## Serial

1. serialThread serialWorker will be moved onto this thread
2. SerialUI QSerialUI from helpers.Qserial\_helper
3. serialWorker QSerial from helpers.Qserial\_helper, separate thread handling serial i/o
4. low level PSerial

Communication between Threads is through Signals only. User Interface does not have access to workers on separate thread and worker thread does not have access to User Interface.

Startup emit scanPortsRequest and setupReceiverRequest to populate the pull down menus

SerialReceiveState:

stopped, awaitingData, receivingData, finshedReceivingData, timedOut

Since we don’t know when receiving serial data is finished we regularly update receiver. If no timeout occurred, and there is still data in the buffer we remain in receiving mode, if there is no data available after the update and we were in receivingData we move to finishedReceivingData. Timeout occurs when state machine is not advancing.

### finished

-serialWorker serialThread quit & deletelater

-serialThread serialThread deletelater

### PSerial

Use python Serial to manage Serial Ports.

The Low Level Functions are:

-scanports

-open

-close

-changeport

-readline

-writeline

-avail

### serialWorker – 2 – QserialUI Signal Connections

Signals Slots

-textReceived on\_SerialReceivedText

-newPortListReady on\_newPortListReady

-newBaudListReady on\_newBaudListReady

-serialStatusReady on\_serialStatusReady

### SerialUI – 2 – serialWorker Signal Connections

Signals Slots

-changePortRequest on\_changePortRequest

-closePortRequest on\_closePortRequest

-changeBaudRequest on\_changeBaudRequest

-scanPortsRequest on\_scanPortsRequest

-scanBaudRatesRequest on\_scanBaudRatesRequest

-setupReceiverRequest on\_setupReceiverRequest

-startReceiverRequest on\_startReceiverRequest

-sendTextRequest on\_sendTextRequest

-sentLinesRequest on\_sendLinesRequest

-serialStatusRequest on\_serialStatusRequest

-finshedWorkerRequest on\_stopWorkerRequest

-startReceiverRequest on\_startReceiverRequest

-likely not used on\_stopReceiverRequest

-receiverTimer.timeout \_updateReceiver

### Serial User Interface Boxes and Elements

UI Elements Slots

-comboBoxDropDown\_SerialPorts index changed to on\_comboBoxDropDown\_SerialPorts

-comboBoxDropDown\_BaudRates index changed to on\_comboBoxDropDown\_Baudrates

-pushButton\_SerialScan clicked to on\_pushButton\_SerialScan

-pushButton\_SerialSend clicked to on\_serialMonitorSend

-lineEdit\_SerialText returnPressed to on\_serialMonitorSend

-pushButton\_SerialClearOutput clicked to on\_pushButton\_SerialClearOutput

-pushButton\_SerialSave clicked to on\_pushButton\_SerialSave

-shortcutUpArrow on\_serialMonitorSend..UpArrow

-shortcutDownArrow on\_serialMonitorSend..DownArr

## LightSource

The lightsource class is communicating between user interface and the serial Worker.

1. lightSourceWorker QLightSource

## QLightSource

Defines the text commands needed to the lightsource controller

Support Functions:

\_setChannelIntensity change intensity

\_manualTurnOnChannel enable channel

\_manualTurnOffChannel disable channel

## lightSourceWorker Signal to serialWorker Slot

Signals Slots

sendTextRequest on\_sendTextRequest

sendLinesRequest on\_sendLinesRequest

startReceiverRequest on\_startReceiverRequest

connectLightSourceRequest connect textReceived signal to on\_ChannelSettings

disconnectLightSourceRequest disconnect textReceived signal to on\_ChannelSettings

## LigthsourceUI to lightSourceWorker

-TurnOff all pushButton\_TurnOnChannel1..13

-Set Max Min horizontalSlider\_Channel1..13

Signals Slots

ui.horizontalSlider.valueChanged on\_IntensitySliderChanged

ui.horizontalSlider.sliderReleased on\_IntensitySliderReleased

ui. lineEdit\_Channel1..13.returnPressed on\_IntensityLineEditChanged

ui.pushButton\_TurnOnChannel.clicked on\_pushButton\_TurnOnChannel

ui.pushButton\_turnOffAllChannels.clicked turnOffAllChannels

ui.pushButton\_EnableAutoAdance setAutoAdvanceOn

ui.pushButton\_DisableAutoadvance setAutoadvanceOff

ui.pushButton\_QueryLightSourceSettings queryChannelSettings

ui.pushButton\_SaveLightSourceSettings storeChannelSettings

ui.pushButton\_LoadLightSourceSettings loadChannelSettings

ui.checkbox\_MeasureChannel1..13.stateChanged on\_enableChannel

## Camera

How should this work?

1. CameraThread cameraWorker will be moved onto this thread
2. CameraUI QCameraUI from helpers.Qcamera\_helper
3. cameraWorker QCamera from helpers.Qcamera\_helper, separate thread handling camera i/o
4. low level BalckflyCapture and OpenCVCapture

Thread will be create in main program

cameraWorker opens lowlevel camera object, initializes it and then runs update which will be continuously running.

QCamera will need to have self.cam = BackflyCapture() or OpenCVCapture()

CameraUI will need to signal to QCamera which type of Camera is wanted when user selects

cameraType opencv, blackfly, nano, rtp, rtsp, libcam, pi

### QCameraUI

-init scene,graphicsView,pixmap

-on\_FPSready: displays fps

-on\_ImageDataReady: displays image in pixmap object of ui.

-on\_Start:

1) setup measured channels

2) setup analysis pipeline

3) setup display pipeline

4) emit start

-on\_Stop

1) TBD

-measuredChannels: check which channels are selected for measurement

NEED TO DEVELOP:

When users changes selected channels (ui.checkBox\_MeasureChannel1..13.clicked), stop camera, make sure it triggers light source commands, restart camera if camera was running.

### QCamera

This will need to run on separate Thread

Signal

imageDataReady

cameraStatusReady

finished

Will need to capture dataCubeReady signal

-init: setsup BlackFlyCapture or OpenCVCapture and opens camera

-on\_startCamera

1) create datacube structure, we will need to pass to camera

2) camera.startAcquistion

3) camera.update

-on\_stopCamera

1) camera.stopAcquistion

### BlackflyCapture

Signals

imageDataRead np.ndarray

fpsRead float

-init

-update: loop continuously

1) get next image from camera

2) add to datacube should simply emit imageDataready

-openCamera: opens camera interface, applies settings

-closeCamera

-startAcquistion

-stopAcquisition

Features to read and set

width, height, resolution, offset, binning, exposure, autoexposure, fps, adc (8,10,12,14bit), pixelformat (Mono8,Mono10,Mono12, Monot16…), ttlinv, trigout (want tigger on line 1 or 2), trigin (on line 0,2,3 or software)

### OpenCVCapture

imageDataRead np.ndarray

fpsRead float

-init

-update: loop continuously, converts BGR to gray scale

1) get next image from camera

2) emits imageDataReady

-openCamera: opens camera interface, applies settings

-closeCamera

-startAcquistion

-stopAcquisition

-opensettings: opens user interface for operating system camera subsystem

Features to read and set

width, height, resolution, exposure, autoexposure, fps, fourcc, buffersize, gain, wbtemperature, autowb

## Main User Interface

### Camera

Start: ui.pushButton\_CAMERASTART

Stop: ui.pushButton\_CAMERASTOP

Calibrate: ui.pushButton\_CALIBRATE

Measure: ui.checkBox\_MeasureChannel1..13

Exposure: ui.lineEdit\_CameraExposureTime

ui.lineEdit\_CameraFrameRate

### Display

-Display Channel:

ui.checkBox\_DisplayBackground

ui. checkBox\_DisplayChannel1..13

-Display Computed

Analysis: ui.checkBox\_DisplayAnalysis

ui.comboBox\_FirstChannel

ui.comboBox\_MathOperation

ui.comboBox\_SecondChannel

Physio: ui.checkBox\_DisplayPhysio

ui.checkBox\_EnableCalcHb

ui.checkBox\_EnableCalcHbO2

ui.checkBox\_EnableCalcMelanin

ui.checkBox\_EnableCalcBilirubin

ui.checkBox\_EnableCalcWater

ui.checkBox\_EnableCalcFat

ui.checkBox\_EnableCalcOther

ui.checkBox\_EnableCalcScat

ui.checkBox\_EnableCalcR

ui.checkBox\_EnableCalcMie

Color: ui.checkBox\_DisplayColor

ui.comboBox\_SelectRedChannel

ui.comboBox\_SelectGreenChannel

ui.comboBox\_SelectBlueChannel

Spectrum: ui.checkBox\_DisplaySpectrum

-FPS

ui.lcdNumber\_FPSIN

ui.lcdNumber\_FPSOUT

### Processing

Subtract Background

ui.checkBox\_SubtractBackground

Flatfield Correction

ui.checkBox\_ApplyFlatFieldCorrection

Binning

ui.checkBox\_ApplyBinning

ui.comboBox\_SelectBinning

Temporal Filter

ui.checkBox\_ApplyTemporalFilter

ui.horizontalSlider\_LowFrequency

ui.horizontalSlider\_MiddleFrequency

ui.horizontalSlider\_HighFrequency

ui.lineEdit\_LowCutOff

ui.lineEdit\_HighCutOff

Save to File

ui.checkBox\_EnableSaving2File

Save to RAM

ui.checkBox\_EnableSaving2RAM

ui.lineEdit\_Filename

## Processing

### QDataCube

-width, height, depth,

-data: the data cube,

-bg: single bg image,

-flat: datacube with flatfields,

-inten: average intensity in each image,

-data\_index: pointer to current image plane

add(image): adds image and increments data\_indx

sort((dx,dy)): first image in cube becomes image with lowest intensity

cube2DisplayImage()

bgflat8(data,bg,flat): subtracts bg and multiplies with flat, (8bit input data, 16 bit output)

bgflat16 (16bit input data, 32bit output data)

bin2,3,4,5,6,9,10,12,15,18,20

### QDataDisplay

-displaytrans: converts data to better dynamic range

-resizeimage

### threeBandEqualizerProcessor

-equalize

poormansHighpassProcessor

runningsumHighpassProcessor