NAME

libtiff - introduction to libtiff, a library for reading and writing TIFF files

SYNOPSIS

```
#include <tiffio.h>

cc file.c -ltiff
```

DESCRIPTION

libtiff is a library for reading and writing data files encoded with the *"Tag Image File"* format, Revision 6.0 (or revision 5.0 or revision 4.0). This file format is suitable for archiving multi-color and monochromatic image data.

The library supports several compression algorithms, as indicated by the **Compression** field, including: no compression (1), CCITT 1D Huffman compression (2), CCITT Group 3 Facsimile compression (3), CCITT Group 4 Facsimile compression (4), Lempel-Ziv & Welch compression (5), baseline JPEG compression (7), word-aligned 1D Huffman compression (32771), PackBits compression (32773). In addition, several nonstandard compression algorithms are supported: the 4-bit compression algorithm used by the *ThunderScan* program (32809) (decompression only), NeXT's 2-bit compression algorithm (32766) (decompression only), an experimental LZ-style algorithm known as Deflate (32946), and an experimental CIE LogLuv compression scheme designed for images with high dynamic range (32845 for LogL and 32845 for LogLuv). Directory information may be in either little- or big-endian byte order; byte swapping is automatically done by the library. Data bit ordering may be either Most Significant Bit (**MSB**) to Least Significant Bit (**LSB**) or LSB to MSB. Finally, the library does not support files in which the **BitsPerSample** , **Compression** , **MinSampleValue** , or **MaxSampleValue** fields are defined differently on a per-sample basis (in Rev. 6.0 the **Compression** tag is not defined on a per-sample basis, so this is immaterial).

DATA TYPES

The library makes extensive use of C typedefs to promote portability. Two sets of typedefs are used, one for communication with clients of the library and one for internal data structures and parsing of the TIFF format. The following typedefs are exposed to users either through function definitions or through parameters passed through the varargs interfaces.

```
typedef uint32_t ttag_t;    // directory tag
typedef uint32_t tdir_t;    // directory index
typedef uint16_t tsample_t; // sample number
typedef uint32_t tstrip_t;  // strip number
typedef uint32_t ttile_t;   // tile number
```

```
typedef int64_t tmsize_t;   // signed size type (int32_t on 32-bit platforms)
typedef tmsize_t tsize_t;   // i/o size in bytes
typedef void* tdata_t;      // image data ref
typedef void* thandle_t;    // client data handle
typedef uint64_t toff_t;    // file offset
```

Note that **tstrip_t**, **ttile_t**, and **tsize_t** are constrained to be no more than 32-bit quantities by 32-bit fields they are stored in in the TIFF image. Likewise **tsample_t** is limited by the 16-bit field used to store the **SamplesPerPixel** tag.

tdir_t constrains the maximum number of IFDs that may appear in an image and may be an arbitrary size (w/o penalty). Starting with libtiff 4.5.0, tdir_t is a 32-bit unsigned integer. Previously, it was a 16-bit unsigned integer.

ttag_t must be either int, unsigned int, pointer, or double because the library uses a varargs interface and C restricts the type of the parameter before an ellipsis to be a promoted type. **toff_t** is defined as **uint64_t** because TIFF file offsets are (unsigned) 32-bit quantities, and BigTIFF file offsets are unsigned 64-bit quantities. A signed value is used because some interfaces return -1 on error. Finally, note that user-specified data references are passed as opaque handles and only cast at the lowest layers where their type is presumed.

LIST OF ROUTINES

The following routines are part of the library. Consult specific manual pages for details on their operation; on most systems doing **man function-name** will work.

LIBTIFF FUNCTIONS

Name	Description
TIFFAccessTagMethods()	provides read/write access to the TIFFTagMethods within the TIFF structure to application code without giving access to the private TIFF structure
TIFFCheckpointDirectory()	writes the current state of the directory
TIFFCheckTile()	very x,y,z,sample is within

	image
TIFFCIELabToRGBInit()	initialize CIE L*a*b* 1976 to RGB conversion state
TIFFCIELabToXYZ()	perform CIE L*a*b* 1976 to CIE XYZ conversion
TIFFCleanup()	auxiliary function to free the TIFF structure
TIFFClientdata()	return open file's clientdata handle, which represents the file descriptor used within **libtiff**.
TIFFClientOpen()	open a file for reading or writing
TIFFClientOpenExt()	open a file for reading or writing with options, such as re-entrant error and warning handlers may be passed
TIFFClose()	close a previously opened TIFF file
TIFFComputeStrip()	return strip containing y,sample
TIFFComputeTile()	return tile containing x,y,z,sample
TIFFCreateCustomDirectory()	setup for a *custom* directory in a open TIFF file
TIFFCreateDirectory()	setup for a directory in a open TIFF file
TIFFCreateEXIFDirectory()	setup for a *EXIF* custom directory in a open TIFF file within a TIFF

	tag
TIFFCreateGPSDirectory()	setup for a *GPS* custom directory in a open TIFF file within a TIFF tag
TIFFCurrentDirectory()	return index of current directory
TIFFCurrentDirOffset()	return file offset of the current directory (instead of an index)
TIFFCurrentRow()	return index of current scanline
TIFFCurrentStrip()	return index of current strip
TIFFCurrentTile()	return index of current tile
TIFFDataWidth()	return the size of TIFF data types
TIFFDefaultStripSize()	return number of rows for a reasonable-sized strip according to the current settings of the ImageWidth, BitsPerSample and SamplesPerPixel, tags and any compression-specific requirements
TIFFDefaultTileSize()	return pixel width and height of a reasonable-sized tile; suitable for setting up the TileWidth and TileLength tags
TIFFDeferStrileArrayWriting()	is an advanced writing function to control when/where the [Strip/Tile][Offsets/ByteCounts] arrays are written into the file, and

	must be used in a particular sequence together with TIFFForceStrileArrayWriting() (see description)
TIFFError()	library-wide error handling function printing to **stderr**
TIFFErrorExt()	user-specific library-wide error handling function that can be passed a file handle, which is set to the open TIFF file within **libtiff**
TIFFErrorExtR()	user-specific re-entrant library error handling function, to which its TIFF structure is passed containing the pointer to a user-specific data object
TIFFFdOpen()	open a file for reading or writing
TIFFFdOpenExt()	open a file for reading or writing with options, such as re-entrant error and warning handlers may be passed
TIFFFieldDataType()	get data type from field information
TIFFFieldIsAnonymous()	returns if field was unknown to **libtiff** and has been auto-registered
TIFFFieldName()	get field name from field information
TIFFFieldPassCount()	get whether to pass a value count to Get/SetField
TIFFFieldReadCount()	get number of values to be read

	from field
TIFFFieldSetGetCountSize()	returns size of **count** parameter of *TIFFSetField()* and *TIFFGetField()*
TIFFFieldSetGetSize()	return data size in bytes of the field data type used for **libtiff** internal storage.
TIFFFieldTag()	get tag value from field information
TIFFFieldWithName()	get field information given field name
TIFFFieldWithTag()	get field information given tag
TIFFFieldWriteCount()	get number of values to be written to field
TIFFFileName()	return name of open file
TIFFFileno()	return open file descriptor
TIFFFindCODEC()	find standard codec for the specific scheme
TIFFFindField()	get field information given tag and data type
TIFFFlush()	flush all pending writes
TIFFFlushData()	flush pending data writes

TIFFForceStrileArrayWriting()	is an advanced writing function that writes the [Strip/Tile][Offsets/ByteCounts] arrays at the end of the file (see description)
TIFFFreeDirectory()	release storage associated with a directory
TIFFGetBitRevTable()	return bit reversal table
TIFFGetClientInfo()	returns a pointer to the data of the named entry in the clientinfo-list
TIFFGetCloseProc()	returns a pointer to file close method
TIFFGetConfiguredCODECs()	gets list of configured codecs, both built-in and registered by user
TIFFGetField()	return tag value in current directory
TIFFGetFieldDefaulted()	return tag value in current directory with default value set if the value is not already set and a default is defined
TIFFGetMapFileProc()	returns a pointer to memory mapping method
TIFFGetMode()	return open file mode
TIFFGetReadProc()	returns a pointer to file read method
TIFFGetSeekProc()	returns a pointer to file seek

	method
TIFFGetSizeProc()	returns a pointer to file size requesting method
TIFFGetStrileByteCount()	return value of the TileByteCounts/StripByteCounts array for the specified tile/strile
TIFFGetStrileByteCountWithErr()	same as *TIFFGetStrileByteCount()* and additionally provides an error return
TIFFGetStrileOffset()	return value of the TileOffsets/StripOffsets array for the specified tile/strile
TIFFGetStrileOffsetWithErr()	same as *TIFFGetStrileOffset()* and additionally provides an error return
TIFFGetTagListCount()	return number of entries in the custom tag list
TIFFGetTagListEntry()	return tag number of the (n.th - 1) entry within the custom tag list
TIFFGetUnmapFileProc()	returns a pointer to memory unmapping method
TIFFGetVersion()	return library version string
TIFFGetWriteProc()	returns a pointer to file write method
TIFFIsBigEndian()	returns a non-zero value if the file is BigEndian and zero if the file is LittleEndian

TIFFIsBigTIFF()	returns a non-zero value if the file is in BigTIFF style
TIFFIsByteSwapped()	return true if image data is byte-swapped
TIFFIsCODECConfigured()	check, whether we have working codec
TIFFIsMSB2LSB()	return true if image data is being returned with bit 0 as the most significant bit
TIFFIsTiled()	return true if image data is tiled
TIFFIsUpSampled()	returns a non-zero value if image data returned through the read interface Routines is being up-sampled
TIFFLastDirectory()	returns a non-zero value if the current directory is the last directory in the file; otherwise zero is returned
TIFFMergeFieldInfo()	adds application defined TIFF tags to the list of known **libtiff** tags
TIFFNumberOfDirectories()	return number of directories in a file
TIFFNumberOfStrips()	return number of strips in an image
TIFFNumberOfTiles()	return number of tiles in an image
TIFFOpen()	open a file for reading or

	writing
TIFFOpenExt()	open a file for reading or writing with options, such as re-entrant error and warning handlers may be passed
TIFFOpenW()	opens a TIFF file with a Unicode filename, for read/writing
TIFFOpenWExt()	opens a TIFF file with a Unicode filename, for read/writing with options, such as re-entrant error and warning handlers may be passed
TIFFOpenOptionsAlloc()	allocates memory for *TIFFOpenOptions* opaque structure
TIFFOpenOptionsFree()	releases the allocated memory for *TIFFOpenOptions*
TIFFOpenOptionsSetMaxSingleMemAlloc()	limits the maximum single memory allocation within **libtiff**
TIFFOpenOptionsSetErrorHandlerExtR()	setup of a user-specific and per-TIFF handle (re-entrant) error handler
TIFFOpenOptionsSetWarningHandlerExtR()	setup of a user-specific and per-TIFF handle (re-entrant) warning handler
TIFFPrintDirectory()	print description of the current directory
TIFFRasterScanlineSize()	returns the size in bytes of a complete decoded and packed raster scanline

TIFFRasterScanlineSize64()	return size as **uint64_t**
TIFFRawStripSize()	return number of bytes in a raw strip
TIFFRawStripSize64()	return number of bytes in a raw strip as **uint64_t**
TIFFReadBufferSetup()	specify i/o buffer for reading
TIFFReadCustomDirectory()	read the custom directory from the given offset and set the context of the TIFF-handle tif to that custom directory
TIFFReadDirectory()	read the next directory
TIFFReadEncodedStrip()	read and decode a strip of data
TIFFReadEncodedTile()	read and decode a tile of data
TIFFReadEXIFDirectory()	read the EXIF directory from the given offset and set the context of the TIFF-handle tif to that EXIF directory
TIFFReadFromUserBuffer()	replaces the use of *TIFFReadEncodedStrip()* / *TIFFReadEncodedTile()* when the user can provide the buffer for the input data
TIFFReadGPSDirectory()	read the GPS directory from the given offset and set the context of

	the TIFF-handle tif to that GPS directory
TIFFReadRawStrip()	read a raw strip of data
TIFFReadRawTile()	read a raw tile of data
TIFFReadRGBAImage()	read an image into a fixed format raster
TIFFReadRGBAImageOriented()	works like *TIFFReadRGBAImage()* except that the user can specify the raster origin position
TIFFReadRGBAStrip()	reads a single strip of a strip-based image into memory, storing the result in the user supplied RGBA raster
TIFFReadRGBAStripExt()	same as *TIFFReadRGBAStrip()* but providing the parameter *stop_on_error*
TIFFReadRGBATile()	reads a single tile of a tile-based image into memory, storing the result in the user supplied RGBA raster
TIFFReadRGBATileExt()	same as *TIFFReadRGBATile()* but providing the parameter *stop_on_error*
TIFFReadScanline()	read and decode a row of data
TIFFReadTile()	read and decode a tile of data

TIFFRegisterCODEC()	override standard codec for the specific scheme
TIFFReverseBits()	reverse bits in an array of bytes
TIFFRewriteDirectory()	operates similarly to *TIFFWriteDirectory()*, but can be called with directories previously read or written that already have an established location in the file and places it at the end of the file
TIFFRGBAImageBegin()	setup decoder state for TIFFRGBAImageGet
TIFFRGBAImageEnd()	release TIFFRGBAImage decoder state
TIFFRGBAImageGet()	read and decode an image
TIFFRGBAImageOK()	is image readable by TIFFRGBAImageGet
TIFFScanlineSize()	return size of a scanline
TIFFScanlineSize64()	return size of a scanline as **uint64_t**
TIFFSetClientdata()	set open file’s clientdata (file descriptor/handle), and return previous value
TIFFSetClientInfo()	adds or replaces an entry in the clientinfo-list
TIFFSetCompressionScheme()	set compression

	scheme
TIFFSetDirectory()	set the current directory
TIFFSetErrorHandler()	set error handler function
TIFFSetErrorHandlerExt()	set error handler function with a file handle as parameter
TIFFSetField()	set a tag's value in the current directory
TIFFSetFileName()	sets the file name in the TIFF-structure and returns the old file name
TIFFSetFileno()	overwrites a copy of the open file's I/O descriptor, and return previous value (refer to detailed description)
TIFFSetMode()	sets the **libtiff** open mode in the TIFF-structure and returns the old mode
TIFFSetSubDirectory()	set the current directory
TIFFSetTagExtender()	is used to register the merge function for user defined tags as an extender callback with **libtiff**
TIFFSetupStrips()	setup or reset strip parameters and strip array memory
TIFFSetWarningHandler()	set warning handler function

TIFFSetWarningHandlerExt()	set warning handler function with a file handle as parameter
TIFFSetWriteOffset()	set current write offset
TIFFStripSize()	return size of a strip
TIFFStripSize64()	return equivalent size for a strip of data as **uint64_t**
TIFFSwabArrayOfDouble()	swap bytes of an array of doubles
TIFFSwabArrayOfFloat()	swap bytes of an array of floats
TIFFSwabArrayOfLong()	swap bytes of an array of longs
TIFFSwabArrayOfLong8()	swap bytes of an array of uint64_t
TIFFSwabArrayOfShort()	swap bytes of an array of shorts
TIFFSwabArrayOfTriples()	swap the first and third byte of each triple within an array of bytes
TIFFSwabDouble()	swap bytes of double
TIFFSwabFloat()	swap bytes of float
TIFFSwabLong()	swap bytes of long
TIFFSwabLong8()	swap bytes of long long

	|(uint64_t)
TIFFSwabShort()	swap bytes of short
TIFFTileRowSize()	return size of a row in a tile
TIFFTileRowSize64()	return size of a row in a tile as **uint64_t**
TIFFTileSize()	return size of a tile
TIFFTileSize64()	return size of a tile as **uint64_t**
TIFFUnlinkDirectory()	unlink the specified directory from the directory chain
TIFFUnRegisterCODEC()	unregisters the codec
TIFFUnsetField()	clear the contents of the field in the internal structure
TIFFVGetField()	return tag value in current directory
TIFFVGetFieldDefaulted()	return tag value in current directory
TIFFVSetField()	set a tag’s value in the current directory
TIFFVStripSize()	return number of bytes in a strip
TIFFVStripSize64()	return number of bytes in a strip with *nrows* rows of data as

	|**uint64_t**
TIFFVTileSize()	returns the number of bytes in a row-aligned tile with *nrows* of data
TIFFVTileSize64()	returns the number of bytes in a row-aligned tile with *nrows* of data a **uint64_t** number
TIFFWarning()	library-wide warning function printing to **stderr**
TIFFWarningExt()	user-specific library-wide warning handling function that can be passed a file handle, which is set to the open TIFF file within **libtiff**
TIFFWarningExtR()	user-specific re-entrant library warning handling function, to which its TIFF structure is passed containing the pointer to a user-specific data object
TIFFWriteBufferSetup()	sets up the data buffer used to write raw (encoded) data to a file
TIFFWriteCheck()	verify file is writable and that the directory information is setup properly
TIFFWriteCustomDirectory()	write the current custom directory (also EXIF or GPS) to file
TIFFWriteDirectory()	write the current directory
TIFFWriteEncodedStrip()	compress and write a strip of data

TIFFWriteEncodedTile()	compress and write a tile of data
TIFFWriteRawStrip()	write a raw strip of data
TIFFWriteRawTile()	write a raw tile of data
TIFFWriteScanline()	write a scanline of data
TIFFWriteTile()	compress and write a tile of data
TIFFXYZToRGB()	perform CIE XYZ to RGB conversion
TIFFYCbCrtoRGB()	perform YCbCr to RGB conversion
TIFFYCbCrToRGBInit()	initialize YCbCr to RGB conversion state

LIBTIFF AUXILLARY FUNCTIONS

Name	Description
_TIFFCheckMalloc()	checking for integer overflow before dynamically allocate memory buffer
_TIFFCheckRealloc()	checking for integer overflow before dynamically reallocate memory buffer
_TIFFClampDoubleToUInt32()	clamps double values into the range of **uint32_t** (i.e. 0 ..

	0xFFFFFFFF)
_TIFFfree()	free memory buffer
_TIFFGetExifFields()	return a pointer to the **libtiff** internal definition list of the EXIF tags
_TIFFGetGpsFields()	return a pointer to the **libtiff** internal definition list of the GPS tags
_TIFFmalloc()	dynamically allocate memory buffer
_TIFFmemcmp()	compare contents of the memory buffers
_TIFFmemcpy()	copy contents of the one buffer to another
_TIFFmemset()	fill memory buffer with a constant byte
_TIFFMultiply32()	checks for an integer overflow of the multiplication result of *uint32_t* and return the multiplication result or *0* if an overflow would happen
_TIFFMultiply64()	checks for an integer overflow of the multiplication result of *uint64_t* and return the multiplication result or *0* if an overflow would happen
_TIFFrealloc()	dynamically reallocate memory buffer

_TIFFRewriteField()	Rewrite a field in the directory on disk without regard to updating the TIFF directory structure in memory

TAG USAGE

For a table of TIFF tags recognized by the library refer to *LibTIFF Coverage of the TIFF 6.0 Specification*.

"PSEUDO TAGS"

In addition to the normal TIFF tags the library supports a collection of tags whose values lie in a range outside the valid range of TIFF tags. These tags are termed *pseudo-tags* and are used to control various codec-specific functions within the library. The table below summarizes the defined pseudo-tags.

LIBTIFF SUPPORTED TAGS

Tag name	Codec	R/W	Library Use/Notes
TIFFTAG_FAXMODE	G3	R/W	general codec operation
TIFFTAG_FAXFILLFUNC	G3/G4	R/W	bitmap fill function
TIFFTAG_JPEGQUALITY	JPEG	R/W	compression quality control
TIFFTAG_JPEGCOLORMODE	JPEG	R/W	control colorspace conversions
TIFFTAG_JPEGTABLESMODE	JPEG	R/W	control contents of **JPEGTables** tag
TIFFTAG_ZIPQUALITY	Deflate	R/W	compression quality level

TIFFTAG_PIXARLOGDATAFMT	PixarLog	R/W	user data format
TIFFTAG_PIXARLOGQUALITY	PixarLog	R/W	compression quality level
TIFFTAG_SGILOGDATAFMT	SGILog	R/W	user data format

TIFFTAG_FAXMODE:
Control the operation of the Group 3 codec. Possible values (independent bits that can be combined by or'ing them together) are:

FAXMODE_CLASSIC:
(enable old-style format in which the **RTC** is written at the end of the last strip),

FAXMODE_NORTC:
(opposite of **FAXMODE_CLASSIC**; also called **FAXMODE_CLASSF**),

FAXMODE_NOEOL:
(do not write **EOL** codes at the start of each row of data),

FAXMODE_BYTEALIGN:
(align each encoded row to an 8-bit boundary),

FAXMODE_WORDALIGN:
(align each encoded row to an 16-bit boundary),

The default value is dependent on the compression scheme; this pseudo-tag is used by the various G3 and G4 codecs to share code.

TIFFTAG_FAXFILLFUNC:
Control the function used to convert arrays of black and white runs to packed bit arrays. This hook can be used to image decoded scanlines in multi-bit depth rasters (e.g. for display in colormap mode) or for other purposes. The default value is a pointer to a builtin function that images packed bilevel data.

TIFFTAG_IPTCNEWSPHOTO:

Tag contains image metadata per the IPTC newsphoto spec: Headline, captioning, credit, etc<?> Used by most wire services.

TIFFTAG_PHOTOSHOP:
Tag contains Photoshop captioning information and metadata. Photoshop uses in parallel and redundantly alongside **IPTCNEWSPHOTO** information.

TIFFTAG_JPEGQUALITY:
Control the compression quality level used in the baseline algorithm. Note that quality levels are in the range 0-100 with a default value of 75.

TIFFTAG_JPEGCOLORMODE:
Control whether or not conversion is done between RGB and YCbCr colorspaces. Possible values are:

JPEGCOLORMODE_RAW:
(do not convert), and

JPEGCOLORMODE_RGB:
(convert to/from RGB)

The default value is **JPEGCOLORMODE_RAW**.

TIFFTAG_JPEGTABLESMODE:
Control the information written in the **JPEGTables** tag. Possible values (independent bits that can be combined by or'ing them together) are:

JPEGTABLESMODE_QUANT:
(include quantization tables), and

JPEGTABLESMODE_HUFF:
(include Huffman encoding tables).

The default value is JPEGTABLESMODE_QUANT | JPEGTABLESMODE_HUFF.

TIFFTAG_ZIPQUALITY:
Control the compression technique used by the Deflate codec. Quality levels are in the range 1-9 with larger numbers yielding better compression at the cost of more computation. The default quality level is 6 which yields a good time-space tradeoff.

TIFFTAG_PIXARLOGDATAFMT:
Control the format of user data passed *in* to the PixarLog codec when encoding and passed *out* from when decoding. Possible values are:

PIXARLOGDATAFMT_8BIT:
for 8-bit unsigned pixels,

PIXARLOGDATAFMT_8BITABGR:
for 8-bit unsigned ABGR-ordered pixels,

PIXARLOGDATAFMT_11BITLOG:
for 11-bit log-encoded raw data,

PIXARLOGDATAFMT_12BITPICIO:
for 12-bit PICIO-compatible data,

PIXARLOGDATAFMT_16BIT:
for 16-bit signed samples, and

PIXARLOGDATAFMT_FLOAT:
for 32-bit IEEE floating point samples.

TIFFTAG_PIXARLOGQUALITY:
Control the compression technique used by the PixarLog codec. This value is treated identically to **TIFFTAG_ZIPQUALITY**; see the above description.

TIFFTAG_SGILOGDATAFMT:
Control the format of client data passed *in* to the SGILog codec when encoding and passed *out* from when decoding. Possible values are:

SGILOGDATAFMT_FLTXYZ:
for converting between LogLuv and 32-bit IEEE floating valued XYZ pixels,

SGILOGDATAFMT_16BITLUV:
for 16-bit encoded Luv pixels,

SGILOGDATAFMT_32BITRAW: **SGILOGDATAFMT_24BITRAW**:
for no conversion of data,

SGILOGDATAFMT_8BITRGB:

for returning 8-bit RGB data (valid only when decoding LogLuv-encoded data),

SGILOGDATAFMT_FLTY:
for converting between LogL and 32-bit IEEE floating valued Y pixels,

SGILOGDATAFMT_16BITL:
for 16-bit encoded L pixels, and

SGILOGDATAFMT_8BITGRY:
for returning 8-bit greyscale data (valid only when decoding LogL-encoded data).

DIAGNOSTICS

All error messages are directed through the *TIFFErrorExtR()* routine. By default messages are directed to **stderr** in the form: **module: message\n**. Warning messages are likewise directed through the *TIFFWarningExtR()* routine.

SEE ALSO

tiffdump, tiffcp, tiffinfo, tiffsplit,

"Tag Image File Format Specification *Revision 6.0*", an Aldus Technical Memorandum.

"The Spirit of TIFF Class F" , an appendix to the TIFF 5.0 specification prepared by Cygnet Technologies.

BUGS

- The library does not support multi-sample images where some samples have different bits/sample.
- The library does not support random access to compressed data that is organized with more than one row per tile or strip.

AUTHOR

LibTIFF contributors

COPYRIGHT