illunis Canon Lens Controller

illunis RFLC Manual.

Canon Lens Controller
EMC and M58 Mounts

INTRODUCTION

This document details the setup and operation of the illunis RFLC Canon Lens Controller.

Canon's RF lens mount debuted 30 years after Canon's EF mount was introduced, the RF mount was designed to enable a new generation of optical technology.

The Canon RF mount retains the same wide 54mm diameter as the EF mount, but with a reduction in the back focus distance – the distance between the mount and the sensor – from 44mm in the EF mount to 20mm in the RF mount. The RFLC is available in EMC custom mount and a industry standard M58 mount.

One benefit of the RF mount is the Lens Control Ring featured on all RF lenses in addition to the standard focus and zoom rings. This ring can be customized to control focus or aperture.

Canon RF lenses are better than EF lenses because the RF mount architecture enables much faster communication between the camera and the lens, much greater data transfer, and support for the latest focusing, image stabilization and optical technologies. The reduced back focus distance also allows for lens designs with no performance compromises.

For illunis EMC

Standard M58 Mount

Notes

This document explains the command interface to the RFLC EMC Canon Lens controller. A USB and UART hardware interface are available. Both versions use identical firmware.

The following table outlines the features.

Feature	UART	USB
Power	5.0V	USB (5.0V)
Connector	JST 4 pin	USB Type C Light proof seal
Serial Port Baud	115,200 only	115,200 only
Field programmable	Yes	Yes
Internal EEPROM User accessible	Yes (1K)	Yes (1K)
Use Case	Embedded system	Windows/Linux PC
Wi-Fi interface	Yes	Yes
Locking Bayonet	Yes	Yes
Locking Connector	No	Yes

Rev	Date	Modification
A	07/30/2024	First Revision
B	07/13/2024	Added product ordering information WIFI commands
C	08/26/24	Added Part Numbers and pricing.
D	09/04/24	Added Focus Lock, Dial controller.
E	10/7/24	Added Wifi and controller dial. Updated app.

Note: Product information subject to change without prior notice.

Product Ordering Information

1pc USA pricing shown, Call for volume OEM pricing and distribution pricing.

Name	Description	P#	MSRP
RFLC-EMC	Canon RF mount lens controller for illunis EMC cameras. 2nd generation mount with Wi-Fi, locking bayonet, locking USB-C.	13-03722	$600 USD
RFLC-M58	Canon RF mount lens controller for industry standard M58 screw mount. 2nd generation mount with Wi-Fi, locking bayonet, locking USB-C.	13-03721	$600 USD
EFLC-EMC	Canon EF mount lens controller for illunis EMC cameras. 2nd generation mount with Wi-Fi, locking bayonet, locking USB-C.	13-03724	$600 USD
EFLC-M58	Canon EF mount lens controller for industry standard M58 screw mount. 2nd generation mount with Wi-Fi, locking bayonet, locking USB-C.	13-037212	$600 USD
EFLC-OEM	Canon EF mount lens controller OEM mount. 2nd generation mount with Wi-Fi, locking bayonet, locking USB-C.	13-03725	$Call for OEM volume pricing.
EFLC-CMOUNT	Canon EF mount lens controller for C-Mount cameras. 2nd generation mount with Wi-Fi, locking bayonet, locking USB-C.	13-03726	$600 USD
Dial Controller	Battery powered, hand held WiFi controller for the the RFLC/EFLC lens controllers.	49-03733	$300

Table of Contents

Important Note:

If the focus position is not moving, check the position of the lens AF/MF switch.

Quick Start

The illunis Canon Lens Controller (RFLC) is a mechanical lens mount for a Canon RF lens with an integrated lens controller circuit board. The lens controller USB Version uses a virtual communication port to send and receive commands via a USB 2.0 connection to a computer. The UART implementation is intended for embedded use. The RFLC is available in EMC custom mount and a industry standard M58 mount.

USB drivers are include in the standard windows installation.

Comm Port Setup

The lens controller port settings are as follows:

Baud Rate:	115,200 recommended (The port has baud rate sensing)
Parity:	None
Data Bits:	8
Stop Bits:	1
Flow Control:	RTS/CTS (for best results)

Cables

The RFLC uses a USB type C connector; any commercial USB type C cable may be used to connect the lens controller to the PC.

The RFLC can be used with screw locking USB cables.

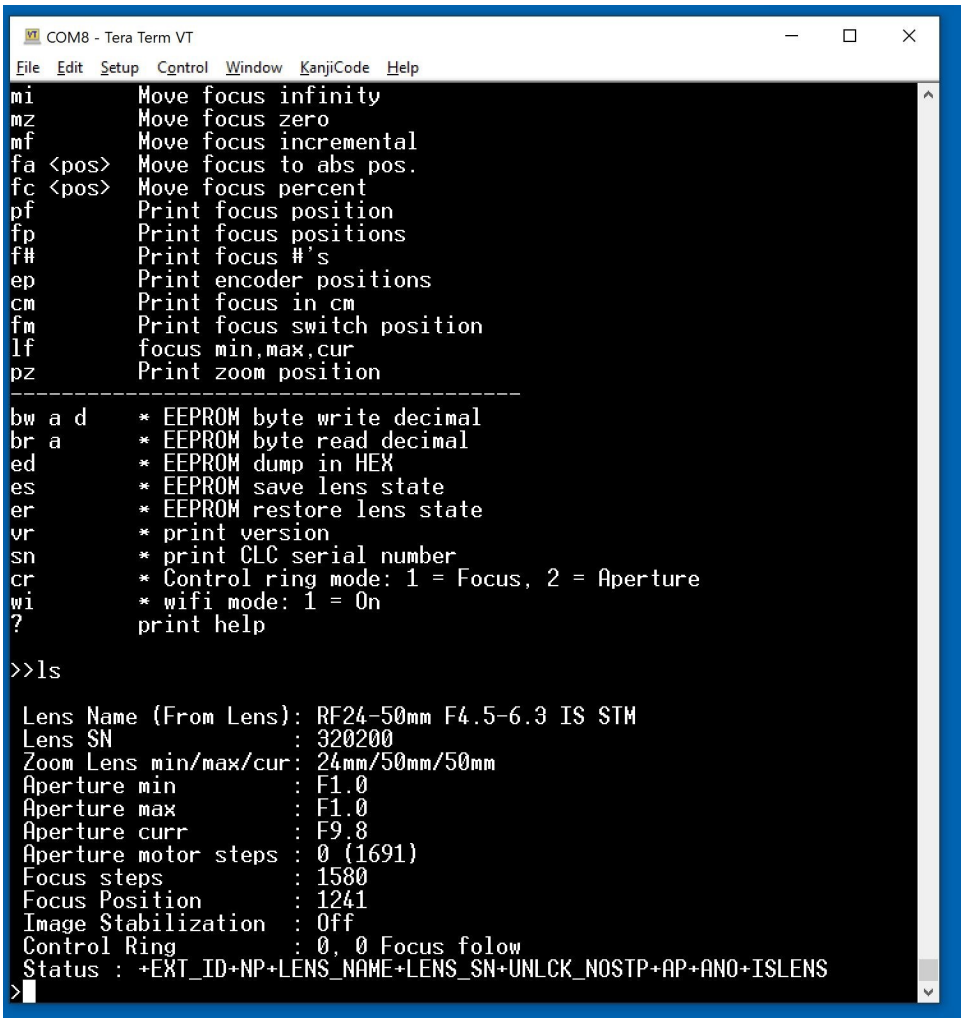
Quick Start

Lens Control using Software

To assist with writing lens control software, illunis provides a lens control program example for Visual Studio C# as well as an installable executable version. The project source code and executable are available in the illunis.com Help Center. A .Net .dll is available simplifying the configuration and communication to the lens.

Lens Control using a Terminal Program

Any lens command may simply be typed into a Terminal program such as Tera Term which is available here:
https://teratermproject.github.io/index-en.html

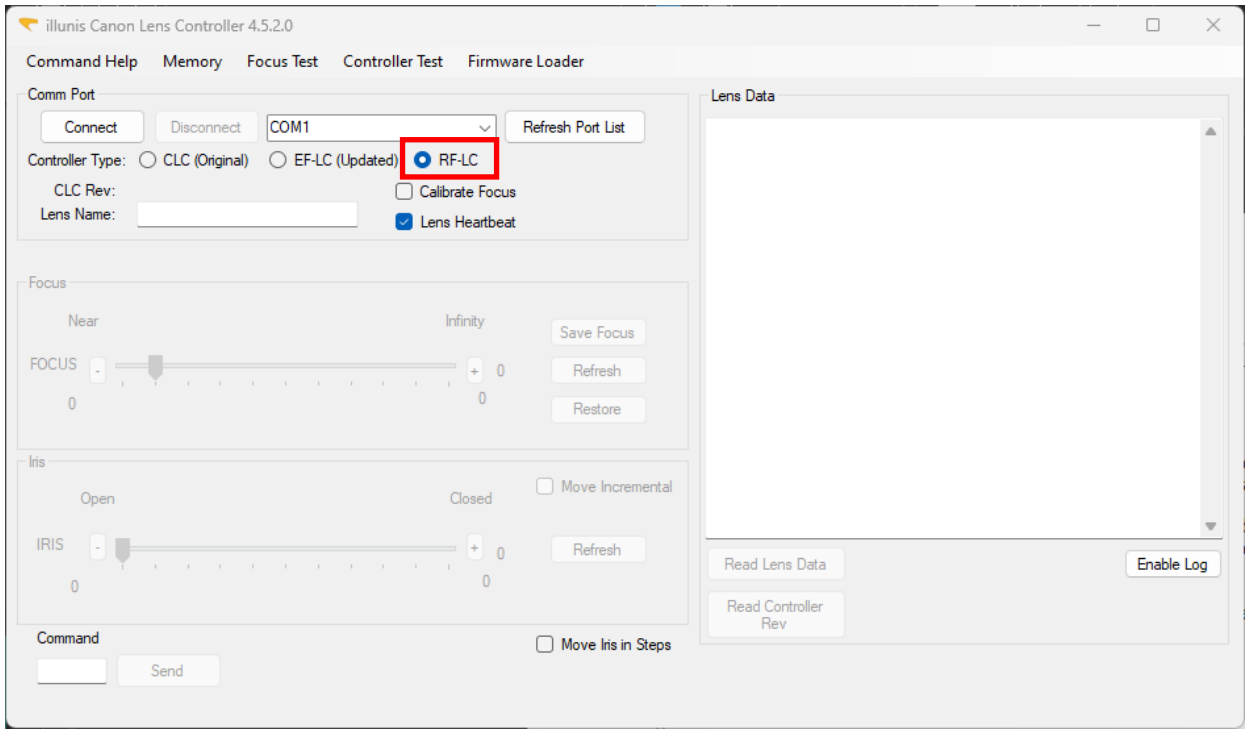


Lens Control using Software Continued

Step 1 Download illunis Lens Control Software

To assist with writing lens control software, illunis provides a lens control program example for Visual Studio C# as well as an installable executable version and lens control SDK. The sample Visual Studio Project may be opened directly in Visual Studio and compiled. It is provided to show examples of the software interface implemented in order to reduce the time spent on writing application software. A directly executable version of the application may be found in the /bin/x64/Release folder as CanonController.exe.

Contact illunis for the latest version of the Lens Control App installer or source code.



Note: The original CLC and the new RFLC require different COM port connection properties. It is important to select the correct controller type in the app before connecting.

Lens Control using a Terminal Emulator

Step 1 Obtain and install a Terminal Emulation Program

Tera Term is a free Terminal Emulator for windows available here:

https://teratermproject.github.io/index-en.html

Step 2 Run the Terminal program and issue commands from this manual to control the lens:

Set the serial baud rate to 115,200

Set the Receive to “CR”

Set the Transmit to “CR+LF”

Enable “Local Echo”

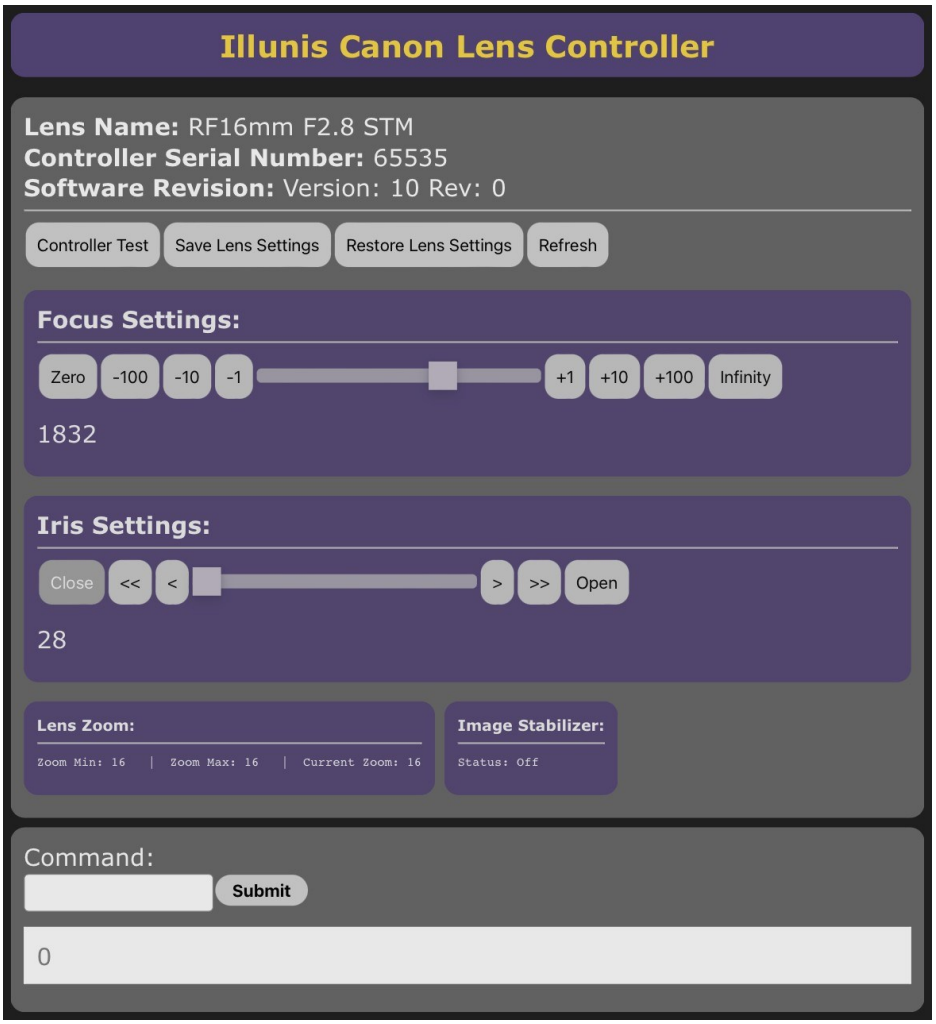
Set flow control to CTS/RTS

Lens Control using a Wi-Fi

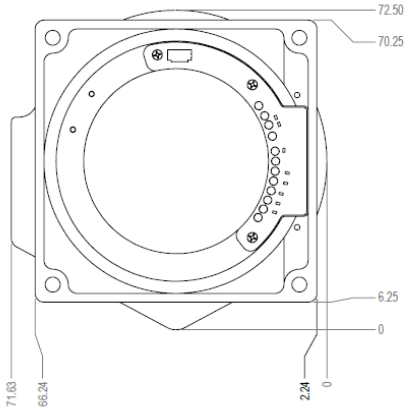
Step 1: Power up the RFLC and enable the WiFi. The RFLC requires a reset “rst” command after the WiFi state is changed.

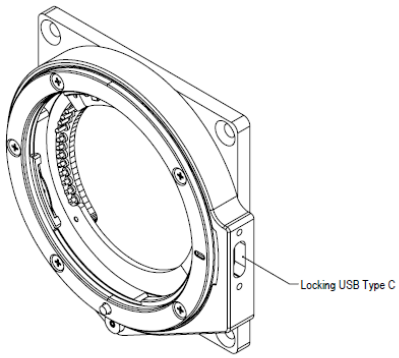
Step 2: Connect your PC, Phone, or tablet to the RFLC WiFi. “illunis RFLC *MAC address*”

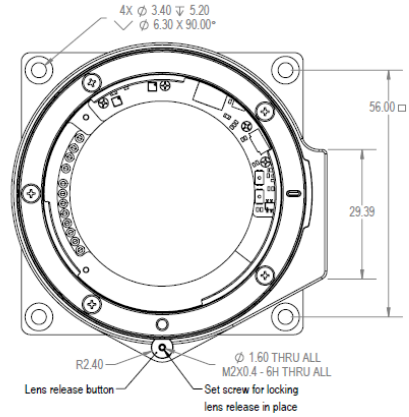
Step 3 Open a web browser and go to: http://192.168.4.1 A Wi-Fi control window will open.

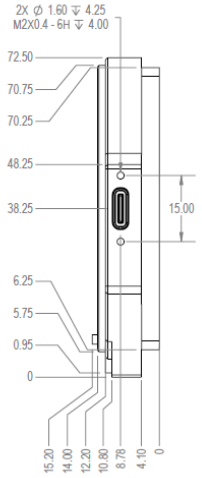


Mechanical Drawings







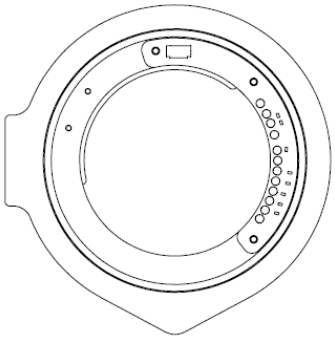


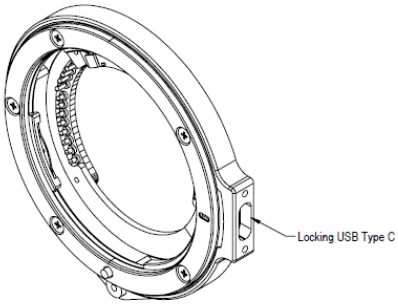
Illunis RFLC EMC Version

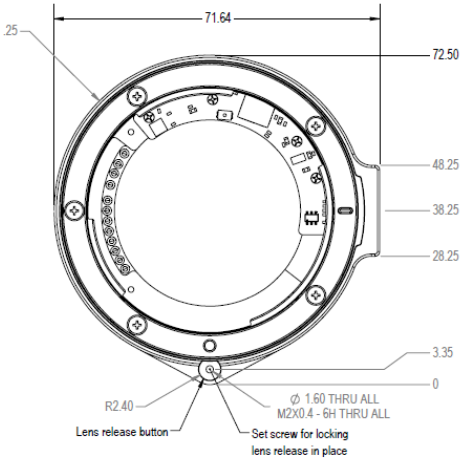
- **Mounting Interface:** EMC Camera Line
- **Data Interface:** Locking USB Type C, Wifi
- **Lens Mount:** Canon RF Mount

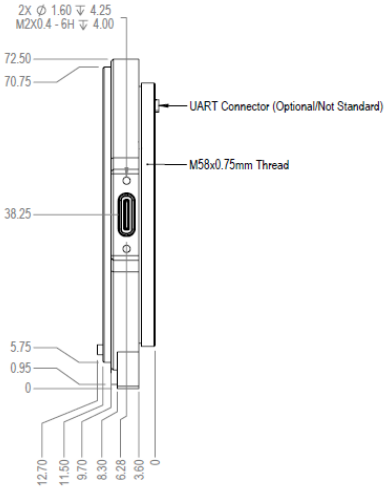
(Dimensions are in mm)

Mechanical Drawings





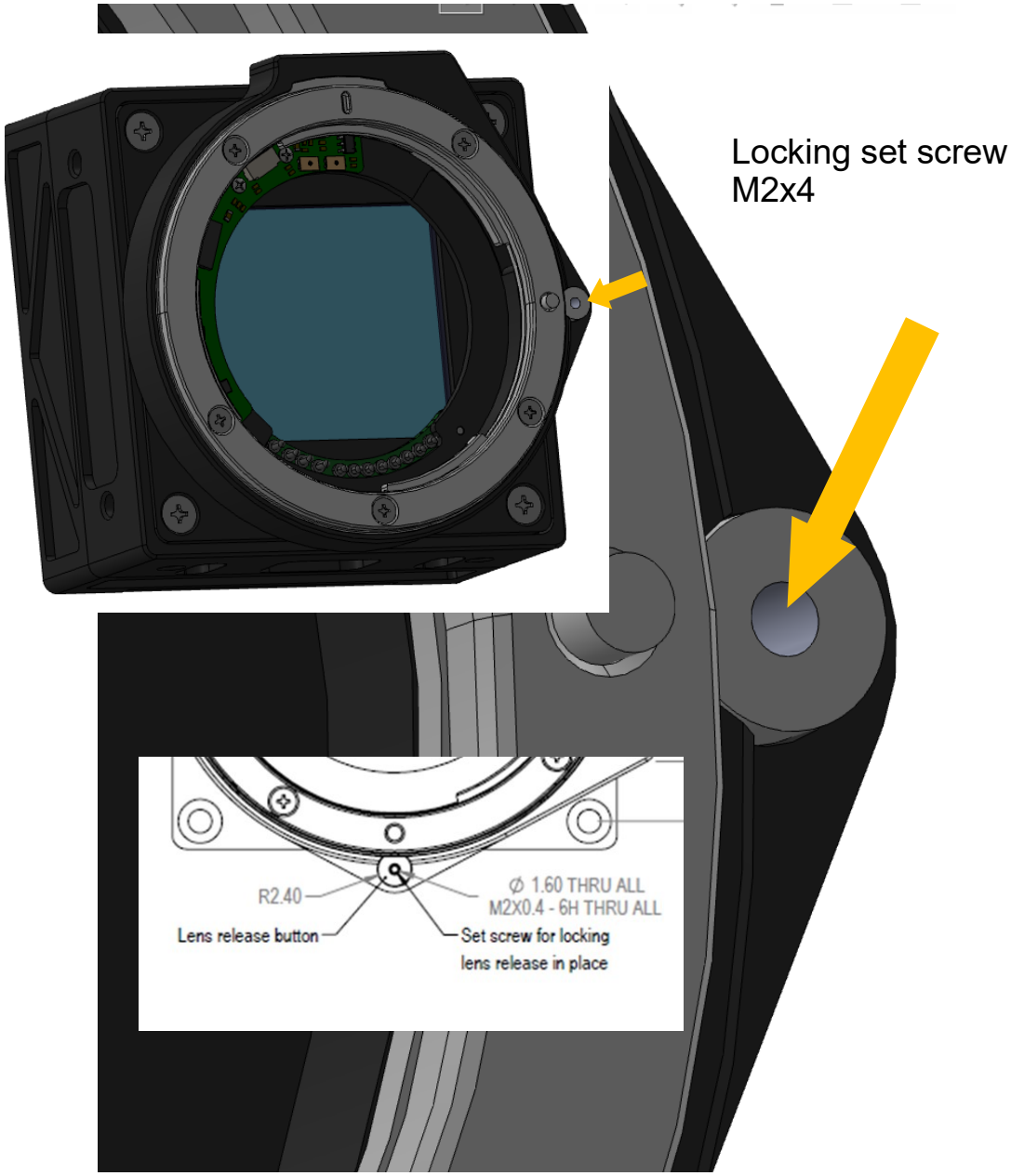




Illunis RFLC M58 Version

- **Mounting Interface:** M58x0.75mm Thread
- **Data Interface:** Locking USB Type C, Wifi
- **Lens Interface:** Canon RF Mount

(Dimensions are in mm)

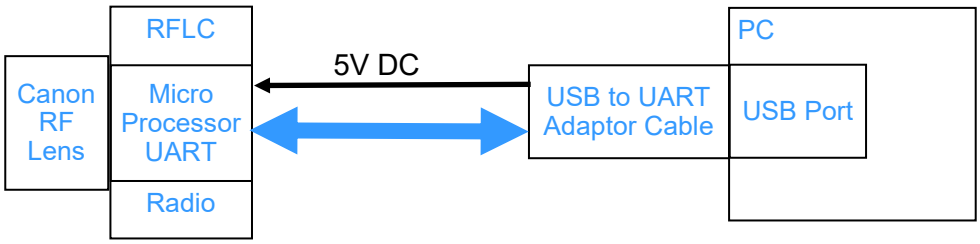


RF Lens Locking Set Screw (optional)

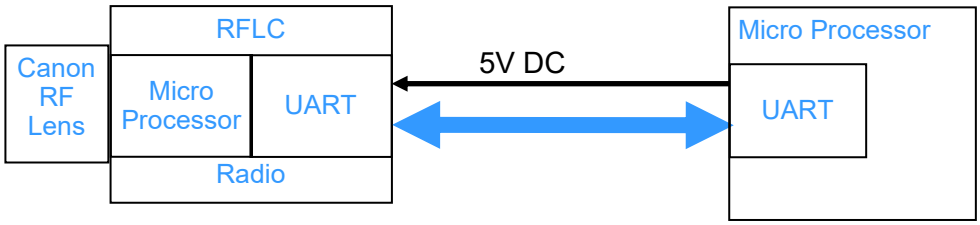
RFLC & Command Overview

The RFLC is a mechanical lens mount for Canon RF lenses with an integrated micro controller. Please note the following important items:

The RF official communication protocol is implemented. It is not reverse engineered.
The RFLC can be configured with USB or UART interfaces.
The USB interface is controlled though a Windows Com Port at 115,200 baud.
The USB connector powers the RFLC-USB.
The UART connector supplies 5V power for the RFLC



RFLC USB Block Diagram



RFLC UART Block Diagram

RFLC Command Overview Continued

The RFLC interfaces with the Canon RF mounted lens through a command protocol using a micro processor. The micro processor reads data from the RF lens, and commands the lens based on this information. The native lens data is described as follows:

Attached lens is detected by the micro processor and causes the lens to be initialized by the RFLC. This initialization performs the following: 1) The zero and infinity positions are set and the encoder/motor positions are measured. 2) The lens status, flags and aperture information are read. 3) The lens internal type code and protocol is read and decoded.

Aperture data is measured in 1/256th F-Stop increments. The F-Stop data is accessed as 10x the value of the F-Stop; thus the value reported from the lens as F28 is actually F2.8. The aperture of the fully open and fully closed positions are provided by the lens.

Focus data is measured in lens encoder and motor units. Individual lens types will have different encoder/motor ranges reported by the lens. The encoder/motor counts for infinity focus and zero focus and is measured from the lens when it is attached.

The internal EEPROM in the RFLC microprocessor can be accessed by the user. EEPROM data is read and written as bytes. An EEPROM dump command is provided. The first 144 bytes of EEPROM is reserved for RFLC use. DO NOT WRITE to these locations.

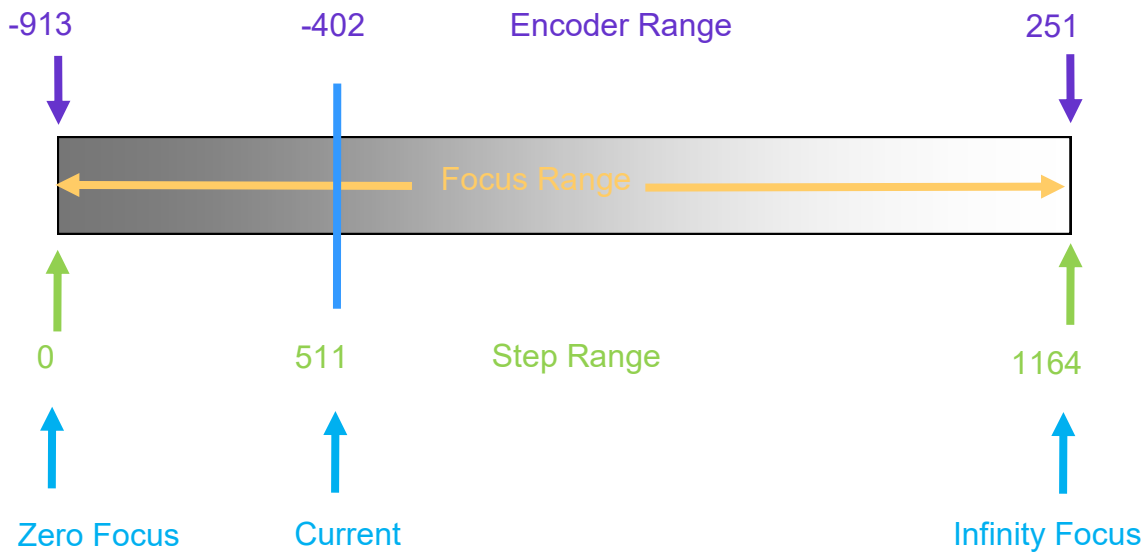
RFLC Command Overview Continued

Focus Control:

The lens internal focus mechanism controls the focus position through a stepper or ultrasonic motor in steps. The mechanism uses an encoder to determine its absolute position. The RFLC reads the encoder values and reports them in the 'f#' command. Encoder values can be negative and thus confusing to use. Please note that every lens has different encoder values. **(Note All Commands are in normalized motor units, from 0 to N)**

To simplify focus control, the RFLC calculates the focus range in steps of motor position. This allows for control in the step based numerical range. The maximum step value can be read 'fp' command which returns #steps and range. A step value of 0 is equal to 'focus zero' and a step value of #steps is equal to 'focus infinity' using the 'fa' focus absolute command.

An additional command is provided to set the focus in percent of full range. This command is 'fc' <value> where value is between 0.0 and 100.0 in a float format.



Example of focus range values

RFLC Command Overview Continued

Aperture Control:

Each lens has an aperture with various ranges. The design of the lens itself determines the range of aperture settings. The RFLC provides three methods of setting the aperture.

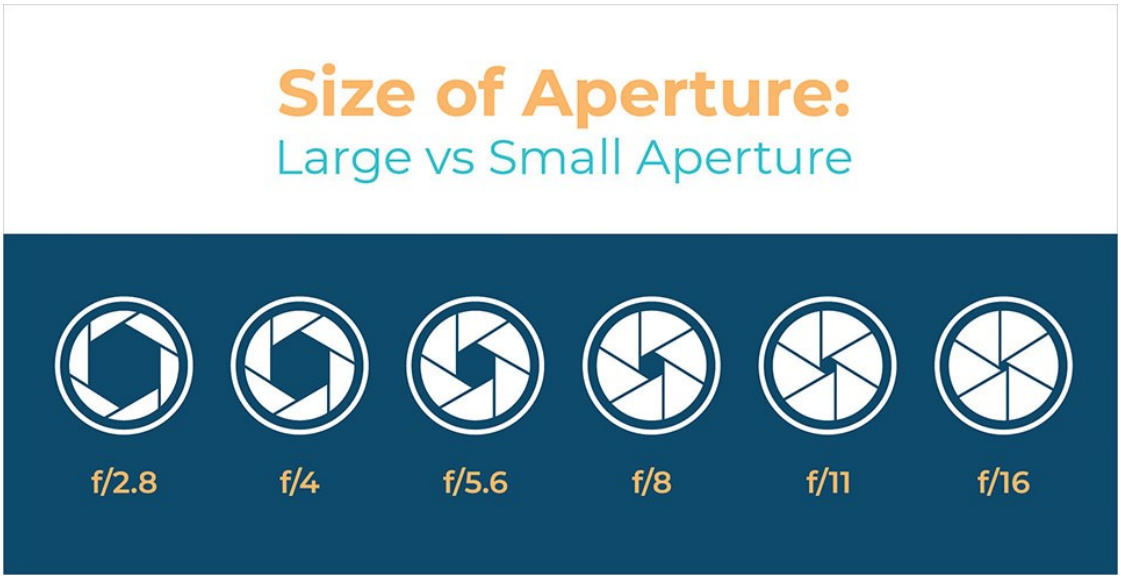
#1 The aperture can be forced full open and full closed.

#2 The aperture can be set in native (internal) lens steps.
These steps are in 1/256th of an F-stop.
The aperture can be set in absolute or incremental steps.

#3 The aperture can be set as a F-Stop.
F-Stop is specified as 10X the value requested
Example: > ms 18 <LF> = Set F-stop 1.8
Example: > ms 165 <LF> = Set F-stop 16.5

Note : Aperture display status is output as 10X the F-Stop
Example:

Aperture min	: F28	(f/2.8)
Aperture max	: F160	(f/16)
Aperture curr	: F56	(f/5.6)



RFLC Commands

The illunis RFLC uses a text based interface to command the lens and set parameters. The BAUD rate is fixed to 115,200, 8 bit, no parity. The serial interaction can be operated in the following modes

Quiet Mode : No text is sent by the RFLC unless it is commanded by the host.
Normal Mode: (Non Quiet) Camera info and command help is sent on startup
Verbose Mode: This mode sends detailed text data for each command.

Setting Quiet Mode saves the setting in EEPROM and is restored at startup. This allows for a simpler command and control interface to the RFLC-USB.

The RFLC Normal Mode detects an attached lens and displays the following text at startup:

```
RFLC Lens Controller (c) illunis LLC 2024, www.illunis.com
Lens attached : RF16mm F2.8 STM
Lens EEPROM state restored
CLC SN: 1,  Lens SN              : 280100
RFLC Commands:'*'=EEPROM        Ver: 10.1.5
----------------------------------------
la         Lens attach
ls         Lens status
ln         Lens name
sl         Lens serial number if available
is         IS status
lc         brief status: focal len,Ap:min,#steps,max
ge <#>     Get Info Cmd
qm <0/1>   * Quiet Mode
---------------------------------------
in         Initialize and open aperture
mc         Move aperture fully closed
mo         Move aperture fully open
ma <stop>  Move aperture abs. 1/256stop
mn <pos>   Move aperture inc. 1/256stop
ms <FX10>  Move aperture to f-stop (10X 22=F2.2)
ad         Print aperture info. brief
da         Print aperture info.
pa         Print aperture position
ga         Get aperture position from lens
---------------------------------------
mi         Move focus infinity
mz         Move focus zero
mf <inc>   Move focus incremental
fa <pos>   Move focus to abs pos.
fc <pos>   Move focus percent
pf         Print focus position
fp         Print focus positions
f#         Print focus #'s
ep         Print encoder positions
cm         Print focus in cm
fm         Print focus switch position
lf         focus min,max,cur
pz         Print zoom position
---------------------------------------
bw a d     * EEPROM byte write decimal
br a       * EEPROM byte read decimal
ed         * EEPROM dump in HEX
es         * EEPROM save lens state
er         * EEPROM restore lens state
cr         * Control ring mode: 1 = Focus, 2 = Aperture
wi         * wifi mode: 1 = On
ws         get/set wifi SSID (32 char max)
wp         get/set wifi password (8-32 char)
wr         reset wifi SSID and clear password
gm         print ESP32 MAC address
rst        Reset
vr         print version
sn           print CLC serial number
?            print help
```

Lens Info Commands

The RFLC detects the attached lens. The “ls” command shows lens info.:

```
>ls

 Lens Name (From Lens): RF50mm F1.2 L USM
 Lens SN             : 7400
 Prime Lens          : 50mm
 Aperture min        : F1.0
 Aperture max        : F1.0
 Aperture curr       : F1.2
 Aperture motor steps : 0 (160)
 Focus steps         : 20551
 Focus Position      : 17262
 Control Ring        : 0, 0 Focus follow
 Status : +EXT_ID+NP+LENS_NAME+LENS_SN
>
```

When a lens is dynamically detached or attached a message is displayed.:

```
>Lens detached...
Lens attached : RF85mm f/1.2 USM
Lens EEPROM state restored
>Lens detached...
Lens attached : RF85mm f/1.2 USM
Lens EEPROM state restored
>
```

Lens Info Commands Continued

RFLC Commands
- Version
- Serial number
- Quiet Mode
- Get Info

Lens Info Commands
- Lens status
- Lens attach
- Lens name
- Lens status register

Aperture Commands
- Initialize aperture
- Print aperture info
- Print aperture position
- Move aperture absolute 1/256 stop
- Move aperture fully closed
- Move aperture incremental 1/256 stop
- Move aperture fully open
- Move aperture to F-stop # (10X 22 = F2.2)

Focus Commands
- Move focus incremental
- Move focus infinity
- Move focus zero
- Move focus to absolute position
- Move focus percent
- Print focus position
- Print focus positions
- Print focus steps
- List focus min,max,current
- Focus switch position

Zoom Commands
- Print Zoom position

EEPROM Commands
- EEPROM dump
- Write byte
- Read byte
- Save lens state to EEPROM
- Restore lens state from EEPROM

Control Ring Commands
- Follow Focus
- Follow Aperture

Lens Info Commands Continued

Command: Help (menu)
Syntax: **?**
Returns: Table of commands
Description: Returns table of commands and descriptions
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
?
Canon RF Commands:  '*'=EEPROM
Ver: 1 Rev: 1
ls          Lens status
lc          briRF status: focal len,Ap min,#steps,Ap max
la          Lens attach
ln          Lens name
….
vr          * print version
gs          * get serial number
sn          * print serial number
help or ?   print help
>
```

Command: Print Version
Syntax: **vr**
Returns: Test Version :<number> Rev ::<number>
Description: Returns internal version information from RFLC-USB.
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
vr <LF>
Version :3 Rev :3
>
```

Command: Serial number
Syntax: **sn**
Returns: :<number>
Description: Returns serial number of the RFLC-USB
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
3
>
```

Command: Quiet Mode
Syntax: **qm <0,1>**
Returns: nothing
Description: Sets quiet <1> or normal <0> mode.
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
qm 0
qm 1
>
```

Lens Info Commands Continued

Command: Get Info
Syntax: **ge <#>**
Returns: RFLC internal information
Description: Returns data in the form of a signed integer
Prompt is returned if in normal mode.

Request #'s

```
#define   CLCD_SN                 0
#define   CLCD_LENS_ATTACHED      1
#define   CLCD_VER_MAJOR          2
#define   CLCD_VER_MINOR          3
#define   CLCD_APERTURE_MIN       4
#define   CLCD_APERTURE_CUR       5
#define   CLCD_APERTURE_MAX       6
#define   CLCD_FOCUS_MIN          7
#define   CLCD_FOCUS_CUR          8
#define   CLCD_FOCUS_MAX          9
#define   CLCD_ZOOM_MIN           10
#define   CLCD_ZOOM_CUR           11
#define   CLCD_ZOOM_MAX           12
#define   CLCD_MF_ON              13
#define   CLCD_IS_ON              14
#define   CLCD_LENS_ID            15
#define   CLCD_FNUM_MIN           16
#define   CLCD_FNUM_CUR           17
#define   CLCD_FNUM_MAX           18
#define   CLCD_EXTENDED_DATA      19
#define   CLCD_STATUS             20

#define   CLCD_LENSSN             21
#define   CLCD_SN0                30
#define   CLCD_SN1                31
#define   CLCD_SN2                32
#define   CLCD_SN3                33
#define   CLCD_SN4                34

#define   CLCD_IS_STATUS          40
#define   CLCD_IS_LENS            41
#define   CLCD_CRING              42
#define   CLCD_CRINGFIN           43

#define   CLCD_CRINGMODE          44
#define   CLCD_WIFIMODE           45

#define   CLCD_STATUSCHG          50
```

Example:

```
>ge 16
18
>
```

Lens Info Commands Continued

Command: Lens status
Syntax: **ls**
Returns: Table of lens status values
Description: Returns all lens data in table format
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
ls <LF>
Lens Name             : RF 50mm f/1.8 II
Prime Lens            : 50mm
Aperture min          : F18
Aperture max          : F226
Aperture curr         : F32
Aperture motor steps  : 58
Focus steps           : 984
Focus Position        : 50
>
```

Command: Lens attach
Syntax: **la**
Returns: nothing
Description: Moves lens focus to find endpoints, sets focus to infinity
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
la <LF>
>
```

Command: Lens name
Syntax: **ln**
Returns: <string>
Description: Returns lens name if in internal data base
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
ln <LF>
RF 50mm f/1.8 II
>
```

Command: Lens internal status register
Syntax: **st**
Returns: <string>
Description: AF/MF=Auto/Manual Focus, IS=Image Stabilizer On
F@Stop/FAcell/FMoving/F@Rest = Focus Motor status
A-MotorOn, A@FullOpen = Aperture status
Example:

```
>st <LF>
0x10:AF+F@Stop+F@Rest+A@FullOpen
>
```

Aperture Commands

Command: Initialize aperture
Syntax: **in**
Returns: nothing
Description: Initializes aperture motor and move aperture fully open
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
in <LF>
>
```

Command: Print aperture info
Syntax: **da**
Returns: <string>
Description: Returns lens aperture min, max, and current settings
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
da <LF>
Aperture min          : F18
Aperture max          : F226
Aperture curr         : F18
>
```

Command: Print aperture position
Syntax: **pa**
Returns: <string>
Description: Returns lens aperture current - current stop and F#
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
pa <LF>
0,f18
>
```

Command: Move aperture absolute 1/256 stop
Syntax: **ma <stop>**
Returns: <stop>,f<number>
Description: Moves aperture to absolute position in 1/256 stop's
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
ma 22 <LF>
22,f47
>
```

Aperture Commands Continued

Command: Move aperture fully closed
Syntax: **mc**
Returns: <stop>,f<number>
Description: Moves aperture fully closed
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
mc <LF>
58,f226
>
```

Command: Move aperture fully open
Syntax: **mo**
Returns: <string>
Description: <stop>,f<number>
Description: Moves aperture fully open.
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
mo <LF>
0,f18
>
```

Command: Move aperture incremental 1/256 stop
Syntax: **mn <stops>**
Returns: <string>
Description: Returns lens aperture min, max, and current settings
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
mn -4 <LF>
14,f33
>
```

Command: Move aperture to F-stop #
Syntax: **ms <fstop>**
Returns: <stop>,f<number>
Description: Moves aperture to absolute F-stop. **fstop** is 10x value
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example: Move to f-stop 2.2

```
ms 22 <LF>
4,f21
>
```

Example: Move to f-stop 11.0

```
ms 110 <LF>
41,f108
>
```

Focus Commands

Command: Move focus infinity
Syntax: **mi**
Returns: nothing
Description: Moves focus position to infinity focus
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
mi <LF>
>
```

Command: Move focus zero
Syntax: **mz**
Returns: nothing
Description: Moves focus position to zero focus
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
mz <LF>
>
```

Command: Move focus to absolute position
Syntax: **fa <position>**
Returns: nothing
Description: Moves focus position to absolute position
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
fa 333 <LF>
>
```

Command: Move focus incremental
Syntax: **mf <delta position>**
Returns: nothing
Description: Moves focus position incrementally from current position
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Negative numbers moves focus towards zero focus.
Positive numbers moves focus towards infinity focus.
Focus motors will stop at end points.
Example:

```
mf -55 <LF>
>
```

Focus Commands Continued

Command: Move focus percent
Syntax: **fc <percent>**
Returns: postion:focus steps
Description: Moves focus position to a percent of full range
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Percent is 0.0 to 100.0
Example:

```
fc 44.4 <LF>
44.40:402
>
```

Command: Print focus position
Syntax: **pf**
Returns: focus step position
Description: Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
pf <LF>
511
>
```

Command: Print focus positions
Syntax: **fp**
Returns: Focus motor positions
Description: Prints focus positions in motor value
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
fp <LF>
Fmin:-913 Fmax:251 current:-402
>
```

Command: Print encoder positions
Syntax: **ep**
Returns: Focus encoder positions (NOTE: Not motor step position)
Description: Prints focus positions in encoder value
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Works only in extended data mode
Example:

```
ep <LF>
>EZero:19458 Einf:608 current:11568
```

Command: Print encoder positions
Syntax: **cm**
Returns: Focus in cm (NOTE: Not motor step position)
Description: Prints focus positions in centimeters.
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Works only in extended data mode
Example:

```
cm <LF>
>Fcm:41
```

Focus Commands Continued

Command: List focus brief
Syntax: **lf**
Returns: <min>,<max>,<current>
Description: Prints focus values in simple format.
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
lf <LF>
-913,251,-402
>
```

Command: Focus switch position
Syntax: **fm**
Returns: “AF” - Auto focus or “MF” - Manual focus
Description: Firmware 3.12 or greater. Prints the state of the focus switch on the lens. Starting with firmware 3.12 focus commands will not function if in manual focus mode.
Example:

```
fm <LF>
AF
>
```

Note:

If the lens switch is set to manual focus (MF), the lens controller will silently ignore all focus commands. It also will not calibrate the focus endpoints of the lens when connected in manual focus mode.

Switching back to auto focus will allow the controller to adjust the focus motor again. When switching back to auto focus, it is important to run the “la” command for the lens controller to calibrate the focus endpoints. This will move the focus to each endpoint and loose the previous focus position.

Auto focus does not mean the lens controller will focus on it’s own, it simply allows the lens controller to send lens focus commands.

Zoom Commands

Command: Print Zoom position
Syntax: **pz**
Returns: <value>,<value>,<value>
Description: Prints Zoom position : min, max, current (Lens is prime if all are equal)
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
pz <LF>
50mm,50mm,50mm
>
```

EEPROM Commands

Command: Byte Write
Syntax: **bw <address> <data>**
Returns: nothing
Description: Writes byte to EEPROM, all values are decimal
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
bw 200, 23 <LF>
>
```

Command: Byte Read
Syntax: **br <address>**
Returns: <value>
Description: Reads byte from EEPROM, all values are decimal
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
br 200 <LF>
23
>
```

Command: EEPROM dump
Syntax: **ed**
Returns: <value>
Description: Reads all bytes from EEPROM (Output is in hexadecimal)
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
ed <LF>
EEPROM (HEX address and data):
 0:   1  FF   0   0   0   0   0   0  FF FF FF FF   D   0 1D
10:   3   0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
20:   AF 1   98   3 98   3 D8   3 FF FF FF FF FF FF FF FF
...
3E0:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3F0:  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
>
```

EEPROM Commands Continued

Command: EEPROM Save lens state
Syntax: **es**
Returns: string
Description: Saves lens state to EEPROM, lens state is restored on power up
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
es <LF>
Lens state saved
>
```

Command: EEPROM Restore lens state
Syntax: **er**
Returns: aperture position <LF> string
Description: Restores lens state from EEPROM (Aperture and Focus)
Prompt is returned if in normal mode. Nothing retuned in quiet mode.
Example:

```
er <LF>
58,f226
Lens state restored
>
```

```
//  Reserved EEPROM Locations
#define             EE_LENSTYPE         0x0000
#define             EE_LENSAPRANGE      0x0001
#define             EE_QUITEMODE        0x0002
#define             EE_APWREN           0x0008
#define             EE_I2CN             0x0009
#define             EE_FOCUS_H          0x000A
#define             EE_FOCUS_L          0x000B
#define             EE_APERTURE_H       0x000C
#define             EE_APERTURE_L       0x000D
#define             EE_LENSID_H         0x000E
#define             EE_LENSID_L         0x000F
#define             EE_LENS_SERIAL      0x0010
```

Control Ring Command

Command: Set Control Ring operation
Syntax: **cr <0/1>**
Returns: string
Description: Slaves Focus or Aperture to the lens control ring
Example:

```
cr <LF>
1 Focus
>
```

Settings: 0 = No follow, 1 = Focus follow, 2 = Aperture follow

```
cr 0 <LF>
>

cr 1 <LF>
>

cr 2 <LF>
>
```

WIFI Commands

Command: Enable/Disable WiFi
Syntax: **wi <0/1>**
Returns: string
Description: Enables/Disables WiFi radio
Example:

```
wi <LF>
1 Wifi On
>
```

Settings: 0 = Wifi Off, 1 = Wifi On

```
wi 0 <LF>
> Restart to disable Wifi

cr 1 <LF>
> Restart to enable Wifi
```

Command: Get/Set WiFi SSID
Syntax: **ws <optional SSID>**
Returns: string
Description: ws with no argument returns the current SSID. With argument sets the lens controllers SSID. Up to 32 characters, spaces allowed.
Example:

```
>ws
illunis RFLC 34B7DA5B05A8
>ws Lens Controller
SSID stored
Restart to take effect (Use rst command)
>
```

WIFI Commands Continued

Command: Get/Set WiFi Password
Syntax: **wp <optional password>**
Returns: string
Description: wp with no argument returns the current WiFi password. With argument sets the lens controllers WiFi password. 8-32 characters, spaces allowed. Default is no password.
Example:

```
>wp test1234
Password stored
Restart to take effect (Use rst command)
>wp
test1234
>
```

Command: Reset WiFi SSID and password to default.
Syntax: **wr**
Returns: string
Description: Default SSID is “illunis RFLC” plus MAC address of device. Default password is blank.
Example:

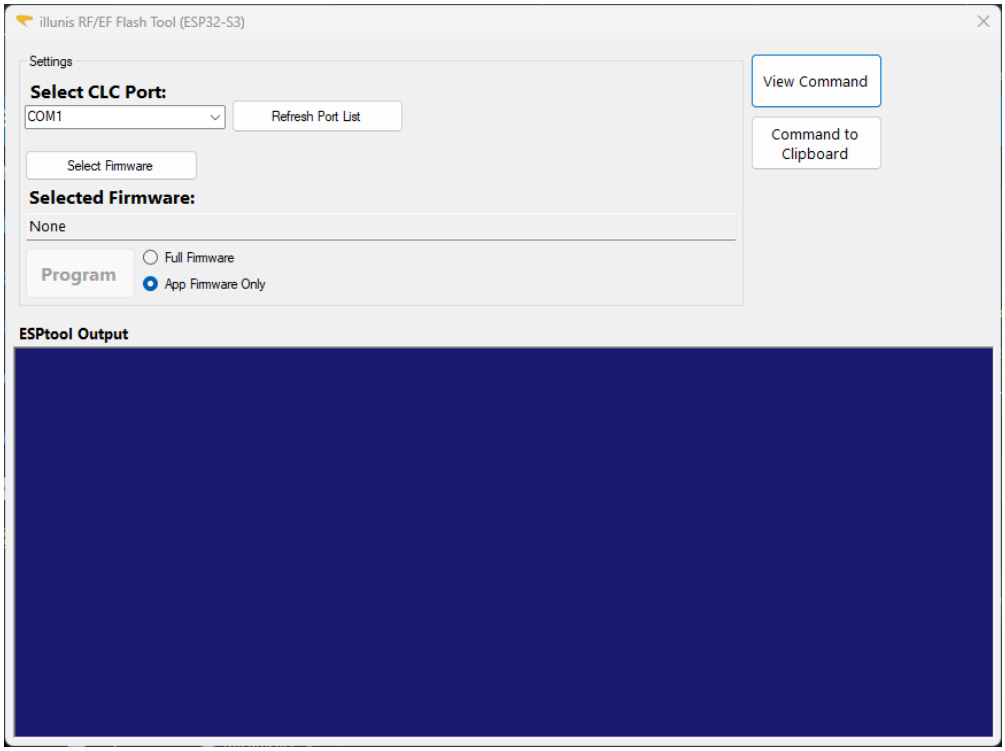
```
>wr
SSID: illunis RFLC 34B7DA5B05A8
Password:
Restart to take effect (Use rst command)
>
```

Command: Read hardware MAC address
Syntax: **gm**
Returns: string
Description: MAC address of the microprocessor. This is a unique number that also functions as the microcontrollers serial number.
Example:

```
>gm
34B7DA5B05A8
>
```

Firmware Update

Firmware Update - illunis ESP32-S3 Flash Tool



1. Select the port for the lens controller
2. Select the firmware provided by illunis.
3. Click **Program** (App Firmware Only)
4. The app will inform you when it's complete or if there are any errors.

The app uses the included ESP tool to perform the update. The command sent to the tool can be copied using the buttons on the right and programed using the cmd.exe if desired.

Firmware Update Troubleshooting

Port open error

Ensure you're connecting to the correct port for the lens controller. Also check that all other connections to the port are closed, such as serial terminal programs.

Port opens but fails to program

The microcontroller may not be able to enter the bootloader without a physical button press on the lens controller. Contact illunis for details on how to upload firmware and correct this issue.

Example Command menu

```
RFLC Commands:'*'=EEPROM        Ver: 10.1.5
---------------------------------------
la         Lens attach
ls         Lens status
ln         Lens name
sl         Lens serial number if available
is         IS status
lc         brief status: focal len,Ap:min,#steps,max
ge <#>     Get Info Cmd
qm <0/1>   * Quiet Mode
---------------------------------------
in         Initialize and open aperture
mc         Move aperture fully closed
mo         Move aperture fully open
ma <stop>  Move aperture abs. 1/256stop
mn <pos>   Move aperture inc. 1/256stop
ms <FX10>  Move aperture to f-stop (10X 22=F2.2)
ad         Print aperture info. brief
da         Print aperture info.
pa         Print aperture position
ga         Get aperture position from lens
---------------------------------------
mi         Move focus infinity
mz         Move focus zero
mf <inc>   Move focus incremental
fa <pos>   Move focus to abs pos.
fc <pos>   Move focus percent
pf         Print focus position
fp         Print focus positions
f#         Print focus #'s
ep         Print encoder positions
cm         Print focus in cm
fm         Print focus switch position
lf         focus min,max,cur
pz         Print zoom position
---------------------------------------
bw a d     * EEPROM byte write decimal
br a       * EEPROM byte read decimal
ed         * EEPROM dump in HEX
es         * EEPROM save lens state
er         * EEPROM restore lens state
cr         * Control ring mode: 1 = Focus, 2 = Aperture
wi         * wifi mode: 1 = On
ws         get/set wifi SSID (32 char max)
wp         get/set wifi password (8-32 char)
wr         reset wifi SSID and clear password
gm         print ESP32 MAC address
rst        Reset
vr         print version
sn         print CLC serial number
?          print help
```

Example Lens Continued

Lens status ‘ls’

```
 Lens Name (From Lens): EF85mm f/1.8 USM
 Prime Lens           : 85mm          <- Lens is prime (not zoom)
 Aperture min         : F18
 Aperture max         : F226
 Aperture curr        : F18           <- Fully open
 Aperture motor steps : 58
 Focus steps          : 1675
 Focus Position       : 1675          <- Focus @ infinity
>
```

Move closed ‘mc’ and Lens status ‘ls’

```
 Lens Name (From Lens): EF85mm f/1.8 USM
 Prime Lens           : 85mm
 Aperture min         : F18
 Aperture max         : F226
 Aperture curr        : F226          <- Aperture reports closed ~ 22
 Aperture motor steps : 58
 Focus steps          : 1677
 Focus Position       : 1677
>
```

Move closed ‘mz’ and Lens status ‘ls’

```
 Lens Name (From Lens): EF85mm f/1.8 USM
 Prime Lens           : 85mm
 Aperture min         : F18
 Aperture max         : F226
 Aperture curr        : F18
 Aperture motor steps : 58
 Focus steps          : 1678
 Focus Position       : 0             <- Focus reports at zero location
>
```

Version ‘vr’

```
Version :3 Rev :6
```

The Canon 100mm RF 80-200mm f/4.5-5.6 USM is shown in this example

```
>
 Lens Name            : RF 80-200mm f/4.5-5.6 USM
 Zoom Lens min/max/cur: 80mm/200mm/195mm  <- Zoom location
 Aperture min         : F56
 Aperture max         : F281
 Aperture curr        : F56
 Aperture motor steps : 37
 Focus steps          : 9
 Focus Position       : 0
>
```

.dll Commands

The iSDK is a .NET .dll provided to aid in connecting to and controlling the lens. It can be found in the Help Center at illunis.com. The following commands are supported.

Note: iSDK 9.2.1.1 or above required for Teledyne Genie cameras.

COM Port Commands

Function: `int PortOpen(string name)`
Returns: 1 for success –1 for failure
Description: Initialize the COM
Example:

```
Int err = initPort("COM4");
```

Function: `void PortClose()`
Returns: void
Description: Close the COM port connection
Example:

```
PortClose();
```

Focus Commands

Function: `void FocusCalibrationOnConnect(bool Enable)`
Returns: void
Description: On some lens models the encoder range can change each time it's attached. This setting must be set before **PortOpen(string name)** or the focus will calibrate. **True** - Moves the lens to 0 and infinity when attached to calibrate controller to the current encoder range. **False** - does not calibrate focus on attach, can be used on lenses with stable encoders. **Default is False.**
Example:

```
FocusCalibrationOnConnect(false);
```

Function: `int GetFocusNear()`
Returns: Near Focus value
Description: Returns the Near Focus Value set by initFocus()
Example:

```
Int FocusNear = GetFocusNear();
```

Function: `int GetFocusFar()`
Returns: Far Focus value
Description: Returns the Far Focus Value set by initFocus()
Example:

```
Int FocusNear = GetFocusFar();
```

.dll Commands Continued

Focus Commands — Continued

Function: `int SetFocusAbsolute(int focus)`
Returns: 1 for success –1 for failure
Description: Sets focus to absolute position between FocusNear and FocusFar
Example: `SetFocusAbsolute( 240 );`

Function: `int GetCurrentFocus()`
Returns: Current focus value
Description: Returns the current focus value
Example: `Int Focus = GetCurrentFocus();`

Function: `int SetFocusInfinity()`
Returns: 1 for success –1 for failure
Description: Sets focus to farthest position
Example:
`SetFocusInfinity();`

Function: `int SetFocusZero()`
Returns: 1 for success –1 for failure
Description: Sets focus to nearest position
Example:
`SetFocusZero();`

Function: `string SaveFocusState();`
Returns: **!** For success **?** For error
Description: Saves current focus position to EEPROM
Example: `String success = SaveFocusState();`

Function: `string RestoreFocusSatate()`
Returns: **!** For success **?** For error
Description: Performs a focus calibration, then sets focus to value saved in EEPROM. Note: This will return after command is sent, but lens may take up to 4 seconds to complete.

Iris Commands

Function: `double GetIrisMin()`
Returns: Minimum Iris Value (Most Open value)
Description: Returns the Minimum Iris Value.
Example: `double MinIris = GetIrisMin();`

Function: `double GetIrisMax()`
Returns: Maximum Iris Value (Most Closed value)
Description: Returns the Maximum Iris Value.
Example: `double MaxIris = GetIrisMax();`

.dll Commands Continued

Iris Commands — Continued

Function: `int SetIrisAbsolute(double focus)`
Returns: 1 for success –1 for failure
Description: Sets focus to absolute position between GetIrisMin and GetIrisMax
Example:
```
SetIrisAbsolute( 1.8 );
```

Function: `int SetIrisStepAbsolute(int steps)`
Returns: 1 for success –1 for failure
Description: Sets focus to step position between 0 and GetIrisSteps
Example:
```
SetIrisStepAbsolute( 20 );
```

Function: `double GetIrisCurrent()`
Returns: Current Iris value
Description: Returns the current iris value
Example:
```
double Iris = GetIrisCurrent();
```

Function: `int GetIrisCurrentStep()`
Returns: Current Iris step value
Description: Returns the current iris value in steps
Example:
```
int IrisStep = GetIrisCurrentStep();
```

Function: `int GetIrisSteps()`
Returns: number of steps from fully open to fully closed iris
Description: Returns total stepper motor step for iris
Example:
```
int steps = GetIrisSteps();
```

Function: `int SetIrisIncremental(int stops)`
Returns: 1 for success –1 for failure
Description: Moves iris stepper motor number of stops. Can be positive or negative.
Example:
```
SetIrisIncremental(1);
SetIrisIncremental(-2);
```

Function: `int SetIrisOpen()`
Returns: 1 for success –1 for failure
Description: Fully opens iris
Example:
```
SetIrisOpen();
```

Function: `int SetIrisClosed()`
Returns: 1 for success –1 for failure
Description: Fully closes iris
Example:
```
SetIrisClosed();
```

.dll Commands Continued

General Commands

Function: `string GetLensName()`
Returns: Lens Name
Description: Returns Lens Name
Example:
```
String Name = GetLensName();
```

Function: `string GetLensStatus()`
Returns: Table of Lens parameters
Description: Returns Lens parameters
Example:
```
String Status = GetLensStatus();
```

Function: `string GetVersion()`
Returns: Lens Controller firmware version
Description: Returns Lens Controller firmware version
Example:
```
String Version = GetVersion();
```

Function: `void LensHeartbeat(bool Enable)`
Returns: void
Description: Enables or disables SDK periodic lens presence checks to raise LensPresenceChanged event.
Default: true
Example:
```
LensHeartbeat(false);
```

Function: `event EventHandler LensPresenceChanged`
Returns: none
Description: Event is raised when a lens is attached or detached from the controller if LensHeartbeat is set to true.
Example:
```
myLens.LensPresenceChanged += LensAttachDetach;
```

Function: `bool LensPresent()`
Returns: True if lens attached to controller
Description: Returns current lens status updated by LensHeartbeat. If LensHeartbeat is disabled it will query lens controller.
Example:
```
bool LensPresent = LensPresent();
```

.dll Commands Continued

General Commands — Continued

Function: `string PortWrite(string command)`
Returns: returns lens controller response to command (if any)
Description: Used to send any command covered earlier in the guide that does not have a SDK function. Returns lens controller response, "!" for success on commands with no response, "?" for failed or unknown command.
Example:

```
string Response = PortWrite("pz");
```

Function: `void SerialLogPath(string logfile)`
Returns: void
Description: Set to a full path and file name to log commands sent to lens controller and responses received. Text will be appended if the file exists or the file will be created if it doesn't exist. If the file can not be opened or created logging will be disabled, to reenable send the SerialLogPath command with a new path/file.
Example:

```
SerialLogPath("D:\Documents\COMlog.txt");
```

Variables (iSDK 9.4.4.4 or higher)

Variable: bool LensConnected
Values: true - lens connected; false - no lens
Description: Reports if a lens is connected

Variable: decimal FirmwareVersion
Values: reports firmware version as a decimal.
Description: C# only, this is not COM visible.
Example: Version 3 Rev 12 would be 3.12

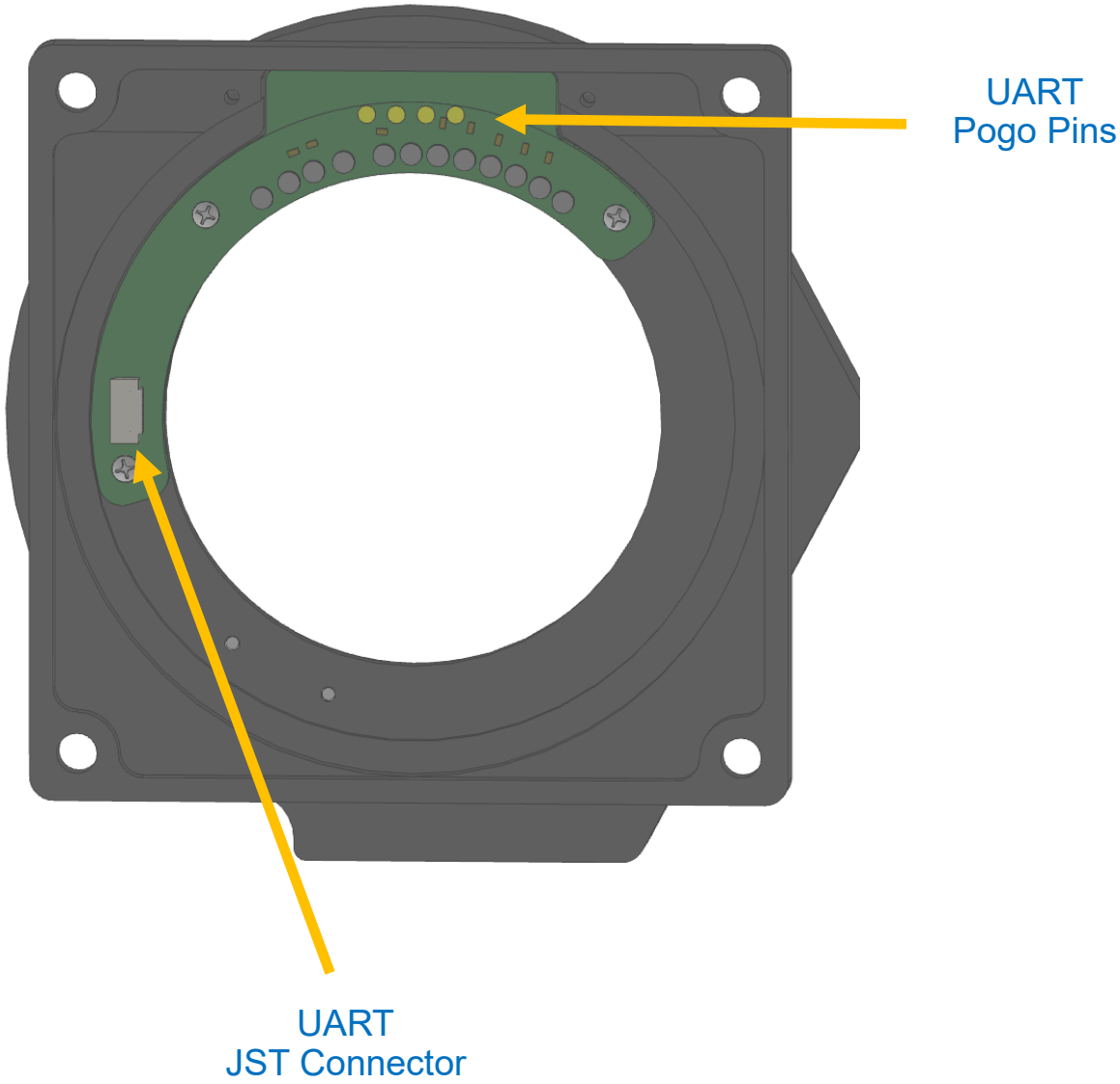
Variable: bool PortConnected
Values: true - port open ; false - port not opened
Description: Reports if the com port is open

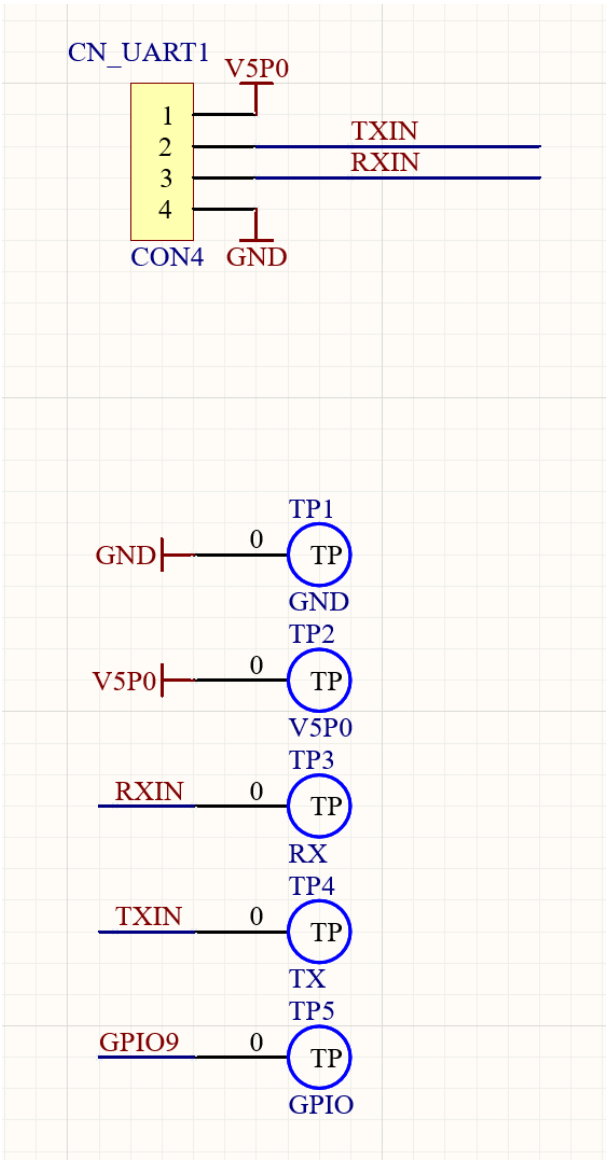
Variable: bool AutoFocusMode
Values: true - lens switch is set to AF; false - lens switch is set to MF
Description: Reports lens switch status.

RFLC Hardware communication interfaces

The RFLC connections for embedded systems are available on two connectors. The UART connection can be accessed through either Pogo Pin pads or a JST SUR connector. There are 4 connections, Power (+5V), Ground, Transmit and Receive.

The RFCL must be pre configured for UART operation. Call for details.

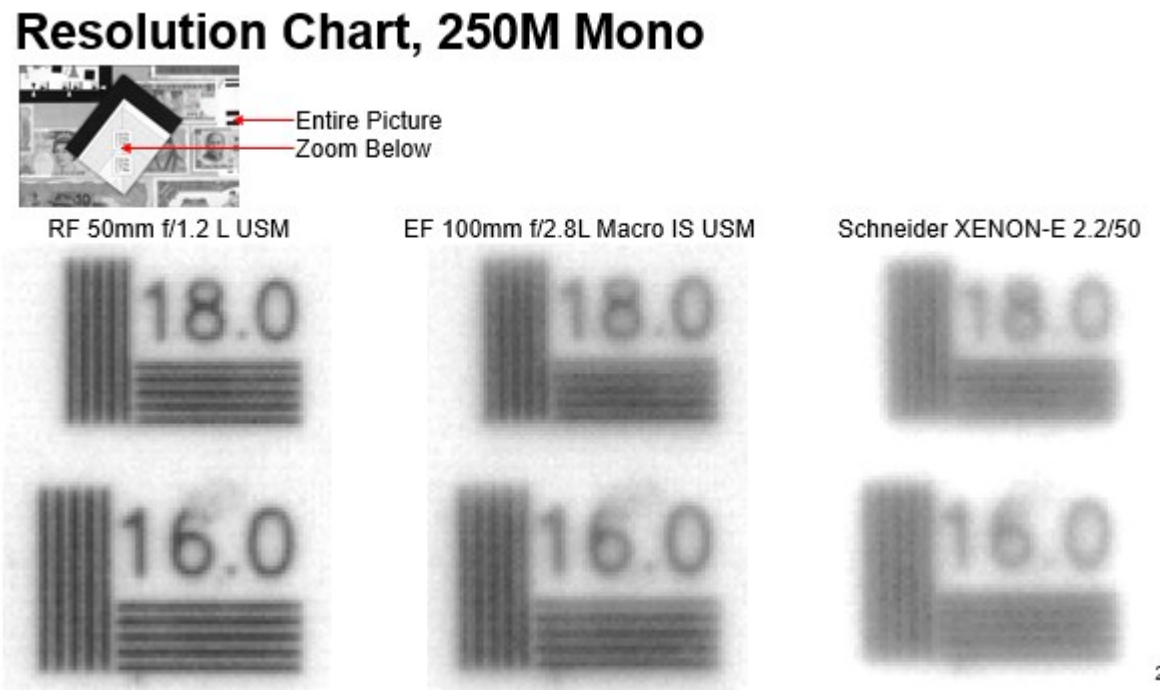


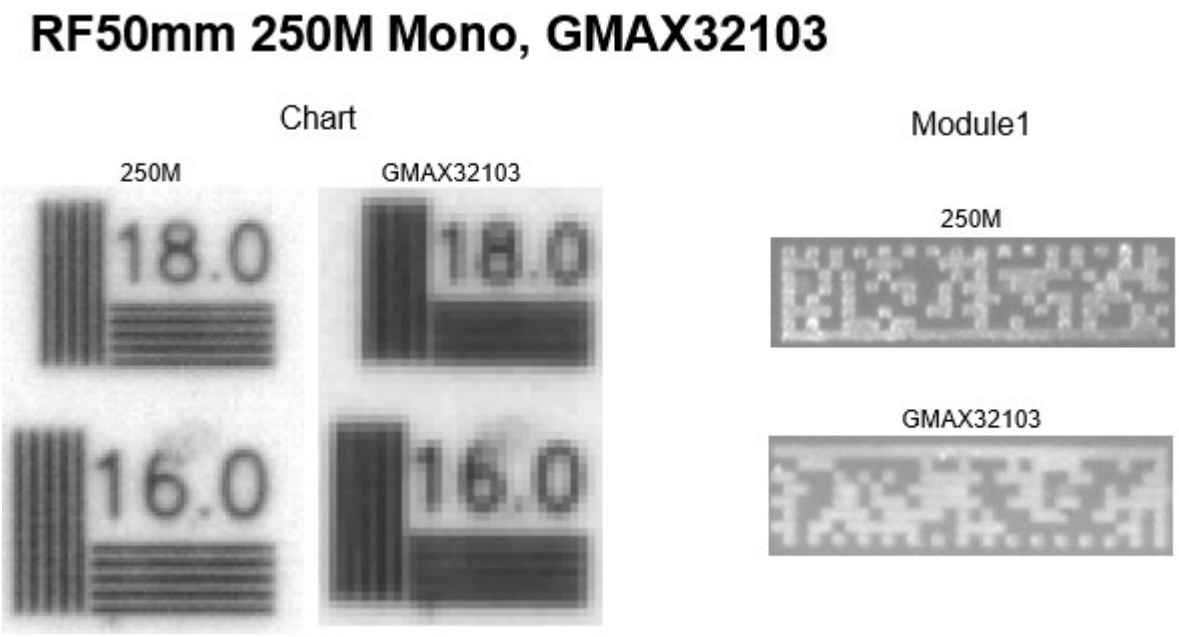


RF Lens performance

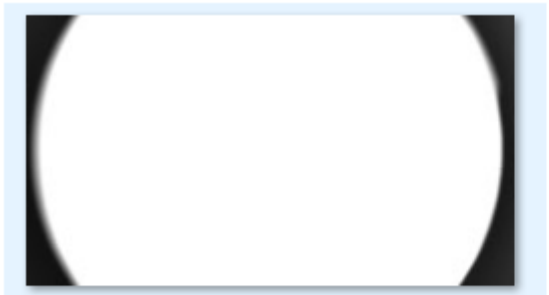
RF prime lenses were tested and the following results are presented.

Tests were made with the illunis EMC-250M which has a 1.5um pixel. This sensor has traditionally been very difficult to select a lens that can resolve to the pixel level. The RF50mm F1.2 lens at F4 can resolve as shown.





A prime use of the RF lens is the EMC-103 camera with a slightly larger focal plane than the traditional 35mm format that the EF/RF lens are designed for. To determine the RF lens image circle we imaged the lenses with a medium format camera (EMC-152) that has a horizontal dimension of 53.0mm. We then overlayed the 103 sensor onto the image circle to determine lens coverage. The results of the prime lens are:

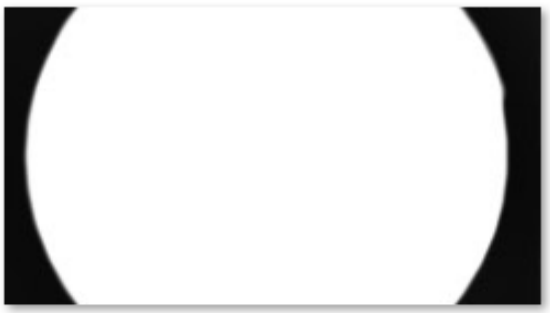


RF50mm F1.8 on EMC-152.tif

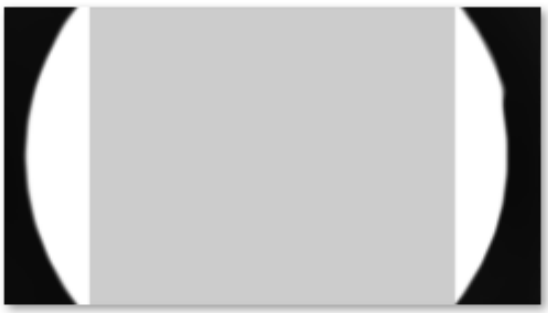
RF50mm F1.8 on EMC-152.jpg

RF85mm F1.2 on EMC-152.tif

RF85mm F1.2 on EMC-152.jpg



RF135mm F1.2 on EMC-152.tif



RF135mm F1.2 on EMC-152.jpg

Lens Dial Controller

The RFLC and EFLC can be controlled remotely with the illunis dial controller. The dial controller (DC) is a Wi-Fi enabled device that communicates with the RF/EF lens controllers. The DC provides a graphical user interface with an easy to use rechargeable hand held device.

Lens Control using Dial Controller

Step 1:

Plug in the dial controller via a USB type C cable to provide power to the RFLC (This will power on the device immediately).

Alternatively, The device can be powered up by pushing down and holding the dial button for a few seconds (Note that the button is located on the encoder where the M5 logo is and can be pushed by pushing down on the case where it says “M5”)

Step 2:

Navigate to the WiFi menu and select it (This can be done by rotating the dials encoder to select the WiFi menu and pushing the “M5” button or by touching the WiFi icon and holding it down)

Step 3:

Push the refresh button on the dials touch screen to find available WiFi networks, then rotate the dials encoder to select the RFLC you wish to connect to (the selected WiFi network will have a red box around its text). You can then connect to the desired network/RFLC device by pushing the dials “M5” button or by pressing the connect button on the touch screen.

Step 4:

If your RFLC has a password for its WiFi login, you will need to input it in the password menu which can be accessed from the WiFi menu by pushing the password button on the touch screen. The password can be entered by turning the dials encoder to select a character and then added by either pushing the enter button or by pushing the dials “M5” button. You can then connect to the desired network/RFLC device by pushing the dials “M5” button or by pressing the connect button on the touch screen.

Once the RFLC is successfully connected to the dial, the network name for the RFLC should be highlighted in green in the WiFi menu.

Step 5:

Exit the WiFi menu by pushing the exit button on the touch screen and navigate to any desired device options using either the dials encoder and the “M5” button or by using the touch screen.

Dial Features:

Auto-Login to the RFLC:

This can be enabled by first connecting the dial controller to an RFLC device and then navigating to the password menu where you want to push the “Save Login” button. You then need to navigate to the WiFi menu and push the “Auto-Login” button which, when enabled, should appear to be a darker shade of grey and say “On”. Your dial controller should then auto connect to the RFLC when the dial is powered on.

Powering and Charging the Dial:

The device can be powered on by pushing down and holding the dials button (the button is pushed by pressing down on the part of the dials case that says “M5”). To power off the device, navigate to the power icon in the dials main menu and either press down the dials button or hold down the touch screen. The dial can be charged using a standard USB Type C cable.

Data and Info Menus:

The Lens Data menu provides information on the attached lenses name, zoom, and the status of the image stabilizer.

The RFLC Info menu provides information regarding the RFLC’s serial number, software version, and its login information (SSID and Password).

Focus and Aperture/Iris Menus:

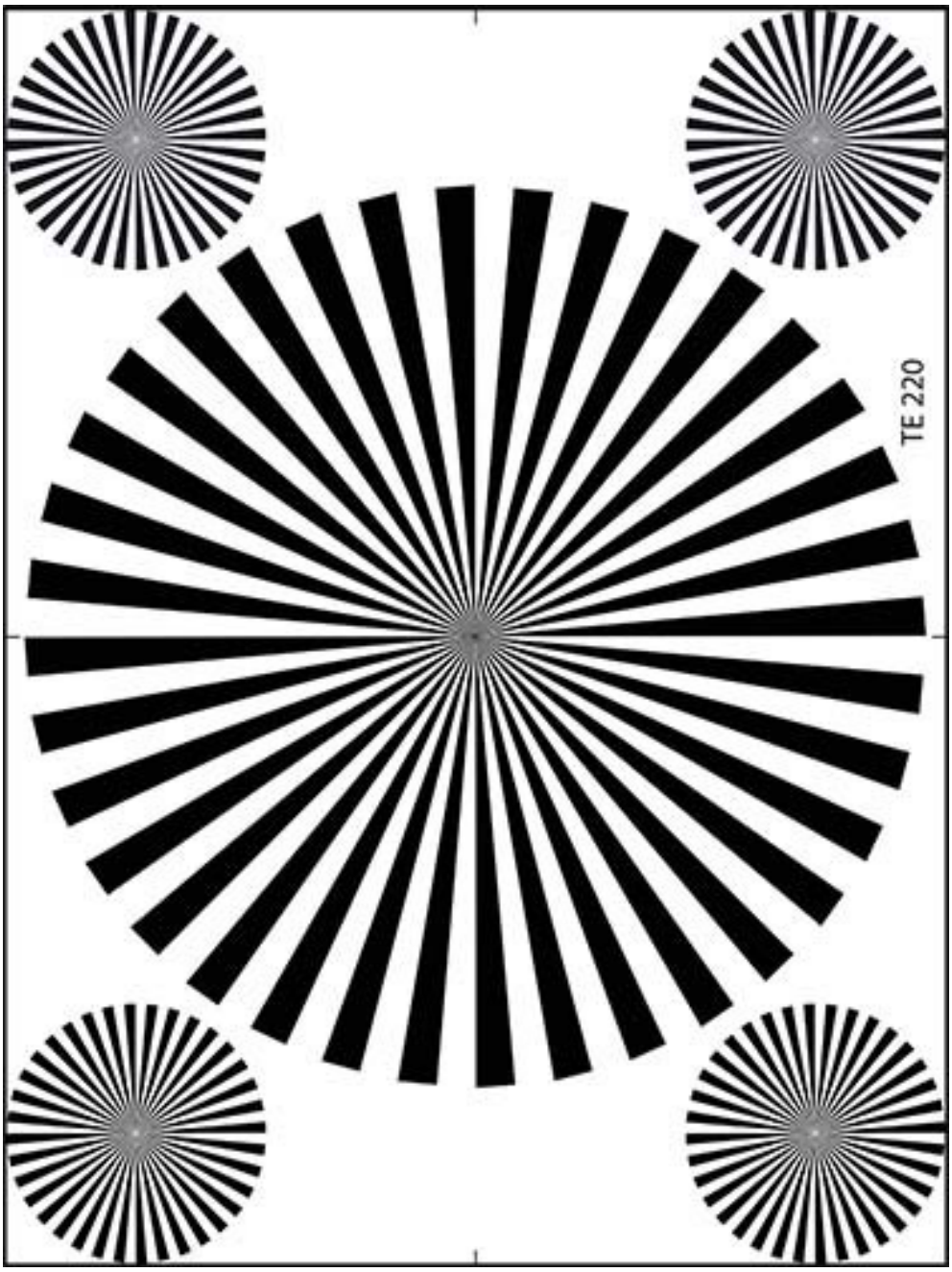
The focus and aperture menus both allow the user to adjust the lenses focus and aperture values respectively. Each menu has a pair of buttons that allows a user to set the lens to its minimum or maximum values (for example, for aperture the buttons are open and close) in addition to a curved slider that can be adjusted either by touching it via the touch screen or by turning the
encoder of the dial

Save/Restore Menu:

The Save/Restore menu is used to save the current state of the lens and to restore the lens to a previously saved state.

RFLC Control Ring Menu:

The control ring menu allows the user to select what the control ring on the RF lens attached to the RFLC does. The control ring can be set to control the aperture or focus of the lens and can also be turned off.



For more information on any illunis product, including detailed specifications and options, please visit our website at **www.illunis.com,** email **info@illunis.com,** or call illunis at the phone number listed below.

illunis LLC
Headquarters
14700 Excelsior Blvd
Minnetonka, MN 55345 USA

Phone: 952.975.9203
FAX: 952.294.8308
www.illunis.com