ISSI LC-PZA2-E

Canon® Ethernet Lens Controller

Application Program Interface

Notice

Contact Information: Innovative Scientific Solutions Inc.

7610 McEwen Road Dayton, OH 45459 Ph.: (937) 630-3012 Fax: (937) 630-3015

Website: www.innssi.com, Email : issi-sales@innssi.com

ISSI LC-PZA2-E API

API based on UDP Protocol. Port 1339/UDP.

1.   Commands

Action: Get controller f/w version

Command: "ver<CR>"

Returns: "*ISSI LCI-EF-232=4.x.y (s/n:19xxx)*"

Action: Initialize lens, and get current motors values and ranges (aperture ranges would change when zoom changed)

Command: "*ping<CR>*"

Returns (in separate UDP packets):

zRange=minZ,maxZ

fRange=minF,maxF

aRange=minA,maxA

Current=ZZ,FF,AA

AF=X

IS=Y

ISactive=S"

where

- minZ– minimum zoom value,
  maxZ – maximum zoom value
- minF– minimum focus value,
  maxF – maximum focus value

- minA– minimum aperture value,
  maxA – maximum aperture value
- ZZ– current zoom value, FF– current focus value, AA - current aperture value
- X = **1** if the focus switch is in *AF* position on lens and **0** if in *MF*.
- Y = **1** if Image Stabilization(IS) switch on the lens is enabled, and **0** if disabled or the lens has no IS
- S = **1** if IS is activated and **0** if it deactivated
- *"nolensfound"* return in case no lens found

Action: Move Focus motor on X units

Command: *"moveFocus=X<CR>"* where X could be as positive as negative values.

Returns: *"Focus=Y"*, where Y is current value of Focus motor or *"errorFocus"* when focus value is not reachable;

Action: Move Aperture motor on X step (one-quarter-stop f-number)

Command: *"moveAper=X<CR>"* where X could be as positive as negative values. **Returns**: *"Iris=Y"*, where Y is current value of Aperture or *"errorAperLimits"* when aperture value is not reachable

Action: Set desired value for Focus motor

Command: *"setFocus=X<CR>"* where X is positive value

Returns (in two UDP packets): *"Focus =XXX"*, where XXX is the current value (for non-USM lens could take more time), and "focusDone" message, *"errorFocus"* - indicate focus positioning problem

Action: Timeout for *fRange/setFocus/setFocus2* for old and slow lens without unreliable feedback from encoder. Timeout will be stored in non-volatile device memory.

Command: "*setFocusT=X<CR>*" where X is a positive value, presents timeout in ms. The default value is 300 ms, available range from 0 to 65000.

Returns: "*OK*"

Action: an alternative way to set lens focus: lens will move the focus motor to minimum position, reset the encoder counter, and then position the focus. Helps to suppress the accumulative lens drift effect for some lenses.

 Command: "*setFocus2=X<CR>*" where X is a positive value

Returns(in two UDP packets): "*Focus =XXX*", where XXX is the current value (for non-USM lens could take more time), and "focusDone",   "*errorFocus*" - indicate focus positioning problem

Action: Set desired value for aperture (in f-number)

Command: "*setAper=X<CR>*" where X is in quarter-stop f-number scale, please use pre-calculated f-stop numbers: *1.0, 1.1, 1.3, 1.4, 1.6, 1.8, 2.0, 2.2, 2.5, 2.8, 3.2, 3.5, 4.0, 4.5, 5.0, 5.6, 6.3, 7.1, 8.0, 9.0, 10, 11, 13, 14, 16, 18, 20, 22, 25, 29, 32, 36, 40, 45, 51, 57, 64, 72 ,80, 90*

Returns: "*Iris=Y*", where Y – current value of Aperture, if X is out of range will answer with "*errorAperLimits*" message

Action: detect IS (Image Stabilization) availability for connected lens

Command: "*isIS<CR>*"

Returns: "*IS=Y*" where Y could be **0** or **1**, "**0**" - means the lens does not have IS function, "**1**" - lens has IS function. Also will answer "**0**" if the IS switch is in OFF position.

Action: activate IS (Image Stabilization) function for connected lens
Command: "*enableIS=X<CR>*"
Returns: "*ISactive=Y*" where X is the number of seconds to keep IS active, from [0-3600]. "**0**" is disable IS, and Y is **0** or **1** is current IS status

Action: Set nickname for LC (stored in LC memory)
Command: "*setName=XXXXXXX<CR>*" where XXXXXXX is a 7-characters name for this controller.
Returns: "*OK*"

Action: Get LC nickname (stored in non-volatile memory)
Command: "*getName<CR>*"
Returns: "*Name=XXXXXX*"

Action: soft restart LC-2
Command: "*Reboot<CR>*"
Returns: "*OK*"

Action: explore focus limits
Command: "*refRange<CR>*"
Returns: "*fRange=minY,maxY*" where minY is minimum focus value, maxY is maximum focus value

Action: get lens name

Command: *"getLens<CR>"*

Returns: *"Lens=XXXXXX"*, where *XXXXXX* is the lens name stored in internal lens memory, the lens might not support this command.

Action: get distance range from internal lens encoder in meters

Command: *"getDist<CR>"*

Returns: *" Dist=XX,YY"*, where *XX and YY means that current lens focus distance is in the range of XX-YY meters*

Action: set focus ring in minimum position

Command: *"goMIN<CR>"*

Returns: *" MIN_OK"*

Action: set focus ring in maximum (infinity) position

Command: *"goMAX<CR>"*

Returns: *" MAX_OK"*

Action: open aperture to the maximum for that lens and current zoom position

Command: *"openAper<CR>"*

Returns: *"OK"*

Action: reset internal encoder counter

Command: *"setZero<CR>"*

Returns: *"OK"*

Action: enable a workaround for Canon EFS 18-135mm lens

Command: "*setProto=X<CR>*", where X is 1 or 0, will be stored in non-volatile device memory.
Returns: "*OK*"

Action: get current status for workaround for Canon EFS 18-135mm lens
Command: "*getProto<CR>*",
Returns: "1" or "0"

Action: set desired lens value for zoom in mm
Command: "*setZoom=X<CR>*", where X is lean focal lengths value in mm.
Returns(in two UDP packets): Zoom=*XXX*", where XXX is the current zoom value and "ZoomDone" message

Action: get current lens zoom value in mm
Command: "*getZoom<CR>*",
Returns: Zoom=*XXX*" where XXX is the zoom length value in mm.

Action: move zoom motor
Command: "*moveZoom=X<CR>*", where X internal zoom steps (for precise move zoom ring), could be negative or positive.
Returns: Zoom=*XXX*", where XXX is the current zoom value

2. Examples

Python

```
----------------------------------------------------------------
import socket
import time


UDP_IP = "192.168.2.251"
UDP_PORT = 1339


counter = 1
print ("ISSI :: Canon Python Script Example 2016-8-25\n")


# LC command sequence
MESSAGES = ["ver", "ping", "moveAper=3","moveAper=-3", "Restart"]


sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # UDP
sock.setblocking(0)
sock.settimeout(2.5)


# main loop
while True:
        print("#%d cycle\n" %counter)
        for i in range(len(MESSAGES)): # LC2 command loop
                print ("Sent:", MESSAGES[i])
                sock.sendto(MESSAGES[i].encode(), (UDP_IP, UDP_PORT))
                try:
                        while True:
                                data, addr = sock.recvfrom(1024)
                                if not data: break
                                print ("Received:", data)
                except  socket.error:
                    print("")
                time.sleep(1.0)
                counter = counter +1
```

```
------------------------------------------------------------------
  Script reply for Canon EF 35mm f/1.4L USM lens
------------------------------------------------------------------
ISSI :: Canon Python Script Example 2016-8-25


#1 cycle


Sent: ver
Received: b'ISSI LC-2=1.8.0 (s/n:19462)'


Sent: ping
Received: b'zRange=35,35'

Received: b'fRange=0,2505'

Received: b'aRange=1.4,22.0'

Received: b'Current=35,2505,1.4'

Received: b'AF=1'

Received: b'IS=0'

Received: b'ISactive=0'


Sent: moveAper=3
Received: b'Iris=2.0'


Sent: moveAper=-3
Received: b'Iris=1.4'


Sent: Restart
Received: b'OK'


[...]
------------------------------------------------------------------
```