Electronic Filter Wheel

# Remote Operations Guide



# OPTECALI-ZEPTO, Inc.

# [www.optecali.com](http://www.optecali.com)

# email: [info@optecali.com](mailto:info@optecali.com)

# Table of Contents

Introduction 3

Installation of Filters 3

Connecting EFW to Camera 4

Driver Installation 5

Troubleshooting Common Problems 5

Introduction - Software 5

Definition of enum-type and struct 6

Typedef enum\_EFW\_ERROR\_CODE 6

Typedef enum\_EFW\_INFO 6

Function declaration 7

EFW Get Num 7

EFW Get ID 7

EFW Get Property 7

EFW Open 8

EFW Get Position 8

EFW Set Position 9

EFW Set Direction 9

EFW Get Direction 10

EFW Close 10

EFW Get Product IDs 10

EFW Calibrate 11

Suggested Call Sequence 11

Contact Information 12

## Introduction

Thank you for purchasing the Electronic Filter Wheel (EFW). Please read this Quick Start Guide for further information and if you have any additional questions, please contact us at info@optecali.com

Electronic filter wheels (EFW) are filter wheels used with mono cameras for astrophotography. Optecali offers several configurations depending on the filter size and number of filters used. A EFW simply requires a USB connection for use and power. You may control it from a variety of hosts such as Windows, Arm Linux, Android and iOS system.

Four configurations of the EFW are currently available:

* + 5 Position: Accepts 1”
  + 5 Position: Accepts 2



## Installation of Filters

1′′ & 2" size mounted filters: (Please note that filters must be less than 7mm in thickness (not including the threads) and threads must be less than 3mm in thickness.)

* + Remove the screws securing the back cover and remove the back cover
  + Set the wheel on a flat surface with the face down
  + Screw the filters into the threaded wheel positions
  + Replace the back cover and tighten the screws

The following lens adapter is necessary for usage of the filter wheel

* + SM1 female to M28.5x0.6 male thread adapter
  + For usage attach the threaded side of the lens adapter to the filter wheel



## 

## Connecting EFW to Camera

You can connect your camera to the EFW with a 1" nose piece or simply thread it directly to the M42 interface. Please pay attention to the direction of EFW when you connect to the camera to ensure that the filters are as close to the sensor as possible to minimize the chance of vignetting.

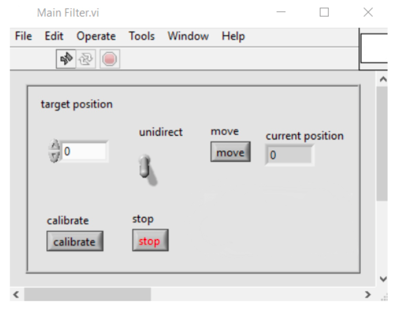


## Driver Installation

The EFW are designed as a USB HID device (like a keyboard or mouse) so you only need to install the ‘Filter Wheel Utility’ driver for use.

Download and install the Filter wheel Utility that is included with your EFW.

-Important: Make sure you also install ‘My Installer’ set up application first.

Application

Description automatically generated

## Troubleshooting Common Problems

* 1. My EFW tries to move but gets stuck and cannot arrive at the right position.
     + Filter is too thick: Filters (not including the threads) must be less than 7mm in thickness and threads must be less than 3mm in thickness.
     + The M2 screw used to secure the filter is too long and causes the wheel to jam.
     + The M42 thread of the scope or the camera is too long and protrudes into the filter wheel, hitting the filter or wheel, causing it to jam.

2. My EFW spins but never stops or stops at the wrong position.

The EFW uses an infrared sensor to check wheel position. If the sensor malfunctions, you can usually resolve by recalibrating the EFW in the driver. Simpl

y click "calibrate" in the driver interface. It can take up to 60 seconds to perform this action.

If this does not resolve your problem, contact us at: info@optecali.com

## Introduction - Software

This SDK is used to operate EFW serial filter wheel, can be used by C, C++, C# and other develop tools, is suit for Windows, Linux, OSX operating system of x86 and x64.

Header file: EFW\_filter.h

Under Windows the import library and dynamic library: EFW\_filter.lib、EFW\_filter.dll

Under Linux the dynamic library and static library: EFW\_filter.so、EFW\_filter.a

Under OSX the dynamic library and static library: EFW\_filter.dylib、EFW\_filter.a

Installation method:

Under Windows, extract the downloaded zip file to any directory, and

add DLL's path to system environment variables, sometimes logout and re-login is

required.

## Definition of enum-type and struct

## Typedef enum\_EFW\_ERROR\_CODE

{EFW\_SUCCESS = 0,

EFW\_ERROR\_INVALID\_INDEX,

EFW\_ERROR\_INVALID\_ID,

EFW\_ERROR\_INVALID\_VALUE,

EFW\_ERROR\_CLOSED, //not opened

EFW\_ERROR\_REMOVED, //failed to find the filter wheel, maybe the filter wheel has been

removed

EFW\_ERROR\_MOVING,//filter wheel is moving

EFW\_ERROR\_GENERAL\_ERROR,//other error

EFW\_ERROR\_CLOSED,

EFW\_ERROR\_END = -1

}EFW\_ERROR\_CODE;

Returned error code

## Typedef enum\_EFW\_INFO

{int ID;

char Name[64];

int slotNum;

} EFW\_INFO;

Filter wheel information

## Function declaration

## EFW Get Num

Syntax： int EFWGetNum()

Descriptions :

This should be the first API to be called, get number of connected EFW filter wheel, call

this API to refresh device list if EFW is connected or disconnected.

Return: number of connected EFW filter wheel. 1 means 1 filter wheel is connected.

## EFW Get ID

Syntax: EFW\_ERROR\_CODE EFWGetID(int index, int\* ID)

Descriptions :

Get ID of filter wheel

Paras:

int index: the index of filter wheel, from 0 to N - 1, N is returned by EFWGetNum()

int\* ID: pointer to ID. if the filter wheel is not opened, the ID is negative, otherwise the ID

is a unique

integer between 0 to EFW\_ID\_MAX - 1, after opened, all the operation is base on this

ID, the ID will

not change before the filter wheel is closed.

Return:

EFW\_ERROR\_INVALID\_INDEX: index value is invalid

EFW\_SUCCESS: operation succeeds

## EFW Get Property

Syntax: EFW\_ERROR\_CODE EFWGetProperty(int ID, EFW\_INFO \*pInfo)

Descriptions :

Get property of filter wheel.

Paras:

int ID: the ID of filter wheel

EFW\_INFO \*pInfo: pointer to structure containing the property of EFW

Return:

EFW\_ERROR\_INVALID\_ID: invalid ID value

EFW\_ERROR\_CLOSED: the filter wheel is closed

EFW\_ERROR\_MOVING: slot number detection is in progress, generally this error will

happen soon after filter wheel is connected.

EFW\_SUCCESS: operation succeeds

## EFW Open

Syntax: EFW\_ERROR\_CODE EFWOpen(int ID)

Descriptions :

Open filter wheel

Paras:

int ID: the ID of filter wheel

Return:

EFW\_ERROR\_INVALID\_ID: invalid ID value

EFW\_ERROR\_GENERAL\_ERROR: number of opened filter wheel reaches the maximum

value.

EFW\_ERROR\_REMOVED: the filter wheel is removed.

EFW\_SUCCESS: operation succeeds

## EFW Get Position

Syntax: EFW\_ERROR\_CODE EFWGetPosition(int ID, int \*pPosition)

Descriptions :

Get position of slot

Paras:

int ID: the ID of filter wheel

int \*pPosition: pointer to slot position, this value is between 0 to M - 1, M is slot number

this value is -1 if filter wheel is moving

Return:

EFW\_ERROR\_INVALID\_ID: invalid ID value

EFW\_ERROR\_CLOSED: the filter wheel is closed

EFW\_SUCCESS: operation succeeds

EFW\_ERROR\_ERROR\_STATE: filter wheel is in error state.

## EFW Set Position

Syntax: EFW\_ERROR\_CODE EFWSetPosition(int ID, int Position)

Descriptions :

Set position of slot

Paras:

int ID: the ID of filter wheel

int Position: slot position, this value is between 0 to M - 1, M is slot number

Return:

EFW\_ERROR\_INVALID\_ID: invalid ID value

EFW\_ERROR\_CLOSED: the filter wheel is closed

EFW\_SUCCESS: operation succeeds

EFW\_ERROR\_INVALID\_VALUE: Position value is invalid

EFW\_ERROR\_MOVING: filter wheel is moving, should wait until idle

EFW\_ERROR\_ERROR\_STATE: filter wheel is in error state

## EFW Set Direction

Syntax: EFW\_ERROR\_CODE EFWSetDirection(int ID, bool bUnidirectional)

Descriptions :

Set unidirection of filter wheel

Paras:

int ID: the ID of filter wheel

bool bUnidirectional: if set as true, the filter wheel will rotate along one direction

Return:

EFW\_ERROR\_INVALID\_ID: invalid ID value

EFW\_ERROR\_CLOSED: the filter wheel is closed

EFW\_SUCCESS: operation succeeds

## EFW Get Direction

## 

Syntax: EFW\_ERROR\_CODE EFWGetDirection(int ID, bool \*bUnidirectional)

Descriptions :

Get unidirection of filter wheel

Paras:

int ID: the ID of filter wheel

bool \*bUnidirectional: pointer to unidirection value .

Return:

EFW\_ERROR\_INVALID\_ID: invalid ID value

EFW\_ERROR\_CLOSED: the filter wheel is closed

EFW\_SUCCESS: operation succeeds

## EFW Close

Syntax: EFW\_ERROR\_CODE EFWClose(int ID)

Descriptions :

Close filter wheel

Paras:

int ID: the ID of filter wheel

Return:

EFW\_ERROR\_INVALID\_ID: invalid ID value

EFW\_SUCCESS: operation succeeds

## EFW Get Product IDs

## 

Syntax: int EFWGetProductIDs(int\* pPIDs)

Descriptions :

get the product ID of each wheel, at first set pPIDs as 0 and get length and then malloc a buffer to load

the PIDs

Paras:

int\* pPIDs: pointer to array of PIDs

Return: length of the array.

## 

## EFW Calibrate

Syntax: EFW\_ERROR\_CODE EFWCalibrate(int ID)

Descriptions :

calibrate filter wheel

Paras:

int ID: the ID of filter wheel

Return:

EFW\_ERROR\_INVALID\_ID: invalid ID value

EFW\_ERROR\_CLOSED: not opened

EFW\_SUCCESS: operation succeeds

EFW\_ERROR\_MOVING: filter wheel is moving, should wait until idle

EFW\_ERROR\_ERROR\_STATE: filter wheel is in error state

EFW\_ERROR\_REMOVED: filter wheel is removed

## Suggested Call Sequence

Get count of connected filter wheels--> EFWGetNum

Get filter wheels' ID-> EFWGetID

Get filter wheels' name--> EFWGetProperty

Open filter wheel --> EFWOpen（Notes： this SDK can operate multiple filter wheels，distinguish by

each filter wheel's ID）

Rotate--> EFWSetPosition

Close filter wheel-->EFWClose

## Contact Information

Customer and Technical Support

OPTECALI-ZEPTO

Email: [info@optecali.com](mailto:seung.kim@optecali.com)

Website: [www.optecali.com](http://www.optecali.com)