FLI Software Development Library

— Version 1.40 —

Windows and Linux support for FLI CCD cameras, filter wheels, and focusers.

Finger Lakes Instrumentation

Contents

1 Introduction

This library provides a core set of functions for programming FLI CCD cameras, filter wheels, and focusers under Windows and Linux. The type definitions, function prototypes, and definitions/enumerations of constant values used by library functions are spcified in `libfli.h`. All library functions return zero on sucessful completion, and non-zero if an error occurred. The exact nature of an error can be found by treating the negative of a function's return value as a system error code, for example:

```
if ((err = FLIOpen(&dev, name, domain)))
{
  fprintf(stderr, "Error FLIOpen: %s\n", strerror((int)-err));
  exit(1);
}
```

2

Library Defined Types

Names

2.1

typedef long **flidev_t**

An opaque handle used by library functions to refer to FLI hardware

An opaque handle used by library functions to refer to FLI hardware

2.2

typedef long **flidomain_t**

The domain of an FLI device.

The domain of an FLI device. This consists of a bitwise ORed combination of interface method and device type. Valid interfaces are `FLIDOMAIN_PARALLEL_PORT`, `FLIDOMAIN_USB`, `FLIDOMAIN_SERIAL`, and `FLIDOMAIN_INET`. Valid device types are `FLIDEVICE_CAMERA`, `FLIDOMAIN_FILTERWHEEL`, and `FLIDOMAIN_FOCUSER`.

See Also: FLIOpen
FLIList

2.3

typedef long **fliframe_t**

The frame type for an FLI CCD camera device.

The frame type for an FLI CCD camera device. Valid frame types are `FLI_FRAME_TYPE_NORMAL` and `FLI_FRAME_TYPE_DARK`.

See Also: FLISetFrameType

2.4

typedef long **flibitdepth_t**

The gray-scale bit depth for an FLI camera device.

The gray-scale bit depth for an FLI camera device. Valid bit depths are `FLI_MODE_8BIT` and `FLI_MODE_16BIT`.

See Also: FLISetBitDepth

2.5

typedef long **flishutter_t**

Type used for shutter operations for an FLI camera device.

Type used for shutter operations for an FLI camera device. Valid shutter types are `FLI_SHUTTER_CLOSE`, `FLI_SHUTTER_OPEN`, `FLI_SHUTTER_EXTERNAL_TRIGGER`, `FLI_SHUTTER_EXTERNAL_TRIGGER_LOW`, and `FLI_SHUTTER_EXTERNAL_TRIGGER_HIGH`.

See Also: FLIControlShutter

2.6

typedef long **flibgflush_t**

Type used for background flush operations for an FLI camera device.

Type used for background flush operations for an FLI camera device. Valid bgflush types are `FLI_BGFLUSH_STOP` and `FLI_BGFLUSH_START`.

See Also: FLIControlBackgroundFlush

2.7

typedef long **flichannel_t**

Type used to determine which temperature channel to read.

Type used to determine which temperature channel to read. Valid channel types are `FLI_TEMPERATURE_INTERNAL` and `FLI_TEMPERATURE_EXTERNAL`.

See Also: FLIReadTemperature

2.8

typedef long **flidebug_t**

Type specifying library debug levels.

Type specifying library debug levels. Valid debug levels are `FLIDEBUG_NONE`, `FLIDEBUG_INFO`, `FLIDEBUG_WARN`, and `FLIDEBUG_FAIL`.

See Also: FLISetDebugLevel

3

Library Functions

Names

3.1

LIBFLIAPI **FLICancelExposure** (flidev_t dev)

Cancel an exposure for a given camera.

Cancel an exposure for a given camera. This function cancels an exposure in progress by closing the shutter.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Camera to cancel the exposure of.

See Also: FLIExposeFrame
FLIGetExposureStatus
FLISetExposureTime

3.2

LIBFLIAPI **FLIClose** (flidev_t dev)

Close a handle to a FLI device

Close a handle to a FLI device

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	The device handle to be closed.
See Also:	FLIOpen	

3.3

LIBFLIAPI **FLIGetArrayArea** (flidev_t dev, long* ul_x, long* ul_y, long* lr_x, long* lr_y)

Get the array area of the given camera.

Get the array area of the given camera. This function finds the *total* area of the CCD array for camera `dev`. This area is specified in terms of a upper-left point and a lower-right point. The upper-left x-coordinate is placed in `ul_x`, the upper-left y-coordinate is placed in `ul_y`, the lower-right x-coordinate is placed in `lr_x`, and the lower-right y-coordinate is placed in `lr_y`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.

Parameters:	`dev`	Camera to get the array area of.
	`ul_x`	Pointer to where the upper-left x-coordinate is to beplaced.
	`ul_y`	Pointer to where the upper-left y-coordinate is to beplaced.
	`lr_x`	Pointer to where the lower-right x-coordinate is to beplaced.
	`lr_y`	Pointer to where the lower-right y-coordinate is to beplaced.
See Also:	FLIGetVisibleArea FLISetImageArea	

3.4

LIBFLIAPI **FLIFlushRow** (flidev_t dev, long rows, long repeat)

Flush rows of a given camera.

Flush rows of a given camera. This function flushes `rows` rows of camera `dev` `repeat` times.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Camera to flush rows of.
	`rows`	Number of rows to flush.
	`repeat`	Number of times to flush each row.
See Also:	FLISetNFlushes	

3.5

LIBFLIAPI **FLIGetFWRevision** (flidev_t dev, long* fwrev)

Get firmware revision of a given device

Get firmware revision of a given device

Return Value:

`Zero`	on success.
`Non-zero`	on failure.

Parameters:

`dev`	Device to find the firmware revision of.
`fwrev`	Pointer to a long which will receive the firmwarerevision.

See Also:
FLIGetModel
FLIGetHWRevision
FLIGetSerialNum

3.6

LIBFLIAPI **FLIGetHWRevision** (flidev_t dev, long* hwrev)

Get the hardware revision of a given device

Get the hardware revision of a given device

Return Value:

`Zero`	on success.
`Non-zero`	on failure.

Parameters:

`dev`	Device to find the hardware revision of.
`hwrev`	Pointer to a long which will receive the hardwarerevision.

See Also:
FLIGetModel
FLIGetFWRevision
FLIGetSerialNum

3.7

LIBFLIAPI **FLIGetLibVersion** (char* ver, size_t len)

Get the current library version.

Get the current library version. This function copies up to `len - 1` characters of the current library version string followed by a terminating `NULL` character into the buffer pointed to by `ver`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`ver`	Pointer to a character buffer where the library versionstring is to be placed.
	`len`	The size in bytes of the buffer pointed to by`ver`.

3.8

LIBFLIAPI **FLIGetModel** (flidev_t dev, char* model, size_t len)

Get the model of a given device.

Get the model of a given device. This function copies up to `len - 1` characters of the model string for device `dev`, followed by a terminating `NULL` character into the buffer pointed to by `model`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Device to find model of.
	`model`	Pointer to a character buffer where the model stringis to be placed.
	`len`	The size in bytes of buffer pointed to by`model`.

See Also: FLIGetHWRevision
FLIGetFWRevision
FLIGetSerialNum

3.9

LIBFLIAPI **FLIGetPixelSize** (flidev_t dev, double* pixel_x, double* pixel_y)

Find the dimensions of a pixel in the array of the given device

Find the dimensions of a pixel in the array of the given device

Return Value:

`Zero`	on success.
`Non-zero`	on failure.

Parameters:

`dev`	Device to find the pixel size of.
`pixel_x`	Pointer to a double which will receive the size (inmicrons) of a pixel in the x direction.
`pixel_y`	Pointer to a double which will receive the size (inmicrons) of a pixel in the y direction.

See Also: FLIGetArrayArea
FLIGetVisibleArea

3.10

LIBFLIAPI **FLIGetVisibleArea** (flidev_t dev, long* ul_x, long* ul_y, long* lr_x, long* lr_y)

Get the visible area of the given camera.

Get the visible area of the given camera. This function finds the *visible* area of the CCD array for the camera `dev`. This area is specified in terms of a upper-left point and a lower-right point. The upper-left x-coordinate is placed in `ul_x`, the upper-left y-coordinate is placed in `ul_y`, the lower-right x-coordinate is placed in `lr_x`, the lower-right y-coordinate is placed in `lr_y`.

Return Value:

`Zero`	on success.
`Non-zero`	on failure.

Parameters:

`dev`	Camera to get the visible area of.
`ul_x`	Pointer to where the upper-left x-coordinate is to beplaced.
`ul_y`	Pointer to where the upper-left y-coordinate is to beplaced.
`lr_x`	Pointer to where the lower-right x-coordinate is to beplaced.
`lr_y`	Pointer to where the lower-right y-coordinate is to beplaced.

See Also: FLIGetArrayArea
FLISetImageArea

3.11

LIBFLIAPI **FLIOpen** (flidev_t* dev, char* name, flidomain_t domain)

Get a handle to an FLI device.

Get a handle to an FLI device. This function requires the filename and domain of the requested device. Valid device filenames can be obtained using the `FLIList()` function. An application may use any number of handles associated with the same physical device. When doing so, it is important to lock the appropriate device to ensure that multiple accesses to the same device do not occur during critical operations.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Pointer to where a device handle will be placed.
	`name`	Pointer to a string where the device filename to beopened is stored. For parallel port devices that are not probed by`FLIList()` (Windows 95/98/Me), place the address of theparallel port in a string in ascii form ie: ”0x378”.
	`domain`	Domain to apply to `name` for device opening.This is a bitwise ORed combination of interface method and devicetype. Valid interfaces include `FLIDOMAIN_PARALLEL_PORT`,`FLIDOMAIN_USB`, `FLIDOMAIN_SERIAL`, and`FLIDOMAIN_INET`. Valid device types include`FLIDEVICE_CAMERA`, `FLIDOMAIN_FILTERWHEEL`, and`FLIDOMAIN_FOCUSER`.
See Also:	FLIList FLIClose flidomain_t	

3.12

LIBFLIAPI **FLISetDebugLevel** (char* host, flidebug_t level)

Enable debugging of API operations and communications.

Enable debugging of API operations and communications. Use this function in combination with FLIDebug to assist in diagnosing problems that may be encountered during programming.

When usings Microsoft Windows operating systems, creating an empty file C: FLIDBG.TXT will override this option. All debug output will then be directed to this file.

Return Value: `Zero` on success.
`Non-zero` on failure.

Parameters: `host` Name of the file to send debugging information to.This parameter is ignored under Linux where `syslog(3)` isused to send debug messages (see `syslog.conf(5)` for how toconfigure syslogd).
`level` Debug level. A value of `FLIDEBUG_NONE` disablesdebugging. Values of `FLIDEBUG_FAIL`, `FLIDEBUG_WARN`, and`FLIDEBUG_INFO` enable progressively more verbose debug messages.

3.13

LIBFLIAPI **FLISetExposureTime** (flidev_t dev, long exptime)

Set the exposure time for a camera.

Set the exposure time for a camera. This function sets the exposure time for the camera `dev` to `exptime` msec.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Camera to set the exposure time of.
	`exptime`	Exposure time in msec.
See Also:	FLIExposeFrame FLICancelExposure FLIGetExposureStatus	

3.14

LIBFLIAPI **FLISetHBin** (flidev_t dev, long hbin)

Set the horizontal bin factor for a given camera.

Set the horizontal bin factor for a given camera. This function sets the horizontal bin factor for the camera `dev` to `hbin`. The valid range of the `hbin` parameter is from 1 to 16.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Camera to set horizontal bin factor of.
	`hbin`	Horizontal bin factor.
See Also:	FLISetVBin FLISetImageArea	

3.15

LIBFLIAPI **FLISetFrameType** (flidev_t dev, fliframe_t frametype)

Set the frame type for a given camera.

Set the frame type for a given camera. This function sets the frame type for camera `dev` to `frametype`. The `frametype` parameter is either `FLI_FRAME_TYPE_NORMAL` for a normal frame where the shutter opens or `FLI_FRAME_TYPE_DARK` for a dark frame where the shutter remains closed.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`cam`	Camera to set the frame type of.
	`frametype`	Frame type: `FLI_FRAME_TYPE_NORMAL` or `FLI_FRAME_TYPE_DARK`.
See Also:	fliframe_t FLIExposeFrame	

3.16

LIBFLIAPI **FLISetImageArea** (flidev_t dev, long ul_x, long ul_y, long lr_x, long lr_y)

Set the image area for a given camera.

Set the image area for a given camera. This function sets the image area for camera `dev` to an area specified in terms of a upper-left point and a lower-right point. The upper-left x-coordinate is `ul_x`, the upper-left y-coordinate is `ul_y`, the lower-right x-coordinate is `lr_x`, and the lower-right y-coordinate is `lr_y`. Note that the given lower-right coordinate must take into account the horizontal and vertical bin factor settings, but the upper-left coordinate is absolute. In other words, the lower-right coordinate used to set the image area is a virtual point (lr'_x, lr'_y) determined by:

$$lr'_x = ul_x + (lr_x - ul_x)/hbin$$

$$lr'_y = ul_y + (lr_y - ul_y)/vbin$$

Where (lr'_x, lr'_y) is the coordinate to pass to the `FLISetImageArea` function, (ul_x, ul_y) and (lr_x, lr_y) are the absolute coordinates of the desired image area, *hbin* is the horizontal bin factor, and *vbin* is the vertical bin factor.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.

Parameters:	`dev`	Camera to set image area of.
	`ul_x`	Upper-left x-coordinate of image area.
	`ul_y`	Upper-left y-coordinate of image area.
	`lr_x`	Lower-right x-coordinate of image area (lr'_x fromabove).
	`lr_y`	Lower-right y-coordinate of image area (lr'_y fromabove).
See Also:	FLIGetVisibleArea FLIGetArrayArea	

3.17

LIBFLIAPI **FLISetVBin** (flidev_t dev, long vbin)

Set the vertical bin factor for a given camera.

Set the vertical bin factor for a given camera. This function sets the vertical bin factor for the camera `dev` to `vbin`. The valid range of the `vbin` parameter is from 1 to 16.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Camera to set vertical bin factor of.
	`vbin`	Vertical bin factor.
See Also:	FLISetHBin FLISetImageArea	

3.18

LIBFLIAPI **FLIGetExposureStatus** (flidev_t dev, long* timeleft)

Find the remaining exposure time of a given camera.

Find the remaining exposure time of a given camera. This functions places the remaining exposure time (in milliseconds) in the location pointed to by `timeleft`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Camera to find the remaining exposure time of.
	`timeleft`	Pointer to where the remaining exposure time (in milliseonds) will be placed.
See Also:	FLIExposeFrame FLICancelExposure FLISetExposureTime	

3.19

LIBFLIAPI **FLISetTemperature** (flidev_t dev, double temperature)

Set the temperature of a given camera.

Set the temperature of a given camera. This function sets the temperature of the CCD camera `dev` to `temperature` degrees Celsius. The valid range of the `temperature` parameter is from -55 C to 45 C.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Camera device to set the temperature of.
	`temperature`	Temperature in Celsius to set CCD camera cold finger to.
See Also:	FLIGetTemperature	

3.20

LIBFLIAPI **FLIGetTemperature** (flidev_t dev, double* temperature)

Get the temperature of a given camera.

Get the temperature of a given camera. This function places the temperature of the CCD camera cold finger of device `dev` in the location pointed to by `temperature`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Camera device to get the temperature of.
	`temperature`	Pointer to where the temperature will be placed.
See Also:	FLISetTemperature	

3.21

LIBFLIAPI **FLIGrabRow** (flidev_t dev, void* buff, size_t width)

Grab a row of an image.

Grab a row of an image. This function grabs the next available row of the image from camera device `dev`. The row of width `width` is placed in the buffer pointed to by `buff`. The size of the buffer pointed to by `buff` must take into account the bit depth of the image, meaning the buffer size must be at least `width` bytes for an 8-bit image, and at least 2*`width` for a 16-bit image.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Camera whose image to grab the next available row from.
	`buff`	Pointer to where the next available row will be placed.
	`width`	Row width in pixels.
See Also:	FLIGrabFrame	

3.22

LIBFLIAPI **FLIExposeFrame** (flidev_t dev)

Expose a frame for a given camera.

Expose a frame for a given camera. This function exposes a frame according to the settings (image area, exposure time, bit depth, etc.) of camera `dev`. The settings of `dev` must be valid for the camera device. They are set by calling the appropriate set library functions. This function returns after the exposure has started.

Return Value:

`Zero`	on success.
`Non-zero`	on failure.

Parameters: `dev` Camera to expose the frame of.

See Also:
FLISetExposureTime
FLISetFrameType
FLISetImageArea
FLISetHBin
FLISetVBin
FLISetNFlushes
FLISetBitDepth
FLIGrabFrame
FLICancelExposure
FLIGetExposureStatus

3.23

LIBFLIAPI **FLISetBitDepth** (flidev_t dev, flibitdepth_t bitdepth)

Set the gray-scale bit depth for a given camera.

Set the gray-scale bit depth for a given camera. This function sets the gray-scale bit depth of camera `dev` to `bitdepth`. The `bitdepth` parameter is either `FLI_MODE_8BIT` for 8-bit mode or `FLI_MODE_16BIT` for 16-bit mode. Many cameras do not support this mode.

Return Value:

`Zero`	on success.
`Non-zero`	on failure.

Parameters:

`dev`	Camera to set the bit depth of.
`bitdepth`	Gray-scale bit depth: `FLI_MODE_8BIT` or`FLI_MODE_16BIT`.

See Also:
flibitdepth_t
FLIExposeFrame

3.24

LIBFLIAPI **FLISetNFlushes** (flidev_t dev, long nflushes)

Set the number of flushes for a given camera.

Set the number of flushes for a given camera. This function sets the number of times the CCD array of camera `dev` is flushed by the FLIExposeFrame *before* exposing a frame to `nflushes`. The valid range of the `nflushes` parameter is from 0 to 16. Some FLI cameras support background flushing. Background flushing continuously flushes the CCD eliminating the need for pre-exposure flushing.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Camera to set the number of flushes of.
	`nflushes`	Number of times to flush CCD array before anexposure.
See Also:	FLIFlushRow FLIExposeFrame FLIControlBackgroundFlush	

3.25

LIBFLIAPI **FLIReadIOPort** (flidev_t dev, long* ioportset)

Read the I/O port of a given camera.

Read the I/O port of a given camera. This function reads the I/O port on camera `dev` and places the value in the location pointed to by `ioportset`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Camera to read the I/O port of.
	`ioportset`	Pointer to where the I/O port data will be stored.
See Also:	FLIWriteIOPort FLIConfigureIOPort	

3.26

LIBFLIAPI **FLIWriteIOPort** (flidev_t dev, long ioportset)

Write to the I/O port of a given camera.

Write to the I/O port of a given camera. This function writes the value `ioportset` to the I/O port on camera `dev`.

Return Value: `Zero` on success.
`Non-zero` on failure.

Parameters: `dev` Camera to write I/O port of.
`ioportset` Data to be written to the I/O port.

See Also: FLIReadIOPort
FLIConfigureIOPort

3.27

LIBFLIAPI **FLIConfigureIOPort** (flidev_t dev, long ioportset)

Configure the I/O port of a given camera.

Configure the I/O port of a given camera. This function configures the I/O port on camera `dev` with the value `ioportset`.

The I/O configuration of each pin on a given camera is determined by the value of `ioportset`. Setting a respective I/O bit enables the port bit for output while clearing an I/O bit enables to port bit for input. By default, all I/O ports are configured as inputs.

Return Value: `Zero` on success.
`Non-zero` on failure.

Parameters: `dev` Camera to configure the I/O port of.
`ioportset` Data to configure the I/O port with.

See Also: FLIReadIOPort
FLIWriteIOPort

3.28

LIBFLIAPI **FLILockDevice** (flidev_t dev)

Lock a specified device.

Lock a specified device. This function establishes an exclusive lock (mutex) on the given device to prevent access to the device by any other function or process.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Device to lock.
See Also:	FLIUnlockDevice	

3.29

LIBFLIAPI **FLIUnlockDevice** (flidev_t dev)

Unlock a specified device.

Unlock a specified device. This function releases a previously established exclusive lock (mutex) on the given device to allow access to the device by any other function or process.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Device to unlock.
See Also:	FLILockDevice	

3.30

LIBFLIAPI **FLIControlShutter** (flidev_t dev, flishutter_t shutter)

Control the shutter on a given camera.

Control the shutter on a given camera. This function controls the shutter function on camera `dev` according to the `shutter` parameter.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Device to control the shutter of.
	`shutter`	How to control the shutter. A value of`FLI_SHUTTER_CLOSE` closes the shutter and`FLI_SHUTTER_OPEN` opens the shutter.`FLI_SHUTTER_EXTERNAL_TRIGGER_LOW`, `FLI_SHUTTER_EXTERNAL_TRIGGER`causes the exposure to beginonly when a logic LOW is detected on I/O port bit 0.`FLI_SHUTTER_EXTERNAL_TRIGGER_HIGH` causes the exposure to beginonly when a logic HIGH is detected on I/O port bit 0. This settingmay not be available on all cameras.
See Also:	flishutter_t	

3.31

LIBFLIAPI **FLIControlBackgroundFlush** (flidev_t dev, flibgflush_t bgflush)

Enables background flushing of CCD array.

Enables background flushing of CCD array. This function enables the background flushing of the CCD array camera `dev` according to the `bgflush` parameter. Note that this function may not succeed on all FLI products as this feature may not be available.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.

Parameters:	`dev`	Device to control the background flushing of.
	`bgflush`	Enables or disables background flushing. A value of`FLI_BGFLUSH_START` begins background flushing. It is important tonote that background flushing is stopped whenever `FLIExposeFrame()`or `FLIControlShutter()` are called. `FLI_BGFLUSH_STOP` stops allbackground flush activity.
See Also:	flibgflush_t	

3.32

LIBFLIAPI **FLIList** (flidomain_t domain, char*** names)

List available devices.

List available devices. This function returns a pointer to a NULL terminated list of device names. The pointer should be freed later with `FLIFreeList()`. Each device name in the returned list includes the filename needed by `FLIOpen()`, a separating semicolon, followed by the model name or user assigned device name.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`domain`	Domain to list the devices of. This is a bitwiseORed combination of interface method and device type. Validinterfaces include `FLIDOMAIN_PARALLEL_PORT`,`FLIDOMAIN_USB`, `FLIDOMAIN_SERIAL`, and`FLIDOMAIN_INET`. Valid device types include`FLIDEVICE_CAMERA`, `FLIDOMAIN_FILTERWHEEL`, and`FLIDOMAIN_FOCUSER`.
	`names`	Pointer to where the device name list will be placed.
See Also:	flidomain_t FLIFreeList FLIOpen	

3.33

LIBFLIAPI **FLIFreeList** (char** names)

Free a previously generated device list.

Free a previously generated device list. Use this function after `FLIList()` to free the list of device names.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`names`	Pointer to the list.
See Also:	FLIList	

3.34

LIBFLIAPI **FLISetFilterPos** (flidev_t dev, long filter)

Set the filter wheel position of a given device.

Set the filter wheel position of a given device. Use this function to set the filter wheel position of `dev` to `filter`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Filter wheel device handle.
	`filter`	Desired filter wheel position.
See Also:	FLIGetFilterPos	

3.35

LIBFLIAPI **FLIGetFilterPos** (flidev_t dev, long* filter)

Get the filter wheel position of a given device.

Get the filter wheel position of a given device. Use this function to get the filter wheel position of `dev`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Filter wheel device handle.
	`filter`	Pointer to where the filter wheel position will beplaced.
See Also:	FLISetFilterPos	

3.36

LIBFLIAPI **FLIGetStepsRemaining** (flidev_t dev, long* steps)

Get the number of motor steps remaining.

Get the number of motor steps remaining. Use this function to determine if the stepper motor of `dev` is still moving.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Filter wheel device handle.
	`filter`	Pointer to where the number of remaning steps will beplaced.
See Also:	FLISetFilterPos	

3.37

LIBFLIAPI **FLIGetFilterCount** (flidev_t dev, long* filter)

Get the filter wheel filter count of a given device.

Get the filter wheel filter count of a given device. Use this function to get the filter count of filter wheel `dev`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Filter wheel device handle.
	`filter`	Pointer to where the filter wheel filter count willbe placed.

3.38

LIBFLIAPI **FLIStepMotorAsync** (flidev_t dev, long steps)

Step the filter wheel or focuser motor of a given device.

Step the filter wheel or focuser motor of a given device. Use this function to move the focuser or filter wheel `dev` by an amount `steps`. This function is non-blocking.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Filter wheel or focuser device handle.
	`steps`	Number of steps to move the focuser or filter wheel.
See Also:	FLIGetStepperPosition	

3.39

LIBFLIAPI **FLIStepMotor** (flidev_t dev, long steps)

Step the filter wheel or focuser motor of a given device.

Step the filter wheel or focuser motor of a given device. Use this function to move the focuser or filter wheel `dev` by an amount `steps`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Filter wheel or focuser device handle.
	`steps`	Number of steps to move the focuser or filter wheel.
See Also:	FLIGetStepperPosition	

3.40 LIBFLIAPI **FLIGetStepperPosition** (flidev_t dev, long* position)

Get the stepper motor position of a given device.

Get the stepper motor position of a given device. Use this function to read the stepper motor position of filter wheel or focuser `dev`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Filter wheel or focuser device handle.
	`position`	Pointer to where the postion of the stepper motorwill be placed.
See Also:	FLIStepMotor	

3.41 LIBFLIAPI **FLIHomeFocuser** (flidev_t dev)

Home focuser `dev`.

Home focuser `dev`. The home position is closed as far as mechanically possiable.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Focuser device handle.

3.42

LIBFLIAPI **FLIGetFocuserExtent** (flidev_t dev, long* extent)

Retreive the maximum extent for FLI focuser `dev`

Retreive the maximum extent for FLI focuser `dev`

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Focuser device handle.
	`extent`	Pointer to where the maximum extent of the focuser will be placed.

3.43

LIBFLIAPI **FLIReadTemperature** (flidev_t dev, flichannel_t channel, double* temperature)

Retreive temperature from the FLI focuser `dev`*.*

Retreive temperature from the FLI focuser `dev`. Valid channels are `FLI_TEMPERATURE_INTERNAL` and `FLI_TEMPERATURE_EXTERNAL`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`dev`	Focuser device handle.
	`channel`	Channel to be read.
	`extent`	Pointer to where the channel temperature will be placed.

3.44

LIBFLIAPI **FLICreateList** (flidomain_t domain)

Creates a list of all devices within a specified `domain`.

Creates a list of all devices within a specified `domain`. Use `FLIDeleteList()` to delete the list created with this function. This function is the first called begin the iteration through the list of current FLI devices attached.

Return Value:

`Zero`	on success.
`Non-zero`	on failure.

Parameters: `domain` Domain to search for devices, set to zero to search all domains.This parameter must contain the device type.

See Also: FLIDeleteList
FLIListFirst
FLIListNext

3.45

LIBFLIAPI **FLIDeleteList** (void)

Deletes a list of devices created by `FLICreateList()`

Deletes a list of devices created by `FLICreateList()`

Return Value:

`Zero`	on success.
`Non-zero`	on failure.

See Also: FLICreateList
FLIListFirst
FLIListNext

3.46

LIBFLIAPI **FLIListFirst** (flidomain_t* domain, char* filename, size_t fnlen, char* name, size_t namelen)

Obtains the first device in the list.

Obtains the first device in the list. Use this function to get the first domain, filename and name from the list of attached FLI devices created using the function FLICreateList(). Use FLIListNext() to obtain more found devices.

Return Value:	Zero	on success.
	Non-zero	on failure.
Parameters:	domain	Pointer to where to domain of the device will be placed.
	filename	Pointer to where the filename of the device will be placed.
	fnlen	Length of the supplied buffer to hold the filename.
	name	Pointer to where the name of the device will be placed.
	namelen	Length of the supplied buffer to hold the name.
See Also:	FLICreateList FLIDeleteList FLIListNext	

3.47

LIBFLIAPI **FLIListNext** (flidomain_t* domain, char* filename, size_t fnlen, char* name, size_t namelen)

Obtains the next device in the list.

Obtains the next device in the list. Use this function to get the next `domain`, `filename` and `name` from the list of attached FLI devices created using the function `FLICreateList()`.

Return Value:	`Zero`	on success.
	`Non-zero`	on failure.
Parameters:	`domain`	Pointer to where to domain of the device will be placed.
	`filename`	Pointer to where the filename of the device will be placed.
	`fnlen`	Length of the supplied buffer to hold the filename.
	`name`	Pointer to where the name of the device will be placed.
	`namelen`	Length of the supplied buffer to hold the name.
See Also:	FLICreateList FLIDeleteList FLIListFirst	