst, mlib\_f32 \*low, mlib\_f32 \*high, mlib\_s32 n)

**mlib\_SignalLimit\_**[F32|F32S]\_[F32|F32S](mlib\_f32 \*dst, mlib\_f32 \*src, mlib\_f32 \*low, mlib\_f32 \*high, mlib\_s32 n)

**mlib\_SignalLimit\_**[S16|S16S](mlib\_s16 \*srcdst, mlib\_s16 \*low, mlib\_s16 \*high, mlib\_s32 n)

**mlib\_SignalLimit\_**[S16|S16S]\_[S16|S16S](mlib\_s16 \*dst, mlib\_s16 \*src, mlib\_s16 \*low, mlib\_s16 \*high, mlib\_s32 n)

---

---

### *Signal and Audio: Infinite Impulse Response (IIR) Filtering*

---

```

mlib_SignalIIR_Biquad_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, void *filter, mlib_s32 n)
mlib_SignalIIR_Biquad_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src, void *filter, mlib_s32 n)
mlib_SignalIIR_P4_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, void *filter, mlib_s32 n)
mlib_SignalIIR_P4_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src, void *filter, mlib_s32 n)
mlib_SignalIIRFree_Biquad_[F32|F32S]_[F32|F32S](void *filter)1
mlib_SignalIIRFree_Biquad_[S16|S16S]_[S16|S16S](void *filter)1
mlib_SignalIIRFree_P4_[F32|F32S]_[F32|F32S](void *filter)1
mlib_SignalIIRFree_P4_[S16|S16S]_[S16|S16S](void *filter)1
mlib_SignalIIRInit_Biquad_[F32|F32S]_[F32|F32S](void **filter, mlib_f32 *flt)
mlib_SignalIIRInit_Biquad_[S16|S16S]_[S16|S16S](void **filter, mlib_f32 *flt)
mlib_SignalIIRInit_P4_[F32|F32S]_[F32|F32S](void **filter, mlib_s32 flt)
mlib_SignalIIRInit_P4_[S16|S16S]_[S16|S16S](void **filter, mlib_f32 *flt)

```

---

1. This function requires the return type “void.”

---

## *Signal and Audio: LMS Adaptive Filtering*

---

```
mlib_SignalLMSFilter_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, mlib_f32 *ref, void *filter, mlib_s32 n)
mlib_SignalLMSFilter_[S16|S16S]_[S16|S16S]_sat(mlib_s16 *dst, mlib_s16 *src, mlib_s16 *ref, void *filter, mlib_s32 n)
mlib_SignalLMSFilterFree_[F32|F32S]_[F32|F32S](void *filter)1
mlib_SignalLMSFilterFree_[S16|S16S]_[S16|S16S](void *filter)1
mlib_SignalLMSFilterInit_[S16|S16S]_[S16|S16S](void **filter, mlib_f32 *flt, mlib_s32 tap, mlib_f32 beta)
mlib_SignalLMSFilterInit_F32|F32S]_[F32|F32S](void **filter, mlib_f32 *flt, mlib_s32 tap, mlib_f32 beta)
```

---

1. This function requires the return type “void.”

---

## *Signal and Audio: Linear Predictive Coding*

---

```
mlib_SignalLPC2Cepstral_F32(mlib_f32 *cepst, mlib_f32 *lpc, mlib_f32 gain, mlib_s32 length, mlib_s32 order)
mlib_SignalLPC2Cepstral_S16(mlib_s16 *cepst, mlib_s32 cscale, mlib_s16 *lpc, mlib_s32 lscale, mlib_s16 gain, mlib_s32
    gscale, mlib_s32 length, mlib_s32 order)
mlib_SignalLPC2Cepstral_S16_Adp(mlib_s16 *cepst, mlib_s32 *cscale, mlib_s16 *lpc, mlib_s32 lscale, mlib_s16 gain,
    mlib_s32 gscale, mlib_s32 length, mlib_s32 order)
mlib_SignalLPC2LSP_F32(mlib_f32 *lsp, mlib_f32 *lpc, mlib_s32 order)
mlib_SignalLPC2LSP_S16(mlib_s16 *lsp, mlib_s16 *lpc, mlib_s32 lscale, mlib_s32 order)
```

---

*Signal and Audio: Linear Predictive Coding (Continued)*

---

```
mlib_SignalLPCAutoCorrel_F32(mlib_f32 *coeff, mlib_f32 *signal, void *state)
```

```
mlib_SignalLPCAutoCorrel_S16(mlib_s16 *coeff, mlib_s32 cscale, mlib_s16 *signal, void *state)
```

```
mlib_SignalLPCAutoCorrel_S16_Adpt(mlib_s16 *coeff, mlib_s32 *cscale, mlib_s16 *signal, void *state)
```

```
mlib_SignalLPCAutoCorrelFree_F32(void *state)1
```

```
mlib_SignalLPCAutoCorrelFree_S16(void *state)1
```

```
mlib_SignalLPCAutoCorrelGetEnergy_F32(mlib_f32 *engery, void *state)
```

```
mlib_SignalLPCAutoCorrelGetEnergy_S16(mlib_s16 *engery, mlib_s32 escale, void *state)
```

```
mlib_SignalLPCAutoCorrelGetEnergy_S16_Adpt(mlib_s16 *engery, mlib_s32 *escale, void *state)
```

```
mlib_SignalLPCAutoCorrelGetPARCOR_F32(mlib_f32 *parcor, void *state)
```

```
mlib_SignalLPCAutoCorrelGetPARCOR_S16(mlib_s16 *parcor, mlib_s32 pscale, void *state)
```

```
mlib_SignalLPCAutoCorrelGetPARCOR_S16_Adpt(mlib_s16 *parcor, mlib_s32 *pscale, void *state)
```

```
mlib_SignalLPCAutoCorrelInit_F32(void **state, mlib_s32 length, mlib_s32 order)
```

```
mlib_SignalLPCAutoCorrelInit_S16(void **state, mlib_s32 length, mlib_s32 order)
```

```
mlib_SignalLPCCovariance_F32(mlib_f32 *coeff, mlib_f32 *signal, void *state)
```

```
mlib_SignalLPCCovariance_S16(mlib_s16 *coeff, mlib_s32 cscale, mlib_s16 *signal, void *state)
```

```
mlib_SignalLPCCovariance_S16_Adpt(mlib_s16 *coeff, mlib_s32 *cscale, mlib_s16 *signal, void *state)
```

```
mlib_SignalLPCCovarianceFree_F32(void *state)1
```

---

## *Signal and Audio: Linear Predictive Coding (Continued)*

---

```
mlib_SignalLPCCovarianceFree_S16(void *state)1  
mlib_SignalLPCCovarianceInit_F32(void **state, mlib_s32 length, mlib_s32 order)  
mlib_SignalLPCCovarianceInit_S16(void **state, mlib_s32 length, mlib_s32 order)  
mlib_SignalLPCPerceptWeight_F32(mlib_f32 *sigwgt, mlib_f32 *signal, mlib_f32 *lpc, mlib_f32 r1, mlib_f32 r2, void  
    *state)  
mlib_SignalLPCPerceptWeight_S16(mlib_s16 *sigwgt, mlib_s16 *signal, mlib_s16 *lpc, mlib_s32 lscale, mlib_s16 r1,  
    mlib_s16 r2, void *state)  
mlib_SignalLPCPerceptWeightFree_F32(void *state)1  
mlib_SignalLPCPerceptWeightFree_S16(void *state)1  
mlib_SignalLPCPerceptWeightInit_F32(void **state, mlib_s32 length, mlib_s32 order)  
mlib_SignalLPCPerceptWeightInit_S16(void **state, mlib_s32 length, mlib_s32 order)  
mlib_SignalLPCPitchAnalyze_F32(mlib_s32 *pitch, mlib_f32 *sigwgt, mlib_s32 *region, mlib_s32 length)  
mlib_SignalLPCPitchAnalyze_S16(mlib_s32 *pitch, mlib_s16 *sigwgt, mlib_s32 *region, mlib_s32 length)  
mlib_SignalLSP2LPC_F32(mlib_f32 *lpc, mlib_f32 *lsp, mlib_s32 order)  
mlib_SignalLSP2LPC_S16(mlib_s16 *lpc, mlib_s32 lscale, mlib_s16 *lsp, mlib_s32 order)  
mlib_SignalLSP2LPC_S16_AdP(mlib_s16 *lpc, mlib_s32 *lscale, mlib_s16 *lsp, mlib_s32 order)
```

---

1. This function requires the return type “void.”

---

## *Signal and Audio: Multiply*

---

```

mlib_SignalMul_[F32|F32S](mlib_f32 *src1dst, mlib_f32 *src2, mlib_s32 n)
mlib_SignalMul_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src1, mlib_f32 *src2, mlib_s32 n)
mlib_SignalMul_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src1, mlib_s16 *src2, mlib_s32 n)
mlib_SignalMul_[S16|S16S]_Sat(mlib_s16 *src1dst, mlib_s16 *src2, mlib_s32 n)
mlib_SignalMuls_[F32|F32S](mlib_f32 *srcdst, mlib_f32 *c, mlib_s32 n)
mlib_SignalMuls_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, mlib_f32 *c, mlib_s32 n)
mlib_SignalMuls_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src, mlib_s16 *c, mlib_s32 n)
mlib_SignalMuls_[S16|S16S]_Sat(mlib_s16 *srcdst, mlib_s16 *c, mlib_s32 n)
mlib_SignalMulsAdd_[F32|F32S](mlib_f32 *src1dst, mlib_f32 *src2, mlib_f32 *c, mlib_s32 n)
mlib_SignalMulsAdd_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src1, mlib_f32 *src2, mlib_f32 *c, mlib_s32 n)
mlib_SignalMulsAdd_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src1, mlib_s16 *src2, mlib_s16 *c, mlib_s32 n)
mlib_SignalMulsAdd_[S16|S16S]_Sat(mlib_s16 *src1dst, mlib_s16 *src2, mlib_s16 *c, mlib_s32 n)
mlib_SignalMulShift_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src1, mlib_s16 *src2, mlib_s32 shift,
    mlib_s32 n)
mlib_SignalMulShift_[S16|S16S]_Sat(mlib_s16 *src1dst, mlib_s16 *src2, mlib_s32 shift, mlib_s32 n)
mlib_SignalMulSShift_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src, mlib_s16 *c, mlib_s32 shift, mlib_s32 n)

```

---

## *Signal and Audio: Multiply (Continued)*

---

`mlib_signalMulSShift_[S16|S16S]_Sat(mlib_s16 *srcdst, mlib_s16 *c, mlib_s32 shift, mlib_s32 n)`

`mlib_signalMulSShiftAdd_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src1, mlib_s16 *src2, mlib_s16 *c, mlib_s32 shift, mlib_s32 n)`

`mlib_signalMulSShiftAdd_[S16|S16S]_Sat(mlib_s16 *srcldst, mlib_s16 *src2, mlib_s16 *c, mlib_s32 shift, mlib_s32 n)`

---

---

## *Signal and Audio: Pre-emphasizing*

---

`mlib_signalEmphasize_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, void *filter, mlib_s32 n)`

`mlib_signalEmphasize_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src, void *filter, mlib_s32 n)`

`mlib_signalEmphasizeFree_[F32|F32S]_[F32|F32S](void *filter)1`

`mlib_signalEmphasizeFree_[S16|S16S]_[S16|S16S](void *filter)1`

`mlib_signalEmphasizeInit_[F32|F32S]_[F32|F32S](void **filter, mlib_f32 alpha)`

`mlib_signalEmphasizeInit_[S16|S16S]_[S16|S16S](void **filter, mlib_f32 alpha)`

---

1. This function requires the return type “void.”

---

### *Signal and Audio: Requantization*

---

```

mlib_SignalQuant2_[S16|S16S]_[F32|F32S](mlib_S16 *dst, mlib_f32 *src, mlib_f32 *thresh, mlib_s32 length, mlib_s16
    offset, mlib_s32 n)
mlib_SignalQuant_[S16|S16S]_[F32|F32S](mlib_S16 *dst, mlib_f32 *src, mlib_f32 *thresh, mlib_s32 n)
mlib_SignalQuant_[U8|U8S]_[F32|F32S](mlib_u8 *dst, mlib_f32 *src, mlib_f32 *thresh, mlib_s32 n)
mlib_SignalQuant_[U8|U8S]_[S16|S16S](mlib_u8 *dst, mlib_s16 *src, mlib_s16 *thresh, mlib_s32 n)

```

---

### *Signal and Audio: Resampling*

---

```

mlib_SignalDownSample_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, mlib_s32 factor, mlib_s32 phase, mlib_s32 n)
mlib_SignalDownSample_[S16|S16S]_[S16|S16S](mlib_s16 *dst, mlib_s16 *src, mlib_s32 factor, mlib_s32 phase, mlib_s32 n)
mlib_SignalUpSample_[F32|F32]_[F32|F32](mlib_f32 *dst, mlib_f32 *src, mlib_s32 factor, mlib_s32 phase, mlib_s32 n)
mlib_SignalUpSample_[S16|S16S]_[S16|S16S](mlib_s16 *dst, mlib_s16 *src, mlib_s32 factor, mlib_s32 phase, mlib_s32 n)
mlib_SignalUpSampleFIR_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, void *state, mlib_s32 n)
mlib_SignalUpSampleFIR_[S16|S16S]_[S16|S16S]_Sat(mlib_s16 *dst, mlib_s16 *src, void *state, mlib_s32 n)
mlib_SignalUpSampleFIRFree_[F32|F32S]_[F32|F32S](void *state)1
mlib_SignalUpSampleFIRFree_[S16|S16S]_[S16|S16S](void *state)1

```

---

## *Signal and Audio: Resampling (Continued)*

---

```
mlib_SignalUpSampleFIRInit_[F32|F32S]_[F32|F32S](void **state, mlib_f32 *flt, mlib_s32 tap, mlib_s32 factor, mlib_s32 phase)
```

```
mlib_SignalUpSampleFIRInit_[S16|S16S]_[S16|S16S](void **state, mlib_f32 *flt, mlib_s32 tap, mlib_s32 factor, mlib_s32 phase)
```

---

1. This function requires the return type “void.”

---

## *Signal and Audio: Signal Generation*

---

```
mlib_SignalGaussNoise_F32(mlib_f32 *gnoise, void *state, mlib_s32 n)
```

```
mlib_SignalGaussNoise_S16(mlib_s16 *gnoise, void *state, mlib_s32 n)
```

```
mlib_SignalGaussNoiseFree_F32(void *state)1
```

```
mlib_SignalGaussNoiseFree_S16(void *state)1
```

```
mlib_SignalGaussNoiseInit_F32(void **state, mlib_f32 mag, mlib_f32 mean, mlib_f32 stddev, mlib_f32 seed)
```

```
mlib_SignalGaussNoiseInit_S16(void **state, mlib_s16 mag, mlib_f32 mean, mlib_f32 stddev, mlib_s16 seed)
```

```
mlib_SignalSineWave_S16(mlib_s16 *sine, void *state, mlib_s32 n)
```

```
mlib_SignalSineWaveFree_F32(void *state)1
```

```
mlib_SignalSineWaveFree_S16(void *state)1
```

```
mlib_SignalSineWaveInit_F32(void **state, mlib_f32 mag, mlib_f32 freq, mlib_f32 phase)
```

---

### *Signal and Audio: Signal Generation (Continued)*

---

`mlib_SignalSineWaveInit_S16`(void \*\*state, mlib\_s16 mag, mlib\_f32 freq, mlib\_f32 phase)

`mlib_SignalSplit_F32_F32S`(mlib\_f32 \*ch0, mlib\_f32 \*ch1, mlib\_f32 \*src, mlib\_f32 n)

`mlib_SignalWhiteNoise_F32`(mlib\_f32 \*wnoise, void \*state, mlib\_s32 n)

`mlib_SignalWhiteNoise_S16`(mlib\_s16 \*wnoise, void \*state, mlib\_s32 n)

`mlib_SignalWhiteNoiseFree_F32`(void \*state)<sup>1</sup>

`mlib_SignalWhiteNoiseFree_S16`(void \*state)<sup>1</sup>

`mlib_SignalWhiteNoiseInit_F32`(void \*\*state, mlib\_f32 mag, mlib\_f32 seed)

`mlib_SignalWhiteNoiseInit_S16`(void \*\*state, mlib\_s16 mag, mlib\_s16 seed)

---

1. This function requires the return type “void.”

---

### *Signal and Audio: Windowing*

---

`mlib_SignalGenBartlett_F32`(mlib\_f32 \*window, mlib\_s32 n)

`mlib_SignalGenBartlett_S16`(mlib\_s16 \*window, mlib\_s32 n)

`mlib_SignalGenBlackman_F32`(mlib\_f32 \*window, mlib\_f32 alpha, mlib\_s32 n)

`mlib_SignalGenBlackman_S16`(mlib\_s16 \*window, mlib\_f32 alpha, mlib\_s32 n)

`mlib_SignalGenHamming_F32`(mlib\_f32 \*window, mlib\_s32 n)

`mlib_SignalGenHamming_S16`(mlib\_s16 \*window, mlib\_s32 n)

---

## *Signal and Audio: Windowing (Continued)*

---

`mlib_SignalGenHanning_F32`(`mlib_f32 *window`, `mlib_s32 n`)

`mlib_SignalGenHanning_S16`(`mlib_s16 *window`, `mlib_s32 n`)

`mlib_SignalGenKaiser_F32`(`mlib_f32 *window`, `mlib_f32 beta`, `mlib_s32 n`)

`mlib_SignalGenKaiser_S16`(`mlib_s16 *window`, `mlib_f32 beta`, `mlib_s32 n`)

`mlib_SignalMulBartlett_[F32|F32S]`(`mlib_f32 *srcdst`, `mlib_s32 n`)

`mlib_SignalMulBartlett_[F32|F32S]_[F32|F32S]`(`mlib_f32 *dst`, `mlib_f32 *src`, `mlib_s32 n`)

`mlib_SignalMulBartlett_[S16|S16S]`(`mlib_s16 *srcdst`, `mlib_s32 n`)

`mlib_SignalMulBartlett_[S16|S16S]_[S16|S16S]`(`mlib_s16 *dst`, `mlib_s16 *src`, `mlib_s32 n`)

`mlib_SignalMulBlackman_[F32|F32S]`(`mlib_f32 *srcdst`, `mlib_s32 alpha`, `mlib_s32 n`)

`mlib_SignalMulBlackman_[F32|F32S]_[F32|F32S]`(`mlib_f32 *dst`, `mlib_f32 *src`, `mlib_f32 alpha`, `mlib_s32 n`)

`mlib_SignalMulBlackman_[S16|S16S]`(`mlib_s16 *srcdst`, `mlib_f32 alpha`, `mlib_s32 n`)

`mlib_SignalMulBlackman_[S16|S16S]_[S16|S16S]`(`mlib_s16 *dst`, `mlib_s16 *src`, `mlib_f32 alpha`, `mlib_s32 n`)

`mlib_SignalMulHamming_[F32|F32S]`(`mlib_f32 *srcdst`, `mlib_s32 n`)

`mlib_SignalMulHamming_[F32|F32S]_[F32|F32S]`(`mlib_f32 *dst`, `mlib_f32 *src`, `mlib_s32 n`)

`mlib_SignalMulHamming_[S16|S16S]`(`mlib_s16 *srcdst`, `mlib_s32 n`)

`mlib_SignalMulHamming_[S16|S16S]_[S16|S16S]`(`mlib_s16 *dst`, `mlib_s16 *src`, `mlib_s32 n`)

`mlib_SignalMulHanning_[F32|F32S]`(`mlib_f32 *srcdst`, `mlib_s32 n`)

---

*Signal and Audio: Windowing (Continued)*

---

`mlib_SignalMulHanning_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, mlib_s32 n)`

`mlib_SignalMulHanning_[S16|S16S](mlib_s16 *srcdst, mlib_s32 n)`

`mlib_SignalMulHanning_[S16|S16S]_[S16|S16S](mlib_s16 *dst, mlib_s16 *src, mlib_s32 n)`

`mlib_SignalMulKaiser_[F32|F32S](mlib_f32 *srcdst, mlib_f32 beta, mlib_s32 n)`

`mlib_SignalMulKaiser_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, mlib_f32 beta, mlib_s32 n)`

`mlib_SignalMulKaiser_[S16|S16S](mlib_s16 *srcdst, mlib_f32 beta, mlib_s32 n)`

`mlib_SignalMulKaiser_[S16|S16S]_[S16|S16S](mlib_s16 *dst, mlib_s16 *src, mlib_f32 beta, mlib_s32 n)`

`mlib_SignalMulRectangular_[F32|F32S](mlib_f32 *srcdst, mlib_s32 m, mlib_s32 n)`

`mlib_SignalMulRectangular_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, mlib_s32 m, mlib_s32 n)`

`mlib_SignalMulRectangular_[S16|S16S](mlib_s16 *srcdst, mlib_s32 m, mlib_s32 n)`

`mlib_SignalMulRectangular_[S16|S16S]_[S16|S16S](mlib_s16 *dst, mlib_s16 *src, mlib_s32 m, mlib_s32 n)`

`mlib_SignalMulWindow_[F32|F32S](mlib_f32 *srcdst, mlib_f32 *window, mlib_s32 n)`

`mlib_SignalMulWindow_[F32|F32S]_[F32|F32S](mlib_f32 *dst, mlib_f32 *src, mlib_f32 *window, mlib_s32 n)`

`mlib_SignalMulWindow_[S16|S16S](mlib_s16 *srcdst, mlib_s16 *window, mlib_s32 n)`

`mlib_SignalMulWindow_[S16|S16S]_[S16|S16S](mlib_s16 *dst, mlib_s16 *src, mlib_s16 *window, mlib_s32 n)`

---

# VIDEO FUNCTIONS AND SYNTAX

---

## *Video: Alpha-blending Functions*<sup>1</sup>

---

```
mlib_VideoColorBlendABGR(mlib_u32 *dst, mlib_u32 *src1, mlib_u32 *src2, mlib_s32 src1_w, mlib_s32 src1_h, mlib_s32 src2_w, mlib_s32 src2_h, mlib_s32 src2_x, mlib_s32 src2_y, mlib_s32 dst_lb, mlib_s32 src1_lb, mlib_s32 src2_lb, mlib_blend src1_blend, mlib_blend src2_blend)
```

```
mlib_VideoColorBlendABGR_Inp(mlib_u32 *src1dst, mlib_u32 *src2, mlib_s32 src1dst_w, mlib_s32 src1dst_h, mlib_s32 src2_w, mlib_s32 src2_h, mlib_s32 src2_x, mlib_s32 src2_y, mlib_s32 src1dst_lb, mlib_s32 src2_lb, mlib_blend src1dst_blend, mlib_blend src2_blend)
```

```
mlib_VideoColorBlendABGR_ResetAlpha(mlib_u32 *dst, mlib_u32 *src1, mlib_u32 *src2, mlib_s32 src1_w, mlib_s32 src1_h, mlib_s32 src2_w, mlib_s32 src2_h, mlib_s32 src2_x, mlib_s32 src2_y, mlib_s32 dst_lb, mlib_s32 src1_lb, mlib_s32 src2_lb, mlib_blend src1_blend, mlib_blend src2_blend)
```

```
mlib_VideoColorBlendABGR_ResetAlpha_Inp(mlib_u32 *src1dst, mlib_u32 *src2, mlib_s32 src1dst_w, mlib_s32 src1dst_h, mlib_s32 src2_w, mlib_s32 src2_h, mlib_s32 src2_x, mlib_s32 src2_y, mlib_s32 src1dst_lb, mlib_s32 src2_lb, mlib_blend src1dst_blend, mlib_blend src2_blend)
```

---

1. These functions require the return type “void.”

---

## *Video: Band-combine Functions*<sup>1</sup>

---

```
mlib_VideoColorBGRint_to_ABGRint(mlib_u32 *ABGR, mlib_u8 *BGR, mlib_u8 *A_array, mlib_s32 A_const, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 slb, mlib_s32 alb)
```

```
mlib_VideoColorRGBint_to_ABGRint(mlib_u32 *ABGR, mlib_u8 *RGB, mlib_u8 *A_array, mlib_s32 A_const, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 slb, mlib_s32 alb)
```

```
mlib_VideoColorRGBseq_to_ABGRint(mlib_u32 *ABGR, mlib_u8 *R, mlib_u8 *G, mlib_u8 *B, mlib_u8 *A_array, mlib_s32 A_const, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 slb)
```

```
mlib_VideoColorRGBXint_to_ABGRint(mlib_u32 *ABGR, mlib_u8 *RGBX, mlib_u8 *A_array, mlib_s32 A_const, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 slb, mlib_s32 alb)
```

```
mlib_VideoColorRGBXint_to_ARGBint((mlib_u32 *ARGB, mlib_u8 *RGBX, mlib_u8 *A_array, mlib_s32 A_const, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 slb, mlib_s32 alb)
```

```
mlib_VideoColorXRGBint_to_ABGRint(mlib_u32 *ABGR, mlib_u8 *XRGB, mlib_u8 *A_array, mlib_s32 A_const, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 slb, mlib_s32 alb)
```

```
mlib_VideoColorXRGBint_to_ARGBint(mlib_u32 *ARGB, mlib_u8 *XRGB, mlib_u8 *A_array, mlib_s32 A_const, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 slb, mlib_s32 alb)
```

```
mlib_VideoColorYUV411seq_to_UYVY422int(mlib_u32 *UYVY, mlib_u8 *Y, mlib_u8 *U, mlib_u8 *V, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 ylb, mlib_s32 uvlb)
```

---

## *Video: Band-combine Functions (Continued)*<sup>1</sup>

**mllib\_VideoColorYUV411seq\_to\_YUYV422int**(mllib\_u32 \*YUYV, mllib\_u8 \*Y, mllib\_u8 \*U, mllib\_u8 \*V, mllib\_s32 w, mllib\_s32 h, mllib\_s32 dlb, mllib\_s32 ylb, mllib\_s32 uvlb)

**mllib\_VideoColorYUV420seq\_to\_UYVY422int**(mllib\_u32 \*UYVY, mllib\_u8 \*Y, mllib\_u8 \*U, mllib\_u8 \*V, mllib\_s32 w, mllib\_s32 h, mllib\_s32 dlb, mllib\_s32 ylb, mllib\_s32 uvlb)

**mllib\_VideoColorYUV420seq\_to\_YUYV422int**(mllib\_u32 \*yuyv, mllib\_u8 \*y, mllib\_u8 \*u, mllib\_u8 \*v, mllib\_s32 width, mllib\_s32 height, mllib\_s32 dst\_stride, mllib\_s32 y\_stride, mllib\_s32 uv\_stride)

**mllib\_VideoColorYUV422seq\_to\_UYVY422int**(mllib\_u32 \*UYVY, mllib\_u8 \*Y, mllib\_u8 \*U, mllib\_u8 \*V, mllib\_s32 w, mllib\_s32 h, mllib\_s32 dlb, mllib\_s32 ylb, mllib\_s32 uvlb)

**mllib\_VideoColorYUV422seq\_to\_YUYV422int**(mllib\_u32 \*YUYV, mllib\_u8 \*Y, mllib\_u8 \*U, mllib\_u8 \*V, mllib\_s32 w, mllib\_s32 h, mllib\_s32 dlb, mllib\_s32 ylb, mllib\_s32 uvlb)

1. These functions require the return type “void.”

---

## *Video: Band-shuffling Functions*<sup>1</sup>

**mllib\_VideoColorABGRint\_to\_ARGBint**(mllib\_u32 \*ARGB, mllib\_u8 \*ABGR, mllib\_s32 w, mllib\_s32 h, mllib\_s32 dlb, mllib\_s32 slb)

**mllib\_VideoColorBGRAint\_to\_ABGRint**(mllib\_u32 \*ABGR, mllib\_u8 \*BGRA, mllib\_s32 w, mllib\_s32 h, mllib\_s32 dlb, mllib\_s32 slb)

**mllib\_VideoColorRGBAint\_to\_ABGRint**(mllib\_u32 \*ABGR, mllib\_u8 \*RGBA, mllib\_s32 w, mllib\_s32 h, mllib\_s32 dlb, mllib\_s32 slb)

1. These functions require the return type “void.”

---

*Video: Inter-picture Processing Functions —DCT*

---

```
mllib_VideoDCT2x2_S16_S16(mllib_s16 coeffs[4], mllib_s16 block[4])
mllib_VideoDCT4x4_S16_S16(mllib_s16 coeffs[16], mllib_s16 block[16])
mllib_VideoDCT8x8_S16_S16(mllib_s16 coeffs[64], mllib_s16 block[64])
mllib_VideoDCT8x8_S16_S16_B12(mllib_s16 coeffs[64], mllib_s16 block[64])
mllib_VideoDCT8x8_S16_S16_NA(mllib_s16 coeffs[64], mllib_s16 block[64])
mllib_VideoDCT16x16_S16_S16(mllib_s16 coeffs[256], mllib_s16 block[256])
mllib_VideoDCT16x16_S16_S16_B10(mllib_s16 coeffs[256], mllib_s16 block[256])
mllib_VideoDeQuantize_S16(mllib_s16 icoeffs[64], mllib_d64 dqtable[64])
mllib_VideoDeQuantizeInit_S16(mllib_d64 dqtable[64], mllib_s16 iqtable[64])
mllib_VideoQuantize_S16(mllib_s16 icoeffs[64], mllib_d64 dqtable[64])
mllib_VideoQuantizeInit_S16(mllib_d64 dqtable[64], mllib_s16 iqtable[64])
```

---

---

### *Video: Inter-picture Processing Functions —IDCT*

---

```
mlib_VideoIDCT_IEEE_S16_S16(mlib_s16 block[64], mlib_s16 coeffs[64])
mlib_VideoIDCT8x8_S16_S16(mlib_s16 block[64], mlib_s16 coeffs[64])
mlib_VideoIDCT8x8_S16_S16_DC(mlib_s16 block[64], mlib_s16 coeffs[64])
mlib_VideoIDCT8x8_S16_S16_NA(mlib_s16 block[64], mlib_s16 coeffs[64])
mlib_VideoIDCT8x8_S16_S16_Q1(mlib_s16 block[64], mlib_s16 coeffs[64])
```

---

---

### *Video: Inter-picture Processing Functions —Motion Compensation*

---

```
mlib_VideoAddBlock_U8_S16(mlib_u8 *curr_block, mlib_s16 *mc_block, mlib_s32 stride)
mlib_VideoCopyRef_S16_U8(mlib_s16 *mc_block, mlib_u8 *ref_block, mlib_s32 width, mlib_s32 height, mlib_s32 stride)
mlib_VideoCopyRef_S16_U8_[16x16|16x8|8x16|8x8|8x4](mlib_s16 *mc_block, mlib_u8 *ref_block, mlib_s32 stride)
mlib_VideoCopyRef_U8_U8(mlib_u8 *curr_block, mlib_u8 *ref_block, mlib_s32 width, mlib_s32 height, mlib_s32 stride)
mlib_VideoCopyRef_U8_U8_[16x16|16x8|8x16|8x8|8x4](mlib_u8 *curr_block, mlib_u8 *ref_block, mlib_s32 width,
    mlib_s32 height, mlib_s32 stride)
mlib_VideoCopyRefAve_U8_U8(mlib_u8 *curr_block, mlib_u8 *ref_block, mlib_s32 width, mlib_s32 height, mlib_s32 stride)
mlib_VideoCopyRefAve_U8_U8_[16x16|16x8|8x16|8x8|8x4](mlib_u8 *curr_block, mlib_u8 *ref_block, mlib_s32 width,
    mlib_s32 height, mlib_s32 stride)
```

---

***Video: Inter-picture Processing Functions —Motion Compensation (Continued)***


---

```
mlib_VideoH263OverlappedMC_S16_U8(mlib_s16 mc_block[64], mlib_u8 *ref_block, mlib_s32 mch, mlib_s32 mcv, mlib_s32 mah,
  mlib_s32 mav, mlib_s32 mbh, mlib_s32 mbv, mlib_s32 mlh, mlib_s32 mlv, mlib_s32 mrh, mlib_s32 mrv,
  mlib_s32 ref_stride)
```

```
mlib_VideoH263OverlappedMC_U8_U8(mlib_u8 *curr_block, mlib_u8 *ref_block, mlib_s32 mch, mlib_s32 mcv, mlib_s32 mah,
  mlib_s32 mav, mlib_s32 mbh, mlib_s32 mbv, mlib_s32 mlh, mlib_s32 mlv, mlib_s32 mrh, mlib_s32 mrv,
  mlib_s32 curr_stride, mlib_s32 ref_stride)
```

```
mlib_VideoInterpAveX_U8_U8(mlib_u8 *curr_block, mlib_u8 *ref_block, mlib_s32 width, mlib_s32 height,
  mlib_s32 frm_stride, mlib_s32 fld_stride)
```

```
mlib_VideoInterpAveX_U8_U8_[16x16|16x8|8x16|8x8|8x4](mlib_u8 *curr_block, mlib_u8 *ref_block, mlib_s32 frm_stride,
  mlib_s32 fld_stride)
```

```
mlib_VideoInterpAveXY_U8_U8(mlib_u8 *curr_block, mlib_u8 *ref_block, mlib_s32 width, mlib_s32 height,
  mlib_s32 frm_stride, mlib_s32 fld_stride)
```

```
mlib_VideoInterpAveXY_U8_U8_[16x16|16x8|8x16|8x8|8x4](mlib_u8 *curr_block, mlib_u8 *ref_block, mlib_s32 frm_stride,
  mlib_s32 fld_stride)
```

```
mlib_VideoInterpAveY_U8_U8(mlib_u8 *curr_block, mlib_u8 *ref_block, mlib_s32 width, mlib_s32 height,
  mlib_s32 frm_stride, mlib_s32 fld_stride)
```

```
mlib_VideoInterpAveY_U8_U8_[16x16|16x8|8x16|8x8|8x4](mlib_u8 *curr_block, mlib_u8 *ref_block, mlib_s32 frm_stride,
  mlib_s32 fld_stride)
```

---

## *Video: Inter-picture Processing Functions —Motion Compensation (Continued)*

---

**mlib\_VideoInterpX\_S16\_U8**(mlib\_s16 \*mc\_block, mlib\_u8 \*ref\_block, mlib\_s32 width, mlib\_s32 height, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

**mlib\_VideoInterpX\_S16\_U8\_[16x16|16x8|8x16|8x8|8x4]**(mlib\_s16 \*mc\_block, mlib\_u8 \*ref\_block, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

**mlib\_VideoInterpX\_U8\_U8**(mlib\_u8 \*curr\_block, mlib\_u8 \*ref\_block, mlib\_s32 width, mlib\_s32 height, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

**mlib\_VideoInterpX\_U8\_U8\_[16x16|16x8|8x16|8x8|8x4]**(mlib\_u8 \*curr\_block, mlib\_u8 \*ref\_block, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

**mlib\_VideoInterpXY\_S16\_U8**(mlib\_s16 \*mc\_block, mlib\_u8 \*ref\_block, mlib\_s32 width, mlib\_s32 height, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

**mlib\_VideoInterpXY\_S16\_U8\_[16x16|16x8|8x16|8x8|8x4]**(mlib\_s16 \*mc\_block, mlib\_u8 \*ref\_block, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

**mlib\_VideoInterpXY\_U8\_U8**(mlib\_u8 \*curr\_block, mlib\_u8 \*ref\_block, mlib\_s32 width, mlib\_s32 height, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

**mlib\_VideoInterpXY\_U8\_U8\_[16x16|16x8|8x16|8x8|8x4]**(mlib\_u8 \*curr\_block, mlib\_u8 \*ref\_block, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

**mlib\_VideoInterpY\_S16\_U8**(mlib\_s16 \*mc\_block, mlib\_u8 \*ref\_block, mlib\_s32 width, mlib\_s32 height, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

---

### *Video: Inter-picture Processing Functions —Motion Compensation (Continued)*

---

**mlib\_VideoInterpY\_S16\_U8\_[16x16|16x8|8x16|8x8|8x4]**(mlib\_s16 \*mc\_block, mlib\_u8 \*ref\_block, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

**mlib\_VideoInterpY\_U8\_U8**(mlib\_u8 \*curr\_block, mlib\_u8 \*ref\_block, mlib\_s32 width, mlib\_s32 height, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

**mlib\_VideoInterpY\_U8\_U8\_[16x16|16x8|8x16|8x8|8x4]**(mlib\_u8 \*curr\_block, mlib\_u8 \*ref\_block, mlib\_s32 frm\_stride, mlib\_s32 fld\_stride)

**mlib\_VideoP64Decimate\_U8\_U8**(mlib\_u8 \*dst, mlib\_u8 \*src, mlib\_s32 width, mlib\_s32 height, mlib\_s32 dst\_stride, mlib\_s32 src\_stride)

**mlib\_VideoP64Loop\_S16\_U8**(mlib\_s16 mc\_block[64], mlib\_u8 \*ref\_block, mlib\_s32 stride)

**mlib\_VideoP64Loop\_U8\_U8**(mlib\_u8 \*curr\_block, mlib\_u8 \*ref\_block, mlib\_s32 stride)

---

### *Video: Inter-picture Processing Functions —Motion Estimation*

---

**mlib\_VideoSumAbsDiff**(mlib\_u8 \*curr\_block, mlib\_u8 \*ref\_block, mlib\_s32 width, mlib\_s32 height, mlib\_s32 stride)<sup>1</sup>

---

1. This function requires the return type “mlib\_s32.”

---

## *Video: Intra-picture Processing Functions —Color Conversion*

---

`mlib_VideoColorABGR2RGB(mlib_u8 *rgb, mlib_u8 *abgr, mlib_s32 n)`

`mlib_VideoColorARGB2RGB(mlib_u8 *rgb, mlib_u8 *argb, mlib_s32 n)`

`mlib_VideoColorMerge2(mlib_u8 *colors, mlib_u8 *color1, mlib_u8 *color2, mlib_s32 n)`

`mlib_VideoColorMerge2_S16(mlib_s16 *colors, mlib_s16 *color1, mlib_s16 *color2, mlib_s32 n)`

`mlib_VideoColorMerge3(mlib_u8 *colors, mlib_u8 *color1, mlib_u8 *color2, mlib_u8 *color3, mlib_s32 n)`

`mlib_VideoColorMerge3_S16(mlib_s16 *colors, mlib_s16 *color1, mlib_s16 *color2, mlib_s16 *color3, mlib_s32 n)`

`mlib_VideoColorMerge4(mlib_u8 *colors, mlib_u8 *color1, mlib_u8 *color2, mlib_u8 *color3, mlib_u8 *color4, mlib_s32 n)`

`mlib_VideoColorMerge4_S16(mlib_s16 *colors, mlib_s16 *color1, mlib_s16 *color2, mlib_s16 *color3, mlib_s16 *color4,  
mlib_s32 n)`

`mlib_VideoColorRGB2ABGR(mlib_u8 *abgr, mlib_u8 *rgb, mlib_s32 n)`

`mlib_VideoColorRGB2ARGB(mlib_u8 *argb, mlib_u8 *rgb, mlib_s32 n)`

`mlib_VideoColorSplit2(mlib_u8 *color1, mlib_u8 *color2, mlib_u8 *colors, mlib_s32 n)`

`mlib_VideoColorSplit2_S16(mlib_s16 *color1, mlib_s16 *color2, mlib_s16 *colors, mlib_s32 n)`

`mlib_VideoColorSplit3(mlib_u8 *color1, mlib_u8 *color2, mlib_u8 *color3, mlib_u8 *colors, mlib_s32 n)`

`mlib_VideoColorSplit3_S16(mlib_s16 *color1, mlib_s16 *color2, mlib_s16 *color3, mlib_s16 *colors, mlib_s32 n)`

`mlib_VideoColorSplit4(mlib_u8 *color1, mlib_u8 *color2, mlib_u8 *color3, mlib_u8 *color4, mlib_u8 *colors, mlib_s32 n)`

---

**Video: Intra-picture Processing Functions —Color Conversion (Continued)**

---

**mlib\_VideoColorSplit4\_S16**(mlib\_s16 \*color1, mlib\_s16 \*color2, mlib\_s16 \*color3, mlib\_s16 \*color4, mlib\_s16 \*colors, mlib\_s32 n)

**mlib\_VideoColorUYV444int\_to\_ABGRint**(mlib\_u32 \*ABGR, mlib\_u8 \*UYV, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 slb, mlib\_s32 alb)<sup>1</sup>

**mlib\_VideoColorUYV444int\_to\_ARGBint**(mlib\_u32 \*ARGB, mlib\_u8 \*UYV, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 slb, mlib\_s32 alb)<sup>1</sup>

**mlib\_VideoColorUYVY422int\_to\_ABGRint**(mlib\_u32 \*ABGR, mlib\_u8 \*UYVY, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 slb, mlib\_s32 alb)<sup>1</sup>

**mlib\_VideoColorUYVY422int\_to\_ARGBint**(mlib\_u32 \*ARGB, mlib\_u8 \*UYVY, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 slb, mlib\_s32 alb)<sup>1</sup>

**mlib\_VideoColorYUV2ABGR411**(mlib\_u8 \*abgr, mlib\_u8 \*y, mlib\_u8 \*u, mlib\_u8 \*v, mlib\_s32 width, mlib\_s32 height, mlib\_s32 rgb\_stride, mlib\_s32 y\_stride, mlib\_s32 uv\_stride)

**mlib\_VideoColorYUV2ABGR420**(mlib\_u8 \*abgr, mlib\_u8 \*y, mlib\_u8 \*u, mlib\_u8 \*v, mlib\_s32 width, mlib\_s32 height, mlib\_s32 rgb\_stride, mlib\_s32 y\_stride, mlib\_s32 uv\_stride)

**mlib\_VideoColorYUV2ABGR420\_W**(mlib\_u8 \*abgr, mlib\_u8 \*y, mlib\_u8 \*u, mlib\_u8 \*v, mlib\_s32 width, mlib\_s32 height, mlib\_s32 abgr\_stride, mlib\_s32 y\_stride, mlib\_s32 uv\_stride, mlib\_s32 left, mlib\_s32 top, mlib\_s32 right, mlib\_s32 bottom)

---

## *Video: Intra-picture Processing Functions —Color Conversion (Continued)*

---

**mllib\_VideoColorYUV2ABGR420\_WX2**(mllib\_u8 \*abgr, mllib\_u8 \*y, mllib\_u8 \*u, mllib\_u8 \*v, mllib\_s32 width, mllib\_s32 height, mllib\_s32 abgr\_stride, mllib\_s32 y\_stride, mllib\_s32 uv\_stride, mllib\_s32 left, mllib\_s32 top, mllib\_s32 right, mllib\_s32 bottom)

**mllib\_VideoColorYUV2ABGR420\_WX3**(mllib\_u8 \*abgr, mllib\_u8 \*y, mllib\_u8 \*u, mllib\_u8 \*v, mllib\_s32 width, mllib\_s32 height, mllib\_s32 abgr\_stride, mllib\_s32 y\_stride, mllib\_s32 uv\_stride, mllib\_s32 left, mllib\_s32 top, mllib\_s32 right, mllib\_s32 bottom)

**mllib\_VideoColorYUV2ABGR420\_X2**(mllib\_u8 \*abgr, mllib\_u8 \*y, mllib\_u8 \*u, mllib\_u8 \*v, mllib\_s32 width, mllib\_s32 height, mllib\_s32 abgr\_stride, mllib\_s32 y\_stride, mllib\_s32 uv\_stride)

**mllib\_VideoColorYUV2ABGR420\_X3**(mllib\_u8 \*abgr, mllib\_u8 \*y, mllib\_u8 \*u, mllib\_u8 \*v, mllib\_s32 width, mllib\_s32 height, mllib\_s32 abgr\_stride, mllib\_s32 y\_stride, mllib\_s32 uv\_stride)

**mllib\_VideoColorYUV2ABGR422**(mllib\_u8 \*abgr, mllib\_u8 \*y, mllib\_u8 \*u, mllib\_u8 \*v, mllib\_s32 width, mllib\_s32 height, mllib\_s32 rgb\_stride, mllib\_s32 y\_stride, mllib\_s32 uv\_stride)

**mllib\_VideoColorYUV2ABGR444**(mllib\_u8 \*abgr, mllib\_u8 \*y, mllib\_u8 \*u, mllib\_u8 \*v, mllib\_s32 width, mllib\_s32 height, mllib\_s32 rgb\_stride, mllib\_s32 yuv\_stride)

**mllib\_VideoColorYUV2ARGB411**(mllib\_u8 \*argb, mllib\_u8 \*y, mllib\_u8 \*u, mllib\_u8 \*v, mllib\_s32 width, mllib\_s32 height, mllib\_s32 rgb\_stride, mllib\_s32 y\_stride, mllib\_s32 uv\_stride)

**mllib\_VideoColorYUV2ARGB420**(mllib\_u8 \*argb, mllib\_u8 \*y, mllib\_u8 \*u, mllib\_u8 \*v, mllib\_s32 width, mllib\_s32 height, mllib\_s32 rgb\_stride, mllib\_s32 y\_stride, mllib\_s32 uv\_stride)

---

**Video: Intra-picture Processing Functions —Color Conversion (Continued)**


---

**mlib\_VideoColorYUV2ARGB422**(mlib\_u8 \*argb, mlib\_u8 \*y, mlib\_u8 \*u, mlib\_u8 \*v, mlib\_s32 width, mlib\_s32 height, mlib\_s32 rgb\_stride, mlib\_s32 y\_stride, mlib\_s32 uv\_stride)

**mlib\_VideoColorYUV2ARGB444**(mlib\_u8 \*argb, mlib\_u8 \*y, mlib\_u8 \*u, mlib\_u8 \*v, mlib\_s32 width, mlib\_s32 height, mlib\_s32 rgb\_stride, mlib\_s32 yuv\_stride)

**mlib\_VideoColorYUV2RGB411**(mlib\_u8 \*rgb, mlib\_u8 \*y, mlib\_u8 \*u, mlib\_u8 \*v, mlib\_s32 width, mlib\_s32 height, mlib\_s32 rgb\_stride, mlib\_s32 y\_stride, mlib\_s32 uv\_stride)

**mlib\_VideoColorYUV2RGB420**(mlib\_u8 \*rgb, mlib\_u8 \*y, mlib\_u8 \*u, mlib\_u8 \*v, mlib\_s32 width, mlib\_s32 height, mlib\_s32 rgb\_stride, mlib\_s32 y\_stride, mlib\_s32 uv\_stride)

**mlib\_VideoColorYUV2RGB422**(mlib\_u8 \*rgb, mlib\_u8 \*y, mlib\_u8 \*u, mlib\_u8 \*v, mlib\_s32 width, mlib\_s32 height, mlib\_s32 rgb\_stride, mlib\_s32 y\_stride, mlib\_s32 uv\_stride)

**mlib\_VideoColorYUV2RGB444**(mlib\_u8 \*rgb, mlib\_u8 \*y, mlib\_u8 \*u, mlib\_u8 \*v, mlib\_s32 width, mlib\_s32 height, mlib\_s32 rgb\_stride, mlib\_s32 yuv\_stride)

**mlib\_VideoColorYUV411seq\_to\_ABGRint**(mlib\_u32 \*ABGR, mlib\_u8 \*Y, mlib\_u8 \*U, mlib\_u8 \*V, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 yalb, mlib\_s32 uvlb)<sup>1</sup>

**mlib\_VideoColorYUV411seq\_to\_ARGBint**(mlib\_u32 \*ARGB, mlib\_u8 \*Y, mlib\_u8 \*U, mlib\_u8 \*V, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 yalb, mlib\_s32 uvlb)<sup>1</sup>

**mlib\_VideoColorYUV420seq\_to\_ABGRint**(mlib\_u32 \*ABGR, mlib\_u8 \*Y, mlib\_u8 \*U, mlib\_u8 \*V, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 yalb, mlib\_s32 uvlb)<sup>1</sup>

## *Video: Intra-picture Processing Functions —Color Conversion (Continued)*

**mlib\_VideoColorYUV420seq\_to\_ARGBint**(mlib\_u32 \*ARGB, mlib\_u8 \*Y, mlib\_u8 \*U, mlib\_u8 \*V, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 yalb, mlib\_s32 uvlb)<sup>1</sup>

**mlib\_VideoColorYUV422seq\_to\_ABGRint**(mlib\_u32 \*ABGR, mlib\_u8 \*Y, mlib\_u8 \*U, mlib\_u8 \*V, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 yalb, mlib\_s32 uvlb)<sup>1</sup>

**mlib\_VideoColorYUV422seq\_to\_ARGBint**(mlib\_u32 \*ARGB, mlib\_u8 \*Y, mlib\_u8 \*U, mlib\_u8 \*V, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 yalb, mlib\_s32 uvlb)<sup>1</sup>

**mlib\_VideoColorYUV444int\_to\_ABGRint**(mlib\_u32 \*ABGR, mlib\_u8 \*YUV, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 slb, mlib\_s32 alb)<sup>1</sup>

**mlib\_VideoColorYUV444int\_to\_ARGBint**(mlib\_u32 \*ARGB, mlib\_u8 \*YUV, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 slb, mlib\_s32 alb)<sup>1</sup>

**mlib\_VideoColorYUV444seq\_to\_ABGRint**(mlib\_u32 \*ABGR, mlib\_u8 \*Y, mlib\_u8 \*U, mlib\_u8 \*V, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 slb)<sup>1</sup>

**mlib\_VideoColorYUV444seq\_to\_ARGBint**(mlib\_u32 \*ARGB, mlib\_u8 \*Y, mlib\_u8 \*U, mlib\_u8 \*V, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 slb)<sup>1</sup>

**mlib\_VideoColorYUYV422int\_to\_ABGRint**(mlib\_u32 \*ABGR, mlib\_u8 \*YUYV, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 slb, mlib\_s32 alb)<sup>1</sup>

**mlib\_VideoColorYUYV422int\_to\_ARGBint**(mlib\_u32 \*ARGB, mlib\_u8 \*YUYV, mlib\_u8 \*A\_array, mlib\_s32 A\_const, mlib\_s32 w, mlib\_s32 h, mlib\_s32 dlb, mlib\_s32 slb, mlib\_s32 alb)<sup>1</sup>

1. This function requires the return type “void.”

---

### *Video: Intra-picture Processing Functions —DCT*

---

```
mlib_VideoDCT8x8_S16_U8(mlib_s16 coeffs[64], mlib_u8 *block, mlib_s32 stride)
```

```
mlib_VideoDCT8x8_S16_U8_NA(mlib_s16 coeffs[64], mlib_u8 *block, mlib_s32 stride)
```

---

### *Video: Intra-picture Processing Functions —IDCT*

---

```
mlib_VideoIDCT8x8_U8_S16(mlib_u8 *block, mlib_s16 coeffs[64], mlib_s32 stride)
```

```
mlib_VideoIDCT8x8_U8_S16_DC(mlib_u8 *block, mlib_s16 coeffs[64])
```

```
mlib_VideoIDCT8x8_U8_S16_NA(mlib_u8 *block, mlib_s16 coeffs[64], mlib_s32 stride)
```

```
mlib_VideoIDCT8x8_U8_S16_Q1(mlib_u8 *block, mlib_s16 coeffs[64], mlib_s32 stride)
```

---

### *Video: JPEG File Interchange Format (JFIF) Processing Functions*

---

```
mlib_VideoColorABGR2JFIFYCC420(mlib_u8 *y0, mlib_u8 *y1, mlib_u8 *cb, mlib_u8 *cr, mlib_u8 *abgr0, mlib_u8 *abgr1,
    mlib_s32 n)
```

```
mlib_VideoColorABGR2JFIFYCC422(mlib_u8 *y, mlib_u8 *cb, mlib_u8 *cr, mlib_u8 *abgr, mlib_s32 n)
```

```
mlib_VideoColorABGR2JFIFYCC444(mlib_u8 *y, mlib_u8 *cb, mlib_u8 *cr, mlib_u8 *abgr, mlib_s32 n)
```

---

---

## *Video: JPEG File Interchange Format (JFIF) Processing Functions (Continued)*

---

**mlib\_VideoColorARBG2JFIFYCC444**(mlib\_u8 \*y, mlib\_u8 \*cb, mlib\_u8 \*cr, mlib\_u8 \*argb, mlib\_s32 n)

**mlib\_VideoColorARBG2JFIFYCC420**(mlib\_u8 \*y0, mlib\_u8 \*y1, mlib\_u8 \*cb, mlib\_u8 \*cr, mlib\_u8 \*argb0, mlib\_u8 \*argb1, mlib\_s32 n)

**mlib\_VideoColorARBG2JFIFYCC422**(mlib\_u8 \*y, mlib\_u8 \*cb, mlib\_u8 \*cr, mlib\_u8 \*argb, mlib\_s32 n)

**mlib\_VideoColorCMYK2JFIFYCCK444**(mlib\_u8 \*y, mlib\_u8 \*cb, mlib\_u8 \*cr, mlib\_u8 \*k, mlib\_u8 \*cmyk, mlib\_s32 n)

**mlib\_VideoColorJFIFYCC2ABGR444**(mlib\_u8 \*abgr, mlib\_u8 \*y, mlib\_u8 \*cb, mlib\_u8 \*cr, mlib\_s32 n)

**mlib\_VideoColorJFIFYCC2ARGB444**(mlib\_u8 \*argb, mlib\_u8 \*y, mlib\_u8 \*cb, mlib\_u8 \*cr, mlib\_s32 n)

**mlib\_VideoColorJFIFYCC2RGB420**(mlib\_u8 \*rgb0, mlib\_u8 \*rgb1, mlib\_u8 \*y0, mlib\_u8 \*y1, mlib\_u8 \*cb0, mlib\_u8 \*cr0, mlib\_u8 \*cb1, mlib\_u8 \*cr1, mlib\_u8 \*cb2, mlib\_u8 \*cr2, mlib\_s32 n)

**mlib\_VideoColorJFIFYCC2RGB420\_Nearest**(mlib\_u8 \*rgb0, mlib\_u8 \*rgb1, mlib\_u8 \*y0, mlib\_u8 \*y1, mlib\_u8 \*cb, mlib\_u8 \*cr, mlib\_s32 n)

**mlib\_VideoColorJFIFYCC2RGB422**(mlib\_u8 \*rgb, mlib\_u8 \*y, mlib\_u8 \*cb, mlib\_u8 \*cr, mlib\_s32 n)

**mlib\_VideoColorJFIFYCC2RGB422\_Nearest**(mlib\_u8 \*rgb, mlib\_u8 \*y, mlib\_u8 \*cb, mlib\_u8 \*cr, mlib\_s32 n)

**mlib\_VideoColorJFIFYCC2RGB444**(mlib\_u8 \*rgb, mlib\_u8 \*y, mlib\_u8 \*cb, mlib\_u8 \*cr, mlib\_s32 n)

**mlib\_VideoColorJFIFYCC2RGB444\_S16**(mlib\_s16 \*rgb, mlib\_s16 \*y, mlib\_s16 \*cb, mlib\_s16 \*cr, mlib\_s32 n)

**mlib\_VideoColorJFIFYCCK2CMYK444**(mlib\_u8 \*cmyk, mlib\_u8 \*y, mlib\_u8 \*cb, mlib\_u8 \*cr, mlib\_u8 \*k, mlib\_s32 n)

---

*Video: JPEG File Interchange Format (JFIF) Processing Functions (Continued)*

---

```
mllib_VideoColorRGB2JFIFYCC420(mllib_u8 *y0, mllib_u8 *y1, mllib_u8 *cb, mllib_u8 *cr, mllib_u8 *rgb0, mllib_u8 *rgb1,
    mllib_s32 n)
```

```
mllib_VideoColorRGB2JFIFYCC422(mllib_u8 *y, mllib_u8 *cb, mllib_u8 *cr, mllib_u8 *rgb, mllib_s32 n)
```

```
mllib_VideoColorRGB2JFIFYCC444(mllib_u8 *y, mllib_u8 *cb, mllib_u8 *cr, mllib_u8 *rgb, mllib_s32 n)
```

```
mllib_VideoColorRGB2JFIFYCC444_S16(mllib_s16 *y, mllib_s16 *cb, mllib_s16 *cr, mllib_s16 *rgb, mllib_s32 n)
```

```
mllib_VideoDownSample420(mllib_u8 *dst, mllib_u8 *src0, mllib_u8 *src1, mllib_s32 n)
```

```
mllib_VideoDownSample420_S16(mllib_s16 *dst, mllib_s16 *src0, mllib_s16 *src1, mllib_s32 n)
```

```
mllib_VideoDownSample422(mllib_u8 *dst, mllib_u8 *src, mllib_s32 n)
```

```
mllib_VideoDownSample422_S16(mllib_s16 *dst, mllib_s16 *src, mllib_s32 n)
```

```
mllib_VideoUpSample420(mllib_u8 *dst0, mllib_u8 *dst1, mllib_u8 *src0, mllib_u8 *src1, mllib_u8 *src2, mllib_s32 n)
```

```
mllib_VideoUpSample420_Nearest(mllib_u8 *dst0, mllib_u8 *dst1, mllib_u8 *src, mllib_s32 n)
```

```
mllib_VideoUpSample420_Nearest(mllib_u8 *dst0, mllib_u8 *dst1, mllib_u8 *src, mllib_s32 n)
```

```
mllib_VideoUpSample420_Nearest_S16(mllib_s16 *dst, mllib_s16 *src, mllib_s32 n)
```

```
mllib_VideoUpSample420_S16(mllib_s16 *dst, mllib_s16 *src, mllib_s32 n)
```

```
mllib_VideoUpSample422_Nearest(mllib_u8 *dst, mllib_u8 *src, mllib_s32 n)
```

```
mllib_VideoUpSample422_Nearest_S16(mllib_s16 *dst, mllib_s16 *src, mllib_s32 n)
```

```
mllib_VideoUpSample422_S16(mllib_s16 *dst, mllib_s16 *src, mllib_s32 n)
```

---

---

## *Video: Resizing Functions*

```
mlib_VideoColorResizeABGR(mlib_u32 *dst, mlib_u8 *src, mlib_s32 dst_w, mlib_s32 dst_h, mlib_s32 dst_lb, mlib_s32 src_w,  
    mlib_s32 src_h, mlib_s32 src_lb, mlib_filter filter)1
```

1. This function requires the return type “void.”

---

## *Video: Wavelet Color Space Conversion Functions*

```
mlib_VideoReversibleColorRGB2YUV_S16_S16(mlib_s16 *y, mlib_s16 *u, mlib_s16 *v, mlib_s16 *r, mlib_s16 *g, mlib_s16 *b,  
    mlib_s32 n, mlib_s32 depth)
```

```
mlib_VideoReversibleColorRGB2YUV_S16_U8(mlib_s16 *y, mlib_s16 *u, mlib_s16 *v, mlib_u8 *r, mlib_u8 *g, mlib_u8 *b,  
    mlib_s32 n, mlib_s32 depth)
```

```
mlib_VideoReversibleColorRGB2YUV_S32_S16(mlib_s32 *y, mlib_s32 *u, mlib_s32 *v, mlib_s16 *r, mlib_s16 *g, mlib_s16 *b,  
    mlib_s32 n, mlib_s32 depth)
```

```
mlib_VideoReversibleColorRGB2YUV_U8_U8(mlib_u8 *y, mlib_u8 *u, mlib_u8 *v, mlib_u8 *r, mlib_u8 *g, mlib_u8 *b,  
    mlib_s32 n, mlib_s32 depth)
```

```
mlib_VideoReversibleColorYUV2RGB_S16_S16(mlib_s16 *r, mlib_s16 *g, mlib_s16 *b, mlib_s16 *y, mlib_s16 *u, mlib_s16 *v,  
    mlib_s32 n, mlib_s32 depth)
```

---

*Video: Wavelet Color Space Conversion Functions (Continued)*

---

```
mlib_VideoReversibleColorYUV2RGB_S16_S32(mlib_s16 *r, mlib_s16 *g, mlib_s16 *b, mlib_s32 *y, mlib_s32 *u, mlib_s32 *v,  
      mlib_s32 n, mlib_s32 depth)
```

```
mlib_VideoReversibleColorYUV2RGB_U8_U8(mlib_u8 *r, mlib_u8 *g, mlib_u8 *b, mlib_u8 *y, mlib_u8 *u, mlib_u8 *v,  
      mlib_s32 n, mlib_s32 depth)
```

```
mlib_VideoReversibleColorYUV2RGB_U8_S16(mlib_u8 *r, mlib_u8 *g, mlib_u8 *b, mlib_s16 *y, mlib_s16 *u, mlib_s16 *v,  
      mlib_s32 n, mlib_s32 depth)
```

---

---

*Video: Wavelet Sign-magnitude Conversion Functions*

---

```
mlib_VideoSignMagnitudeConvert_S16(mlib_s16 *srcdst, mlib_s32 n)
```

```
mlib_VideoSignMagnitudeConvert_S16_S16(mlib_s16 *dst, mlib_s16 *src, mlib_s32 n)
```

```
mlib_VideoSignMagnitudeConvert_S32(mlib_s32 *srcdst, mlib_s32 n)
```

```
mlib_VideoSignMagnitudeConvert_S32_S32(mlib_s32 *dst, mlib_s32 *src, mlib_s32 n)
```

---

---

### *Video: Wavelet Transformation Functions—Forward*

---

```
mlib_VideoWaveletForwardTwoTenTrans_S16_S16(mlib_s16 *dst, mlib_s16 *src, mlib_s32 width, mlib_s32 height,  
        mlib_s32 *level)
```

```
mlib_VideoWaveletForwardTwoTenTrans_S16_U8(mlib_s16 *dst, mlib_u8 *src, mlib_s32 width, mlib_s32 height,  
        mlib_s32 *level)
```

```
mlib_VideoWaveletForwardTwoTenTrans_S32_S16(mlib_s32 *dst, mlib_s16 *src, mlib_s32 width, mlib_s32 height,  
        mlib_s32 *level)
```

```
mlib_VideoWaveletForwardTwoTenTrans_S32_S32(mlib_s32 *dst, mlib_s32 *src, mlib_s32 width, mlib_s32 height,  
        mlib_s32 *level)
```

---

---

### *Video: Wavelet Transformation Functions—Inverse*

---

```
mlib_VideoWaveletInverseTwoTenTrans_S16_S16(mlib_s16 *dst, mlib_s16 *src, mlib_s32 width, mlib_s32 height,  
        mlib_s32 *level)
```

```
mlib_VideoWaveletInverseTwoTenTrans_S16_S32(mlib_s16 *dst, mlib_s32 *src, mlib_s32 width, mlib_s32 height,  
        mlib_s32 *level)
```

---

### *Video: Wavelet Transformation Functions—Inverse (Continued)*

---

```
mlib_VideoWaveletInverseTwoTenTrans_S32_S32(mlib_s32 *dst, mlib_s32 *src, mlib_s32 width, mlib_s32 height,
        mlib_s32 *level)
```

```
mlib_VideoWaveletInverseTwoTenTrans_U8_S16(mlib_u8 *dst, mlib_s16 *src, mlib_s32 width, mlib_s32 height,
        mlib_s32 *level)
```

---

### *Video: YUV Subsampling Functions*<sup>1</sup>

---

```
mlib_VideoColorUYV444int_to_UYVY422int(mlib_u32 *UYVY, mlib_u8 *UYV, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 slb)
```

```
mlib_VideoColorUYV444int_to_YUYV422int(mlib_u32 *YUYV, mlib_u8 *UYV, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 slb)
```

```
mlib_VideoColorYUV444int_to_UYVY422int(mlib_u32 *UYVY, mlib_u8 *YUV, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 slb)
```

```
mlib_VideoColorYUV444int_to_YUYV422int(mlib_u32 *YUYV, mlib_u8 *YUV, mlib_s32 w, mlib_s32 h, mlib_s32 dlb, mlib_s32 slb)
```

```
mlib_VideoColorYUV444seq_to_UYVY422int(mlib_u32 *UYVY, mlib_u8 *Y, mlib_u8 *U, mlib_u8 *V, mlib_s32 w, mlib_s32 h,
        mlib_s32 dlb, mlib_s32 slb)
```

```
mlib_VideoColorYUV444seq_to_YUYV422int(mlib_u32 *YUYV, mlib_u8 *Y, mlib_u8 *U, mlib_u8 *V, mlib_s32 w, mlib_s32 h,
        mlib_s32 dlb, mlib_s32 slb)
```

---

1. These functions require the return type “void.”

## GRAPHICS FUNCTIONS AND SYNTAX

---

### *Graphics: Circle and Ellipse Functions*

---

**mllib\_GraphicsDrawCircle\_[8|32]**(mllib\_image \*buffer, mlib\_s16 x, mlib\_s16 y, mlib\_s32 r, mlib\_s32 c)

**mllib\_GraphicsDrawCircle\_A\_[8|32]**(mllib\_image \*buffer, mlib\_s16 x, mlib\_s16 y, mlib\_s32 r, mlib\_s32 c)

**mllib\_GraphicsDrawCircle\_X\_[8|32]**(mllib\_image \*buffer, mlib\_s16 x, mlib\_s16 y, mlib\_s32 r, mlib\_s32 c, mlib\_s32 c2)

**mllib\_GraphicsDrawEllipse\_[8|32]**(mllib\_image \*buffer, mlib\_s16 x, mlib\_s16 y, mlib\_s32 a, mlib\_s32 b, mlib\_f32 t, mlib\_s32 c)

**mllib\_GraphicsDrawEllipse\_A\_[8|32]**(mllib\_image \*buffer, mlib\_s16 x, mlib\_s16 y, mlib\_s32 a, mlib\_s32 b, mlib\_f32 t, mlib\_s32 c)

**mllib\_GraphicsDrawEllipse\_X\_[8|32]**(mllib\_image \*buffer, mlib\_s16 x, mlib\_s16 y, mlib\_s32 a, mlib\_s32 b, mlib\_f32 t, mlib\_s32 c, mlib\_s32 c2)

**mllib\_GraphicsFillCircle\_[8|32]**(mllib\_image \*buffer, mlib\_s16 x, mlib\_s16 y, mlib\_s32 r, mlib\_s32 c)

**mllib\_GraphicsFillCircle\_A\_[8|32]**(mllib\_image \*buffer, mlib\_s16 x, mlib\_s16 y, mlib\_s32 r, mlib\_s32 c)

**mllib\_GraphicsFillCircle\_X\_[8|32]**(mllib\_image \*buffer, mlib\_s16 x, mlib\_s16 y, mlib\_s32 r, mlib\_s32 c, mlib\_s32 c2)

---

*Graphics: Circle and Ellipse Functions (Continued)*

---

```
mllib_GraphicsFillEllipse_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 a, mllib_s32 b, mllib_f32 t,  
    mllib_s32 c)
```

```
mllib_GraphicsFillEllipse_A_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 a, mllib_s32 b, mllib_f32 t,  
    mllib_s32 c)
```

```
mllib_GraphicsFillEllipse_X_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 a, mllib_s32 b, mllib_f32 t,  
    mllib_s32 c, mllib_s32 c2)
```

---

---

*Graphics: Fill Functions*

---

```
mllib_GraphicsBoundaryFill_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 c, mllib_s32 c2)
```

```
mllib_GraphicsFloodFill_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 c, mllib_s32 c2)
```

---

---

## *Graphics: Point, Line, and Arc Functions*

---

```
mllib_GraphicsDrawArc_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 r, mllib_f32 t1, mllib_f32 t2,
    mllib_s32 c)
mllib_GraphicsDrawArc_A_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 r, mllib_f32 t1, mllib_f32 t2,
    mllib_s32 c)
mllib_GraphicsDrawArc_X_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 r, mllib_f32 t1, mllib_f32 t2,
    mllib_s32 c, mllib_s32 c2)
mllib_GraphicsDrawLine_[8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s32 c)
mllib_GraphicsDrawLine_A_[8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s32 c)
mllib_GraphicsDrawLine_AG_[8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s32 c1,
    mllib_s32 c2)
mllib_GraphicsDrawLine_AGZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1,
    mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s32 c1, mllib_s32 c2)
mllib_GraphicsDrawLine_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1,
    mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s32 c)
mllib_GraphicsDrawLine_G_[8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s32 c1,
    mllib_s32 c2)
```

---

*Graphics: Point, Line, and Arc Functions (Continued)*

---

```
mllib_GraphicsDrawLine_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1,
    mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s32 c1, mllib_s32 c2)
```

```
mllib_GraphicsDrawLine_X_[8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s32 c,
    mllib_s32 c2)
```

```
mllib_GraphicsDrawLine_Z_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1,
    mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s32 c)
```

```
mllib_GraphicsDrawLineFanSet_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawLineFanSet_A_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawLineFanSet_AG_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawLineFanSet_AGZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
    mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawLineFanSet_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
    mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawLineFanSet_G_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawLineFanSet_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
    mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawLineFanSet_X_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c,
    mllib_s32 c2)
```

## *Graphics: Point, Line, and Arc Functions (Continued)*

```
mlib_GraphicsDrawLineFanSet_Z[8|32](mlib_image *buffer, mlib_image *zbuffer, mlib_s16 *x, mlib_s16 *y, mlib_s16 *z,
    mlib_s32 npoints, mlib_s32 c)
mlib_GraphicsDrawLineSet[8|32](mlib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c)
mlib_GraphicsDrawLineSet_A[8|32](mlib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c)
mlib_GraphicsDrawLineSet_AG[8|32](mlib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 *c)
mlib_GraphicsDrawLineSet_AGZ[8|32](mlib_image *buffer, mlib_image *zbuffer, mlib_s16 *x, mlib_s16 *y, mlib_s16 *z,
    mlib_s32 npoints, mlib_s32 *c)
mlib_GraphicsDrawLineSet_AZ[8|32](mlib_image *buffer, mlib_image *zbuffer, mlib_s16 *x, mlib_s16 *y, mlib_s16 *z,
    mlib_s32 npoints, mlib_s32 c)
mlib_GraphicsDrawLineSet_G[8|32](mlib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 *c)
mlib_GraphicsDrawLineSet_GZ[8|32](mlib_image *buffer, mlib_image *zbuffer, mlib_s16 *x, mlib_s16 *y, mlib_s16 *z,
    mlib_s32 npoints, mlib_s32 *c)
mlib_GraphicsDrawLineSet_X[8|32](mlib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c,
    mlib_s32 c2)
mlib_GraphicsDrawLineSet_Z[8|32](mlib_image *buffer, mlib_image *zbuffer, mlib_s16 *x, mlib_s16 *y, mlib_s16 *z,
    mlib_s32 npoints, mlib_s32 c)
mlib_GraphicsDrawLineStripSet[8|32](mlib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c)
mlib_GraphicsDrawLineStripSet_A[8|32](mlib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c)
```

---

*Graphics: Point, Line, and Arc Functions (Continued)*

---

```
mllib_GraphicsDrawLineStripSet_AG_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawLineStripSet_AGZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y,  
mllib_s16 *z, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawLineStripSet_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,  
mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawLineStripSet_G_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawLineStripSet_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,  
mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawLineStripSet_X_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c,  
mllib_s32 c2)
```

```
mllib_GraphicsDrawLineStripSet_Z_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,  
mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawPoint_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 c)
```

```
mllib_GraphicsDrawPoint_X_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 c, mllib_s32 c2)
```

```
mllib_GraphicsDrawPointSet_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawPointSet_X_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c,  
mllib_s32 c2)
```

```
mllib_GraphicsDrawPolyline_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

---

## *Graphics: Point, Line, and Arc Functions (Continued)*

---

```
mllib_GraphicsDrawPolyline_A_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c)
mllib_GraphicsDrawPolyline_AG_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 *c)
mllib_GraphicsDrawPolyline_AGZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mlib_s16 *x, mlib_s16 *y, mlib_s16 *z,
    mlib_s32 npoints, mlib_s32 *c)
mllib_GraphicsDrawPolyline_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mlib_s16 *x, mlib_s16 *y, mlib_s16 *z,
    mlib_s32 npoints, mlib_s32 c)
mllib_GraphicsDrawPolyline_G_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 *c)
mllib_GraphicsDrawPolyline_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mlib_s16 *x, mlib_s16 *y, mlib_s16 *z,
    mlib_s32 npoints, mlib_s32 *c)
mllib_GraphicsDrawPolyline_X_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c,
    mlib_s32 c2)
mllib_GraphicsDrawPolyline_Z_[8|32](mllib_image *buffer, mllib_image *zbuffer, mlib_s16 *x, mlib_s16 *y, mlib_s16 *z,
    mlib_s32 npoints, mlib_s32 c)
mllib_GraphicsDrawPolypoint_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c)
mllib_GraphicsDrawPolypoint_X_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c,
    mlib_s32 c2)
```

---

### *Graphics: Point, Line, and Arc Functions (Continued)*

---

```
mllib_GraphicsFillArc_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 r, mllib_f32 t1, mllib_f32 t2, mllib_s32 c)
```

```
mllib_GraphicsFillArc_A_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 r, mllib_f32 t1, mllib_f32 t2, mllib_s32 c)
```

```
mllib_GraphicsFillArc_X_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 r, mllib_f32 t1, mllib_f32 t2, mllib_s32 c, mllib_s32 c2)
```

---

### *Graphics: Polygon Functions*

---

```
mllib_GraphicsDrawPolygon_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawPolygon_A_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawPolygon_AG_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawPolygon_AGZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawPolygon_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawPolygon_G_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
```

---

## *Graphics: Polygon Functions (Continued)*

---

```
mllib_GraphicsDrawPolygon_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawPolygon_X_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c, mllib_s32 c2)
```

```
mllib_GraphicsDrawPolygon_Z_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawRectangle_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 w, mllib_s32 h, mllib_s32 c)
```

```
mllib_GraphicsDrawRectangle_X_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 w, mllib_s32 h, mllib_s32 c, mllib_s32 c2)
```

```
mllib_GraphicsDrawTriangle_[8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s16 x3, mllib_s16 y3, mllib_s32 c)
```

```
mllib_GraphicsDrawTriangle_A_[8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s16 x3, mllib_s16 y3, mllib_s32 c)
```

```
mllib_GraphicsDrawTriangle_AG_[8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s16 x3, mllib_s16 y3, mllib_s32 c1, mllib_s32 c2, mllib_s32 c3)
```

```
mllib_GraphicsDrawTriangle_AGZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1, mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s16 x3, mllib_s16 y3, mllib_s16 z3, mllib_s32 c1, mllib_s32 c2, mllib_s32 c3)
```

---

*Graphics: Polygon Functions (Continued)*

---

```
mllib_GraphicsDrawTriangle_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1, mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s16 x3, mllib_s16 y3, mllib_s16 z3, mllib_s32 c)
```

```
mllib_GraphicsDrawTriangle_G_[8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s16 x3, mllib_s16 y3, mllib_s32 c1, mllib_s32 c2, mllib_s32 c3)
```

```
mllib_GraphicsDrawTriangle_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1, mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s16 x3, mllib_s16 y3, mllib_s16 z3, mllib_s32 c1, mllib_s32 c2, mllib_s32 c3)
```

```
mllib_GraphicsDrawTriangle_X_[8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s16 x3, mllib_s16 y3, mllib_s32 c, mllib_s32 c2)
```

```
mllib_GraphicsDrawTriangle_Z_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1, mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s16 x3, mllib_s16 y3, mllib_s16 z3, mllib_s32 c)
```

```
mllib_GraphicsDrawTriangleFanSet_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawTriangleFanSet_A_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawTriangleFanSet_AG_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawTriangleFanSet_AGZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsDrawTriangleFanSet_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsDrawTriangleFanSet_G_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
```

---

## *Graphics: Polygon Functions (Continued)*

---

```
mllib_GraphicsDrawTriangleFanSet_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y,
    mllib_s16 *z, mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsDrawTriangleFanSet_X_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c,
    mllib_s32 c2)
mllib_GraphicsDrawTriangleFanSet_Z_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y,
    mllib_s16 *z, mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsDrawTriangleSet_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsDrawTriangleSet_A_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsDrawTriangleSet_AG_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsDrawTriangleSet_AGZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
    mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsDrawTriangleSet_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
    mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsDrawTriangleSet_G_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsDrawTriangleSet_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
    mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsDrawTriangleSet_X_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c,
    mllib_s32 c2)
```

---

*Graphics: Polygon Functions (Continued)*

---

```
mllib_GraphicsDrawTriangleSet_Z_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
    mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsDrawTriangleStripSet_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsDrawTriangleStripSet_A_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsDrawTriangleStripSet_AG_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints,
    mllib_s32 *c)
mllib_GraphicsDrawTriangleStripSet_AGZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y,
    mllib_s16 *z, mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsDrawTriangleStripSet_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y,
    mllib_s16 *z, mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsDrawTriangleStripSet_G_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsDrawTriangleStripSet_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y,
    mllib_s16 *z, mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsDrawTriangleStripSet_X_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c,
    mllib_s32 c2)
mllib_GraphicsDrawTriangleStripSet_Z_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y,
    mllib_s16 *z, mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsFillPolygon_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

---

## *Graphics: Polygon Functions (Continued)*

---

```
mllib_GraphicsFillPolygon_A_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsFillPolygon_AG_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsFillPolygon_AGZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
    mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsFillPolygon_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
    mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsFillPolygon_G_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsFillPolygon_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
    mllib_s32 npoints, mllib_s32 *c)
mllib_GraphicsFillPolygon_X_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c,
    mllib_s32 c2)
mllib_GraphicsFillPolygon_Z_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
    mllib_s32 npoints, mllib_s32 c)
mllib_GraphicsFillRectangle_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 w, mllib_s32 h, mllib_s32 c)
mllib_GraphicsFillRectangle_X_[8|32](mllib_image *buffer, mllib_s16 x, mllib_s16 y, mllib_s32 w, mllib_s32 h, mllib_s32 c,
    mllib_s32 c2)
mllib_GraphicsFillTriangle_[8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s16 x3,
    mllib_s16 y3, mllib_s32 c)
```

---

*Graphics: Polygon Functions (Continued)*

---

```
mllib_GraphicsFillTriangle_A [8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s16 x3, mllib_s16 y3, mllib_s32 c)
```

```
mllib_GraphicsFillTriangle_AG [8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s16 x3, mllib_s16 y3, mllib_s32 c1, mllib_s32 c2, mllib_s32 c3)
```

```
mllib_GraphicsFillTriangle_AGZ [8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1, mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s16 x3, mllib_s16 y3, mllib_s16 z3, mllib_s32 c1, mllib_s32 c2, mllib_s32 c3)
```

```
mllib_GraphicsFillTriangle_AZ [8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1, mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s16 x3, mllib_s16 y3, mllib_s16 z3, mllib_s32 c)
```

```
mllib_GraphicsFillTriangle_G [8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s16 x3, mllib_s16 y3, mllib_s32 c1, mllib_s32 c2, mllib_s32 c3)
```

```
mllib_GraphicsFillTriangle_GZ [8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1, mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s16 x3, mllib_s16 y3, mllib_s16 z3, mllib_s32 c1, mllib_s32 c2, mllib_s32 c3)
```

```
mllib_GraphicsFillTriangle_X [8|32](mllib_image *buffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 x2, mllib_s16 y2, mllib_s16 x3, mllib_s16 y3, mllib_s32 c, mllib_s32 c2)
```

```
mllib_GraphicsFillTriangle_Z [8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 x1, mllib_s16 y1, mllib_s16 z1, mllib_s16 x2, mllib_s16 y2, mllib_s16 z2, mllib_s16 x3, mllib_s16 y3, mllib_s16 z3, mllib_s32 c)
```

```
mllib_GraphicsFillTriangleFanSet [8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsFillTriangleFanSet_A [8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

---

## *Graphics: Polygon Functions (Continued)*

---

```
mllib_GraphicsFillTriangleFanSet_AG_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 *c)
mllib_GraphicsFillTriangleFanSet_AGZ_[8|32](mllib_image *buffer, mlib_image *zbuffer, mlib_s16 *x, mlib_s16 *y,
    mlib_s16 *z, mlib_s32 npoints, mlib_s32 *c)
mllib_GraphicsFillTriangleFanSet_AZ_[8|32](mllib_image *buffer, mlib_image *zbuffer, mlib_s16 *x, mlib_s16 *y,
    mlib_s16 *z, mlib_s32 npoints, mlib_s32 c)
mllib_GraphicsFillTriangleFanSet_G_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 *c)
mllib_GraphicsFillTriangleFanSet_GZ_[8|32](mllib_image *buffer, mlib_image *zbuffer, mlib_s16 *x, mlib_s16 *y,
    mlib_s16 *z, mlib_s32 npoints, mlib_s32 *c)
mllib_GraphicsFillTriangleFanSet_X_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c,
    mlib_s32 c2)
mllib_GraphicsFillTriangleFanSet_Z_[8|32](mllib_image *buffer, mlib_image *zbuffer, mlib_s16 *x, mlib_s16 *y,
    mlib_s16 *z, mlib_s32 npoints, mlib_s32 c)
mllib_GraphicsFillTriangleSet_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c)
mllib_GraphicsFillTriangleSet_A_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 c)
mllib_GraphicsFillTriangleSet_AG_[8|32](mllib_image *buffer, mlib_s16 *x, mlib_s16 *y, mlib_s32 npoints, mlib_s32 *c)
mllib_GraphicsFillTriangleSet_AGZ_[8|32](mllib_image *buffer, mlib_image *zbuffer, mlib_s16 *x, mlib_s16 *y, mlib_s16 *z,
    mlib_s32 npoints, mlib_s32 *c)
```

---

*Graphics: Polygon Functions (Continued)*

---

```
mllib_GraphicsFillTriangleSet_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsFillTriangleSet_G_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsFillTriangleSet_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsFillTriangleSet_X_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c,
mllib_s32 c2)
```

```
mllib_GraphicsFillTriangleSet_Z_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y, mllib_s16 *z,
mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsFillTriangleStripSet_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsFillTriangleStripSet_A_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsFillTriangleStripSet_AG_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints,
mllib_s32 *c)
```

```
mllib_GraphicsFillTriangleStripSet_AGZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y,
mllib_s16 *z, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsFillTriangleStripSet_AZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y,
mllib_s16 *z, mllib_s32 npoints, mllib_s32 c)
```

```
mllib_GraphicsFillTriangleStripSet_G_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 *c)
```

---

### *Graphics: Polygon Functions (Continued)*

---

```
mllib_GraphicsFillTriangleStripSet_GZ_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y,  
mllib_s16 *z, mllib_s32 npoints, mllib_s32 *c)
```

```
mllib_GraphicsFillTriangleStripSet_X_[8|32](mllib_image *buffer, mllib_s16 *x, mllib_s16 *y, mllib_s32 npoints, mllib_s32 c,  
mllib_s32 c2)
```

```
mllib_GraphicsFillTriangleStripSet_Z_[8|32](mllib_image *buffer, mllib_image *zbuffer, mllib_s16 *x, mllib_s16 *y,  
mllib_s16 *z, mllib_s32 npoints, mllib_s32 c)
```

---

## **VOLUME IMAGING FUNCTIONS AND SYNTAX**

---

### *Volume Imaging: Maximum Intensity Searching Functions*

---

```
mllib_VolumeFindMax_S16(mllib_s16 *max, mllib_rays *rays)
```

```
mllib_VolumeFindMax_U8(mllib_u8 *max, mllib_rays *rays)
```

```
mllib_VolumeFindMaxBMask_S16(mllib_s16 *max, mllib_rays *rays, mllib_u8 *bmask)
```

```
mllib_VolumeFindMaxBMask_U8(mllib_u8 *max, mllib_rays *rays, mllib_u8 *bmask)
```

```
mllib_VolumeFindMaxCMask_S16(mllib_s16 *max, mllib_rays *rays, mllib_u8 *cmask, mllib_s32 thresh)
```

```
mllib_VolumeFindMaxCMask_U8(mllib_u8 *max, mllib_rays *rays, mllib_u8 *cmask, mllib_s32 thresh)
```

---

---

### *Volume Imaging: Ray Casting Functions*

```
mlib_VolumeRayCast_General_[Parallel|Divergent]_[Nearest|Trilinear]_[U8_Bit|U8_U8|S16_S16](mlib_rays *rays,  
    mlib_genvolume *vol, void *buffer)  
mlib_VolumeRayCast_Blocked_[Parallel|Divergent]_[Nearest|Trilinear]_[U8_U8|S16_S16](mlib_rays *rays,  
    mlib_blkvolume *blk, void *buffer)
```

---

---

### *Volume Imaging: Window-level Operation Functions*

```
mlib_VolumeWindowLevel(mlib_u8 *dst, mlib_s16 *src, mlib_s32 window, mlib_s32 level, mlib_s32 gmax, mlib_s32 gmin,  
    mlib_s32 len)
```

---

## **READER COMMENTS**

Readers of the mediaLib Quick Reference are welcome to make comments by sending email to:

medialib@sun.com

We are interested in your ideas on organization and additional information that might be useful to include in future versions.







Sun Microsystems, Inc.  
901 San Antonio Road  
Palo Alto, CA 94303-4900 USA  
800/681-8845  
<http://www.sun.com/sparc>

Part Number: 802-7801-07

©2001 Sun Microsystems, Inc. All Rights reserved.

THE INFORMATION CONTAINED IN THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT ANY EXPRESS REPRESENTATIONS OF WARRANTIES. IN ADDITION, SUN MICROSYSTEMS, INC. DISCLAIMS ALL IMPLIED REPRESENTATIONS AND WARRANTIES, INCLUDING ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

This document contains proprietary information of Sun Microsystems, Inc. or under license from third parties. No part of this document may be reproduced in any form or by any means or transferred to any third party without the prior written consent of Sun Microsystems, Inc.

Sun, Sun Microsystems and the Sun Logo are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. All SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The information contained in this document is not designed or intended for use in on-line control of aircraft, aircraft navigation or aircraft communications; or in the design, construction, operation or maintenance of any nuclear facility. Sun disclaims any express or implied warranty of fitness for such uses.