Software Specification

Revision 6.0.3c

August 22, 2012

QSICamera

OS X Software for Quantum Scientific Imaging, Inc. 500/600-series CCD Cameras

Joe Shimkus

Copyright © 2012, Joe Shimkus

joe@shimkus.com

Copyright © 2012, Joe Shimkus

Copyright © 2012, Joe Shimkus

License

Copyright (c) 2012, Joe Shimkus

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- The name of Joe Shimkus may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2012, Joe Shimkus

Change Log

Date	Author	Change
August 22, 2012	Joe Shimkus	• Changed all client-facing C data structures to Objective-C classes. This was driven by the realization that one of the data structures contained object references which resulted in QSICamera being incompatible with ARC-based software. APIs that previously took a pointer to one of the C data structures for returning information now take a pointer to an object pointer of the appropriate class and return an auto-released instance of the class. With these changes QSICamera should seamlessly operate with manual reference counting, garbage-collected and ARC-based software. • Added the QSICamera class method `simulatedCameras`. This method returns an array of cameras which, while not allowing soup-to-nuts testing of QSICamera, allows testing of its higher-level functionality. More usefully, these simulated cameras allow the development of applications incorporating QSICamera without requiring the actual connection of a camera from QSI. Of course, you will want to test anything you develop against real cameras.

Copyright © 2012, Joe Shimkus

Date	Author	Change
August 15, 2012	Joe Shimkus	Corrected issues identified via running Xcode's static analysis tool. Added a new class (QSICameraDefaults) interfacing with the Cocoa NSUserDefaults system. Multiple cameras are supported via this system by identifying the specific camera's settings based on the combination of its model and serial numbers. Implemented automatic persistence of the following settings whenever they are explicitly (i.e., not by performing an exposure) modified via the QSICamera API: • anti-blooming • cooler state • cooler set point • fan mode • camera gain • LED alert setting • pre-exposure flushing • readout speed • shutter priority • sound alert setting The persistent settings values are automatically applied to a camera following successful connection. Because of the above persistent settings changes, I eliminated the QSICamera connect: method which took a preferences structure (the preferences structure itself was also removed).
February 26, 2012	Joe Shimkus	Corrected problems related to 600-series camera support. Moved public methods of QSICamera to a QSICamera protocol.

Copyright © 2012, Joe Shimkus

Date	Author	Change
October 27, 2011	Joe Shimkus	Changed version number to reflect the version of QSI software from which this code was created. This doesn't mean that QSICamera has all the same functionality (particularly in regard to functionality that can be created building on what is already present) but that particular changes (such as default values) are consistent with the QSI software of the same version number. Removed the ability to set the auto-zero control parameters. Per QSI, this is something end-users should not be modifying. Added support for 600-series cameras. Added trim adjustment field to QSIFilter. Minor corrections.
November 15, 2010	Joe Shimkus	Correction to discussion of startingColumn and startingRow to indicate they are in units of their respective binning values. Added an `autoZeroControl:` method to read the current settings for the camera.
November 11, 2010	Joe Shimkus	Initial version

Copyright © 2012, Joe Shimkus

Introduction

Trying to minimize the amount of equipment to carry into the field, I had been attempting to do astrophotography utilizing a digital SLR. While some have certainly produced wonderful results with a DSLR, I found it to be more taxing than enjoyable and have not produced any images that I would choose to share. As a consequence, I decided to the bite the bullet and be willing to carry more equipment if that equipment could give me better results; thus I chose to purchase a dedicated CCD camera.

As I looked into the various cameras available and tried to objectively weigh their pros and cons, there was one absolute that had to be met: the camera had to be controllable utilizing native software for Mac OS X. When all was said and done, I chose a Quantum Scientific Imaging (QSI), Inc. 583 camera with integrated filter wheel.

While the QSI 583 met all the requirements I had established, I was disappointed that QSI itself didn't incorporate some control software for Macs in their product package as they do for Windows. Further, while I have no complaints with the third-party software I purchased, the number of choices were severely limited.

QSI, to their credit, makes a set of C++ source code available for creating a library that can be used to control their 500 series cameras. While QSI's website specifically refers to Linux in relation to this code, it's source and I figured I could construct something from it for Mac OS X. What you have downloaded is the fruit of that effort.

Copyright © 2012, Joe Shimkus

Notices

The software has been built solely on x86 architecture and tested solely using a QSI 583ws camera.

Requirements

References

QSI 500 Series Linux API Reference Manual, Quantum Scientific Imaging, Inc.

Terminology

- In, used in method parameter descriptions. Indicates that the parameter is input to the method. In the case of buffer pointers, indicates that the contents of the buffer is input to the method.
- Out, used in method parameter descriptions. Indicates that the parameter is output from the method. In the case of buffer pointers, indicates that the contents of the buffer is output from the method.

Copyright © 2012, Joe Shimkus

What You've Downloaded

Although QSI provides the C++ library source code I previously mentioned, it is not a complete library solution. The software has dependencies on further code from Future Technology Devices International (FTDI) Ltd. as well as libusb. This is not what I wanted. What I wanted was to produce something that would work natively on Mac OS X, had no dependencies beyond what is included on every Mac, and would also be easy enough to deploy and use such that it might help spark efforts in the development of more Mac software for use with these cameras in scientific settings. To meet these self-imposed requirements, I had to understand QSI's software, FTDI's software and learn about USB. Further, I had to decide if I would attempt to utilize QSI's software as-is or potentially modify it as part of achieving my goals.

As part of my vision for ease of use I included thoughts on the ability to extend the software to support future camera products. This argued in favor of an object-oriented approach. Well, QSI's provided software was written in C++ so that would hopefully prove beneficial to such an approach. A drawback, however, of sticking with C++ would be the fact that objects created from it wouldn't be able to directly plug in to the Cocoa libraries from Apple. Certainly they would be able to utilize the Cocoa frameworks' functionality, but they themselves wouldn't be able to be directly referenced from the Cocoa frameworks (e.g., they couldn't be directly stored in instances of `NSArray`). On top of this I knew that the introspective capabilities of Objective-C would allow extending the object hierarchy of supported cameras in a simple, low-effort manner that just wouldn't be attainable using C++. The choice was clear. I wouldn't utilize QSI's C++ code as-is nor would I modify it. Rather, I would construct a new hierarchy of objects designed to integrate with Cocoa, providing the necessary functionality. QSI's source would be used solely as a reference.

So now you know that the software you're getting is:

- An object-oriented hierarchy supporting QSI 500 and 600 series cameras
- Written in Objective-C
- Integrated with Apple's Cocoa libraries
    - all objects ultimately descend from `NSObject`
- Dependent only on natively provided functionality of Mac OS X
    - needs to be linked with the Cocoa and IOKit frameworks

Additionally, the software internally utilizes the Objective-C `@synchronized()` directive to serialize commands sent to the camera and is thus safe for use in multithreaded environments.

Copyright © 2012, Joe Shimkus

What This Document Documents

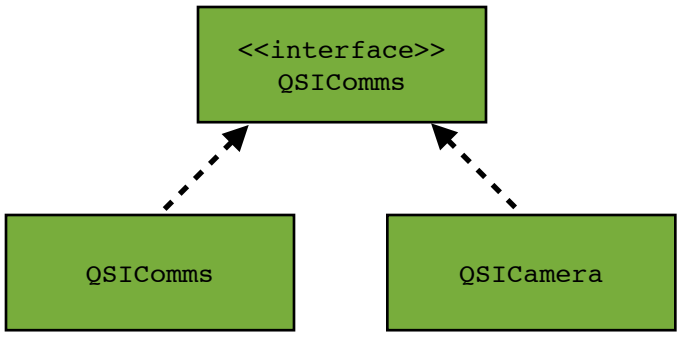
Essentially, what is documented are the entities identified in the All You Need To Know (mostly) section together with supplementary objects, enumerations and structures necessary to utilize those entities. Each section does, however, contain a diagram of the entire class hierarchy implemented by that section's software.

Copyright © 2012, Joe Shimkus

1. All You Need To Know (mostly)

If you just want to jump in and put the software to work, this is the section for you. The details of the various objects are presented later on; this section will simply give you a high-level view of how you can immediately utilize the software.

1.1. Object Hierarchy



1.2. Finding Cameras

Assuming you have your QSI camera(s) turned on and plugged in to your Mac, to have the QSICamera software find them…

```
NSArray *cameras = [QSIComms findAllCameras];
```

As you can see, `QSIComms` will return an `NSArray` with all the QSI cameras it was able to find. If no cameras were identified `QSIComms` will still return an `NSArray`, it will just have no entries. If something truly horrible occurred you won't even get an array back, so always check for `nil`.

Following the Cocoa reference counting object lifecycle paradigm, the `NSArray` has been auto-released so if you want it to stick around you'll need to retain it.

1.3. Using a Camera

Once you have the array of cameras you'll need to decide which one(s) to use; if you only have one camera the decision is easy. Identifying a camera for use in the case where there are multiple cameras attached to your Mac is a little more complicated. As a camera's specifics are not available until you establish a connection to it and the order of the cameras in the returned `NSArray` is not deterministic, to disambiguate the cameras you need to iterate over them establishing a connection to each one and checking each camera's model and serial numbers.

The process of establishing a connection to the camera is a simple one. Assuming that we wish to use the first camera in the returned `NSArray`, you establish a connection to it as follows (you'll want to keep a reference to the camera around for performing operations):

```
QSICamera *camera = [cameras objectAtIndex:0]

QSIStatus status  = [camera connect];
```

You'll need to confirm that the status you receive from the `connect` is `QSISuccess`. If it isn't, the connection attempt failed.

And that's it! You now have an established connection to your QSI camera and can begin using it. Congratulations!

Copyright © 2012, Joe Shimkus

2. Finding Cameras (for the developer)

While finding cameras for someone who just wants to use the QSICamera software is straightforward, for the developer who is trying to debug a problem or extend the software more needs to be known.

The steps to finding a camera (only USB cameras are currently supported) to be included in the `NSArray` returned from `QSIComms`' `findAllCameras` method are:

1. Use the IOKit USB methods to find all USB devices which match the correct product and vendor identifiers
2. For each found device
    2.1. Get the model name and verify that
        2.1.1. It begins with "`QSI`"
        2.1.2. It contains the appropriate series number
    2.2. Get the model number and verify that it is at least three (3) ASCII characters in length
    2.3. Construct a string composed of
        2.3.1. "`QSI`"
        2.3.2. The first three (3) characters of the model number
    2.4. If the model number is at least four (4) ASCII characters in length and the fourth character is "`c`" (denoting a color camera), append "c" to the string
    2.5. Append "`Camera`" to the string
    2.6. Use the string to locate the class object for the class with that name
    2.7. If the class object is found, use it to instantiate a `QSICamera` object of the correct class
    2.8. Add the instantiated `QSICamera` to the found camera array

In the case of the particular camera I own (a 583ws) the constructed string naming the appropriate class is `QSI583Camera`. If I had purchased the color version of the camera the string would have been `QSI583cCamera`.

Using the introspective capabilities of Objective-C allows for easy extension of the class hierarchy to support new cameras from QSI. As an example, if QSI were to introduce a new 500-series camera called the 598 all that would need to be done would be to add a `QSI598Camera` (and, if applicable, a `QSI598cCamera`) class to the class hierarchy. The runtime lookup of the class object via name would then automatically work for the new camera.

Copyright © 2012, Joe Shimkus

3. Common

3.1. QSIStatus

3.1.1. Semantics

Method status return value.

3.1.2. Definition

```
typedef enum _QSIStatus
{
  QSISuccess  = 0x00000000,
  QSIFailure, // Generic failure

  // Specific errors
  QSIAbortedExposure,
  QSIAllocationFailed,
  QSICloseFailed,
  QSICommandFailed,
  QSIConnected,
  QSIControlFailed,
  QSIInvalidParameter,
  QSINoExposure,
  QSINoFilter,
  QSINoImage,
  QSINoMemory,
  QSINotConnected,
  QSINotSupported,
  QSIOpenFailed,
  QSIReadFailed,
  QSIWriteFailed
} QSIStatus;
```

3.1.3. Notes

3.2. QSIObject

3.2.1. Semantics

Base class for all QSICamera classes; descended from `NSObject`.

Provides class-based access to necessary Objective-C runtime capabilities.

3.2.2. APIs

3.2.2.1. classNamed

3.2.2.1.1. Semantics

Provides access to Objective-C runtime capability to determine a class object from its name.

3.2.2.1.2. Declaration

```
+ (Class) className : (NSString *) aName;
```


Copyright © 2012, Joe Shimkus

3.2.2.1.3. Parameters

Name	In/Out	Usage
aName	In	The class name

3.2.2.1.4. Return Values

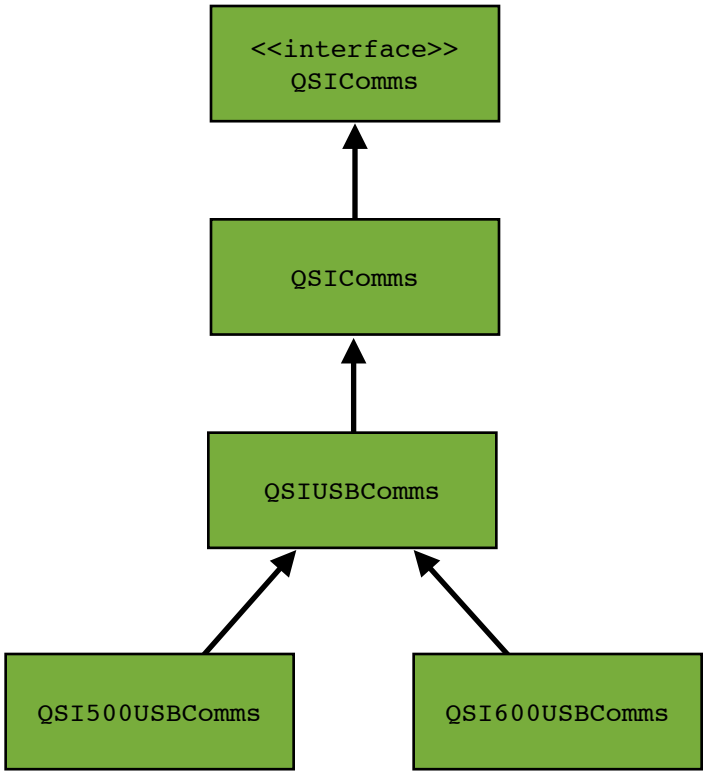
Value	Significance
`nil`	No class with the specified name was found
non-`nil`	The class object of the class with the specified name

3.2.2.1.5. Notes

Copyright © 2012, Joe Shimkus

4. Communications

4.1. Object Hierarchy



Copyright © 2012, Joe Shimkus

4.2. <<interface>> QSIComms

4.2.1. Semantics

An Objective-C protocol defining the abstract communication channel methods.

4.2.2. APIs

4.2.2.1. disconnect

4.2.2.1.1. Semantics

Disconnects from the communication channel.

If already disconnected, this is a no-op.

4.2.2.1.2. Declaration

```
- (OSIStatus) disconnect
```

4.2.2.1.3. Parameters

Name	In/Out	Usage
n/a	n/a	n/a

4.2.2.1.4. Return Values

Value	Significance
`QSISuccess`	Disconnection was successful

4.2.2.1.5. Notes

4.2.2.2. isConnected

4.2.2.2.1. Semantics

Reports as to whether or not a connection to the communication channel exists.

4.2.2.2.2. Declaration

```
- (bool) isConnected
```

4.2.2.2.3. Parameters

Name	In/Out	Usage
n/a	n/a	n/a

4.2.2.2.4. Return Values

Value	Significance
`YES`	A connection exists

Copyright © 2012, Joe Shimkus

Value	Significance
NO	A connection does not exist

4.2.2.2.5. Notes

Copyright © 2012, Joe Shimkus

4.3. QSIComms

4.3.1. Semantics

Implements the `QSIComms` interface.

Used to locate attached QSI cameras and communicate with them.

4.3.2. APIs

4.3.2.1. findAllCameras

4.3.2.1.1. Semantics

Returns an `NSArray` of all found QSI cameras.

4.3.2.1.2. Declaration

```
+ (NSArray *) findAllCameras;
```

4.3.2.1.3. Parameters

Name	In/Out	Usage
n/a	n/a	n/a

4.3.2.1.4. Return Values

Value	Significance
`NSArray *`	An array containing all found QSI cameras. If `nil`, an error occurred during processing.

4.3.2.1.5. Notes

The returned `NSArray` may contain no cameras.

4.3.2.2. connect

4.3.2.2.1. Semantics

Establishes the communication channel connection.

4.3.2.2.2. Declaration

```
- (QSIStatus) connect;
```

4.3.2.2.3. Parameters

Name	In/Out	Usage
n/a	n/a	n/a

Copyright © 2012, Joe Shimkus

4.3.2.2.4. Return Values

Value	Significance
`QSISuccess`	The method completed successfully.

4.3.2.2.5. Notes

4.3.2.3. extendedReadTimeout

4.3.2.3.1. Semantics

Reports the value for the extended read timeout; the value is in milliseconds.

4.3.2.3.2. Declaration

```
- (uint16_t) extendedReadTimeout;
```

4.3.2.3.3. Parameters

Name	In/Out	Usage
n/a	n/a	n/a

4.3.2.3.4. Return Values

Value	Significance
`any`	The timeout in milliseconds

4.3.2.3.5. Notes

4.3.2.4. extendedWriteTimeout

4.3.2.4.1. Semantics

Reports the value for the extended write timeout; the value is in milliseconds.

4.3.2.4.2. Declaration

```
- (uint16_t) extendedWriteTimeout;
```

4.3.2.4.3. Parameters

Name	In/Out	Usage
n/a	n/a	n/a

4.3.2.4.4. Return Values

Value	Significance
`any`	The timeout in milliseconds

Copyright © 2012, Joe Shimkus

4.3.2.4.5. Notes

4.3.2.5. purge

4.3.2.5.1. Semantics

Discards any outstanding data waiting to be either read or written.

4.3.2.5.2. Declaration

```
- (QSIStatus) purge;
```

4.3.2.5.3. Parameters

Name	In/Out	Usage
n/a	n/a	n/a

4.3.2.5.4. Return Values

Value	Significance
`QSISuccess`	The method completed successfully

4.3.2.5.5. Notes

4.3.2.6. readToBuffer:numberOfBytes:

4.3.2.6.1. Semantics

Reads from the communication channel into the specified buffer for the number of bytes specified.

4.3.2.6.2. Declaration

```
- (QSIStatus) readToBuffer : (void *) aBuffer
                numberOfBytes : (uint32_t *) aNumberOfBytes;
```

4.3.2.6.3. Parameters

Name	In/Out	Usage
aBuffer	Out	The buffer to hold the read data
aNumberOfBytes	In/Out	On input, the size of the buffer. On output, the number of bytes read into the buffer.

4.3.2.6.4. Return Values

Value	Significance
`QSISuccess`	The read completed successfully

Copyright © 2012, Joe Shimkus

4.3.2.6.5. Notes

4.3.2.7. setReadTimeout:andWriteTimeout:

4.3.2.7.1. Semantics

Set the read and write timeouts for the communication channel to the specified values; the values are in milliseconds.

4.3.2.7.2. Declaration

```
- (QSIStatus) setReadTimeout : (uint16_t) aReadTimeout
              andWriteTimeout : (uint16_t) aWriteTimeout;
```

4.3.2.7.3. Parameters

Name	In/Out	Usage
aReadTimeout	In	The read timeout in milliseconds
aWriteTimeout	In	The write timeout in milliseconds

4.3.2.7.4. Return Values

Value	Significance
`QSISuccess`	The method completed successfully

4.3.2.7.5. Notes

4.3.2.8. setToExtendedTimeouts

4.3.2.8.1. Semantics

Set the communication channel to use the extended read and write timeouts.

4.3.2.8.2. Declaration

```
- (QSIStatus) setToExtendedTimeouts;
```

4.3.2.8.3. Parameters

Name	In/Out	Usage
n/a	n/a	n/a

4.3.2.8.4. Return Values

Value	Significance
`QSISuccess`	The method completed successfully

Copyright © 2012, Joe Shimkus

4.3.2.8.5. Notes

4.3.2.9. setToStandardTimeouts

4.3.2.9.1. Semantics

Set the communication channel to use the standard read and write timeouts.

4.3.2.9.2. Declaration

```
- (QSIStatus) setToStandardTimeouts;
```

4.3.2.9.3. Parameters

Name	In/Out	Usage
n/a	n/a	n/a

4.3.2.9.4. Return Values

Value	Significance
`QSISuccess`	The method completed successfully

4.3.2.9.5. Notes

4.3.2.10.standardReadTimeout

4.3.2.10.1.Semantics

Reports the value for the standard read timeout; the value is in milliseconds.

4.3.2.10.2.Declaration

```
- (uint16_t) standardReadTimeout;
```

4.3.2.10.3.Parameters

Name	In/Out	Usage
n/a	n/a	n/a

4.3.2.10.4.Return Values

Value	Significance
`any`	The timeout in milliseconds

4.3.2.10.5.Notes

Copyright © 2012, Joe Shimkus

4.3.2.11.standardWriteTimeout

4.3.2.11.1.Semantics

Reports the value for the standard write timeout; the value is in milliseconds.

4.3.2.11.2.Declaration

```
- (uint16_t) standardWriteTimeout;
```

4.3.2.11.3.Parameters

Name	In/Out	Usage
n/a	n/a	n/a

4.3.2.11.4.Return Values

Value	Significance
`any`	The timeout in milliseconds

4.3.2.11.5.Notes

4.3.2.12.writeFromBuffer:numberOfBytes:

4.3.2.12.1.Semantics

Writes to the communication channel from the specified buffer for the number of bytes specified.

4.3.2.12.2.Declaration

```
- (QSIStatus) writeFromBuffer : (void *) aBuffer
                numberOfBytes : (uint32_t *) aNumberOfBytes;
```

4.3.2.12.3.Parameters

Name	In/Out	Usage
aBuffer	Out	The buffer holding the data to write.
aNumberOfBytes	In/Out	On input, the number of bytes to write from the buffer. On output, the number of bytes written from the buffer.

4.3.2.12.4.Return Values

Value	Significance
`QSISuccess`	The write completed successfully

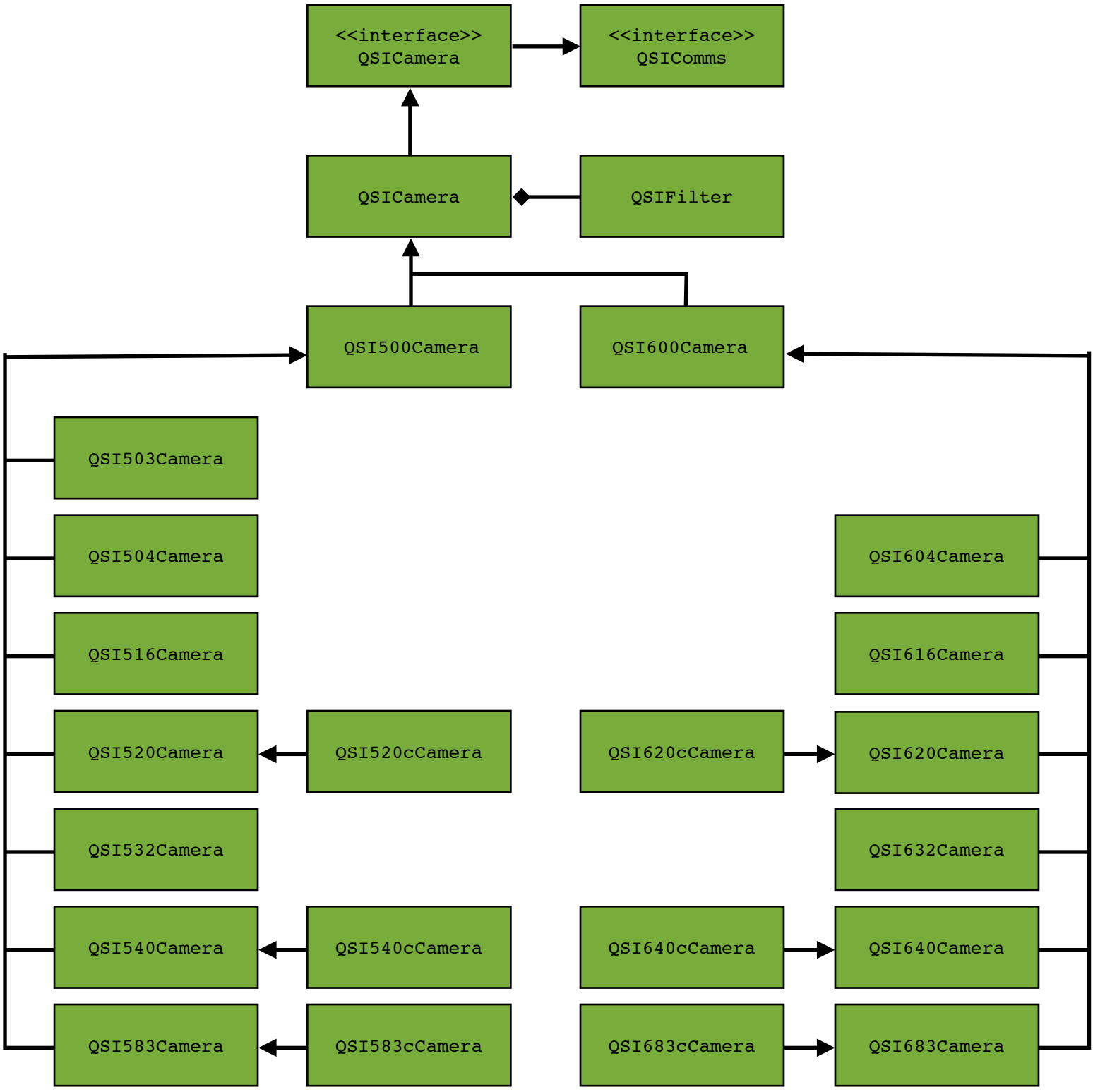
Copyright © 2012, Joe Shimkus

4.3.2.12.5.Notes

Copyright © 2012, Joe Shimkus

5. Components

5.1. Object Hierarchy



Copyright © 2012, Joe Shimkus

5.2. <<interface>> QSICamera

5.2.1. Semantics

An Objective-C protocol defining the abstract QSICamera public methods.

Inherits the QSIComms interface.

5.2.2. APIs

5.2.2.1. abortExposure

5.2.2.1.1. Semantics

If supported, aborts an in-progress exposure.

5.2.2.1.2. Declaration

```
- (QSIStatus) abortExposure;
```

5.2.2.1.3. Parameters

Name	In/Out	Usage
n/a	n/a	n/a

5.2.2.1.4. Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.
QSINotSupported	The requested functionality is not supported by the camera.

5.2.2.1.5. Notes

5.2.2.2. advancedDetails:

5.2.2.2.1. Semantics

Returns the camera's advanced details.

The returned object has been auto-released.

5.2.2.2.2. Declaration

```
- (QSIStatus) advancedDetails :
                (out QSICameraAdvancedDetails * *) anAdvancedDetails
```

5.2.2.2.3. Parameters

Name	In/Out	Usage
anAdvancedDetails	Out	The camera's advanced details.

Copyright © 2012, Joe Shimkus

5.2.2.2.4. Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.2.5. Notes

5.2.2.3. advancedSettings:

5.2.2.3.1. Semantics

Returns the camera's advanced settings.

The returned object has been auto-released.

5.2.2.3.2. Declaration

```
- (QSIStatus) advancedSettings :
    (out QSICameraAdvancedSettingsParameters * *)
                                                anAdvancedSettings;
```

5.2.2.3.3. Parameters

Name	In/Out	Usage
anAdvancedSettings	Out	The camera's advanced settings.

5.2.2.3.4. Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.3.5. Notes

5.2.2.4. autoZeroControl:

5.2.2.4.1. Semantics

Returns the camera's auto-zero control settings.

The returned object has been auto-released.

5.2.2.4.2. Declaration

```
- (QSIStatus) autoZeroControl :
          (out QSICameraAutoZeroControl * *) anAutoZeroControl;
```


Copyright © 2012, Joe Shimkus

5.2.2.4.3. Parameters

Name	In/Out	Usage
anAutoZeroControl	Out	The camera's auto-zero control settings.

5.2.2.4.4. Return Values

Value	Significance
`QSISuccess`	The method completed successfully.

5.2.2.4.5. Notes

5.2.2.5. cameraState:

5.2.2.5.1. Semantics

Returns the camera's current state.

5.2.2.5.2. Declaration

```
- (QSIStatus) cameraState : (out QSICameraState *) aCameraState;
```

5.2.2.5.3. Parameters

Name	In/Out	Usage
aCameraState	Out	The camera's state.

5.2.2.5.4. Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.5.5. Notes

5.2.2.6. canAbortExposure:

5.2.2.6.1. Semantics

Returns whether or not the camera supports aborting an in-progress exposure.

5.2.2.6.2. Declaration

```
- (QSIStatus) canAbortExposure : (out bool *) aCanAbortExposure;
```


Copyright © 2012, Joe Shimkus

5.2.2.6.3. Parameters

Name	In/Out	Usage
aCanAbortExposure	Out	YES, the camera supports aborting an in-progress exposure. NO, the camera does not support aborting an in-progress exposure.

5.2.2.6.4. Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.6.5. Notes

5.2.2.7. canGetCoolerPower:

5.2.2.7.1. Semantics

Returns whether or not the camera supports reporting the cooler's power setting.

5.2.2.7.2. Declaration

```
- (QSIStatus) canGetCoolerPower : (out bool *) aCanGetCoolerPower;
```

5.2.2.7.3. Parameters

Name	In/Out	Usage
aCanGetCoolerPower	Out	YES, the camera supports reporting the cooler's power setting. NO, the camera does not support reporting the cooler's power setting.

5.2.2.7.4. Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.7.5. Notes

Copyright © 2012, Joe Shimkus

5.2.2.8. canPulseGuide:

5.2.2.8.1. Semantics

Returns whether or not the camera supports usage for guiding.

5.2.2.8.2. Declaration

```
- (QSIStatus *) canPulseGuide : (out bool *) aCanPulseGuide;
```

5.2.2.8.3. Parameters

Name	In/Out	Usage
aCanPulseGuide	Out	YES, the camera supports being used to guide. NO, the camera does not support being used to guide.

5.2.2.8.4. Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.8.5. Notes

5.2.2.9. canSetReadoutSpeed:

5.2.2.9.1. Semantics

Returns whether or not the camera supports changing its readout speed.

5.2.2.9.2. Declaration

```
- (QSIStatus *) canSetReadoutSpeed :
                                (out bool *) aCanSetReadoutSpeed;
```

5.2.2.9.3. Parameters

Name	In/Out	Usage
aCanSetReadoutSpeed	Out	YES, the camera supports changing its readout speed. NO, the camera does not support changing its readout speed.

5.2.2.9.4. Return Values

Value	Significance
QSISuccess	The method completed successfully.

Copyright © 2012, Joe Shimkus

Value	Significance
QSINotConnected	There is no connection to the camera.

5.2.2.9.5. Notes

5.2.2.10.canSetTemperature:

5.2.2.10.1.Semantics

Reports whether or not the camera supports setting temperature control.

5.2.2.10.2.Declaration

```
- (QSIStatus) canSetTemperature : (out bool *) aCanSetTemperature;
```

5.2.2.10.3.Parameters

Name	In/Out	Usage
aCanSetTemperature	Out	YES, the camera supports setting temperature control. NO, the camera does not support setting temperature control.

5.2.2.10.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.10.5.Notes

5.2.2.11.ccdHeight:

5.2.2.11.1.Semantics

Returns the CCD height (i.e., number of rows) in pixels.

5.2.2.11.2.Declaration

```
- (QSIStatus) ccdHeight : (out uint16_t *) aHeight;
```

5.2.2.11.3.Parameters

Name	In/Out	Usage
aHeight	Out	The CCD's number of rows.

Copyright © 2012, Joe Shimkus

5.2.2.11.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.11.5.Notes

5.2.2.12.ccdTargetTemperature:

5.2.2.12.1.Semantics

Returns the temperature (°C) the CCD has been targeted to attain.

5.2.2.12.2.Declaration

```
- (QSIStatus) ccdTargetTemperature :
                                (out double *) aTargetTemperature;
```

5.2.2.12.3.Parameters

Name	In/Out	Usage
aTargetTemperature	Out	The CCD's target temperature (°C).

5.2.2.12.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.12.5.Notes

5.2.2.13.ccdTemperature:

5.2.2.13.1.Semantics

Returns the current temperature (°C) of the CCD.

5.2.2.13.2.Declaration

```
- (QSIStatus) ccdTemperature : (out double *) aCCDTemperature;
```

5.2.2.13.3.Parameters

Name	In/Out	Usage
aCCDTemperature	Out	The CCD's current temperature (°C).

Copyright © 2012, Joe Shimkus

5.2.2.13.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.13.5.Notes

5.2.2.14.ccdWidth:

5.2.2.14.1.Semantics

Returns the CCD's width (i.e., number of columns) in pixels.

5.2.2.14.2.Declaration

```
- (OSIStatus) ccdWidth : (out uint16_t *) aWidth;
```

5.2.2.14.3.Parameters

Name	In/Out	Usage
aWidth	Out	The CCD's number of columns.

5.2.2.14.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.14.5.Notes

5.2.2.15.colorCamera

5.2.2.15.1.Semantics

Returns whether the camera is a one-shot color or monochrome camera.

5.2.2.15.2.Declaration

```
- (bool) colorCamera;
```

5.2.2.15.3.Parameters

Name	In/Out	Usage
n/a	n/a	n/a

Copyright © 2012, Joe Shimkus

5.2.2.15.4.Return Values

Value	Significance
YES	The camera is a one-shot color camera.
NO	The camera is a monochrome camera.

5.2.2.15.5.Notes

5.2.2.16.columnBinning:

5.2.2.16.1.Semantics

Returns the current column binning setting.

5.2.2.16.2.Declaration

```
- (OSIStatus) columnBinning : (out uint16_t *) aBinning;
```

5.2.2.16.3.Parameters

Name	In/Out	Usage
aBinning	Out	The current column binning factor.

5.2.2.16.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.

5.2.2.16.5.Notes

5.2.2.17.combinedDetails:

5.2.2.17.1.Semantics

Returns the combined details of the camera. This provides “one-stop” querying for the camera’s particulars.

The returned object has been auto-released.

5.2.2.17.2.Declaration

```
- (OSIStatus) combinedDetails :
                (out OSICameraCombinedDetails * *) aCombinedDetails
```

5.2.2.17.3.Parameters

Name	In/Out	Usage
aCombinedDetails	Out	The camera’s details.

Copyright © 2012, Joe Shimkus

5.2.2.17.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.17.5.Notes

5.2.2.18.connect

5.2.2.18.1.Semantics

Establishes a connection to the camera.

Persistent settings values are applied after successfully connecting.

5.2.2.18.2.Declaration

```
- (QSIStatus) connect
```

5.2.2.18.3.Parameters

Name	In/Out	Usage
n/a	n/a	n/a

5.2.2.18.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.

5.2.2.18.5.Notes

5.2.2.19.coolerPower:

5.2.2.19.1.Semantics

Reports the current state of the cooler's power consumption as a percentage (0 - 100).

5.2.2.19.2.Declaration

```
- (QSIStatus) coolerPower : (out double *) aCoolerPower;
```

5.2.2.19.3.Parameters

Name	In/Out	Usage
aCoolerPower	Out	The cooler's power consumption as a percentage.

Copyright © 2012, Joe Shimkus

5.2.2.19.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.
QSINotSupported	The requested functionality is not supported by the camera.

5.2.2.19.5.Notes

5.2.2.20.electronsPerADU:

5.2.2.20.1.Semantics

Reports the CCD's electrons per ADU value.

5.2.2.20.2.Declaration

```
- (QSIStatus) electronsPerADU : (out double *) anElectronsPerADU;
```

5.2.2.20.3.Parameters

Name	In/Out	Usage
anElectronsPerADU	Out	The CCD's electrons per ADU value.

5.2.2.20.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.20.5.Notes

5.2.2.21.exposeForDuration:withShutterOpen:

5.2.2.21.1.Semantics

Utilizing the camera's current exposure settings, initiates an exposure of the specified duration with the shutter either open or closed.

5.2.2.21.2.Declaration

```
- (QSIStatus) exposeForDuration : (in uint32_t) aDuration
                withShutterOpen : (in bool) aShutterOpen;
```


Copyright © 2012, Joe Shimkus

5.2.2.21.3.Parameters

Name	In/Out	Usage
aDuration	In	The length of the exposure in milliseconds.
aShutterOpen	In	`YES`, the exposure is made with the shutter open. `NO`, the exposure is made with the shutter closed.

5.2.2.21.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSIInvalid Parameter`	One or more of the exposure parameters is out of range.

5.2.2.21.5.Notes

The camera's exposure settings are not validated until a request to initiate the exposure is made. The requirements for valid exposure settings are:

- The exposure's physical pixel width, taking into account the exposure's starting column and column binning setting, must not "run off the end" of the CCD; i.e., the following condition must be satisfied:

  ((startingColumn + width) * columnBinning) <= number of CCD columns

- The exposure's physical pixel height, taking into account the exposure's starting row and row binning setting, must not "run off the top" of the CCD; i.e., the following condition must be satistified:

  ((startingRow + height) * rowBinning) <= number of CCD rows

- Neither the column nor the row binning settings can exceed their respective maximums
- The column and row binning settings must be the same unless the camera supports asymmetric binning
- The exposure duration must either be zero (0) or between the minimum and maximum, inclusive, exposure values the camera supports

5.2.2.22.exposeUsingRequest:

5.2.2.22.1.Semantics

Utilizing the specified exposure request, sets the camera's exposure settings and initiates the exposure.

For those parameters in the exposure request controlling operations that the camera does not support (e.g., the anti-blooming setting), the `QSINotSupported` failures of attempting to set them are ignored.

Copyright © 2012, Joe Shimkus

5.2.2.22.2.Declaration

```
- (QSIStatus) exposeUsingRequest :
                (in QSICameraExposureRequest *) anExposureRequest;
```

5.2.2.22.3.Parameters

Name	In/Out	Usage
n/a	n/a	n/a

5.2.2.22.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSIInvalid Parameter`	One or more of the exposure parameters is out of range.

5.2.2.22.5.Notes

As `exposeUsingRequest:` is built atop `exposeForDuration:withShutterOpen:`, the exposure settings are validated by `exposeForDuration:withShutterOpen:`. Consequently, if a `QSIInvalidParameter` error is returned from `exposeUsingRequest:`, the camera's exposure settings will have been changed from what they were before the invocation of `exposeUsingRequest:`.

5.2.2.23.exposureHeight:

5.2.2.23.1.Semantics

Returns the exposure height (i.e., number of rows) setting.

The value is in logical pixels; i.e., units of the row binning setting.

5.2.2.23.2.Declaration

```
- (QSIStatus) exposureHeight : (out uint16_t *) aHeight;
```

5.2.2.23.3.Parameters

Name	In/Out	Usage
aHeight	Out	The exposure height setting.

5.2.2.23.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.

Copyright © 2012, Joe Shimkus

5.2.2.23.5.Notes

5.2.2.24.exposureLastDuration:

5.2.2.24.1.Semantics

Returns the duration of the last initiated exposure.

The value is in milliseconds.

5.2.2.24.2.Declaration

```
- (QSIStatus) exposureLastDuration : (out uint32_t *) aDuration;
```

5.2.2.24.3.Parameters

Name	In/Out	Usage
aDuration	Out	The duration of the last initiated exposure.

5.2.2.24.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINoExposure`	No exposure has yet been taken.

5.2.2.24.5.Notes

5.2.2.25.exposureSettings:

5.2.2.25.1.Semantics

Returns the entirety of the camera's current exposure settings.

The returned object has been auto-released.

5.2.2.25.2.Declaration

```
- (QSIStatus) exposureSettings :
        (out QSICameraExposureParameters * *) anExposureSettings;
```

5.2.2.25.3.Parameters

Name	In/Out	Usage
anExposureSettings	Out	The camera's current exposure settings.

Copyright © 2012, Joe Shimkus

5.2.2.25.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.

5.2.2.25.5.Notes

5.2.2.26.exposureStartingColumn:

5.2.2.26.1.Semantics

Returns the starting column of the exposure in pixels.

The value is in logical pixels; i.e., units of the column binning setting.

5.2.2.26.2.Declaration

```
- (QSIStatus) exposureStartingColumn : (out uint16_t *) aColumn;
```

5.2.2.26.3.Parameters

Name	In/Out	Usage
aColumn	Out	The exposure's starting column in pixels.

5.2.2.26.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.

5.2.2.26.5.Notes

5.2.2.27.exposureStartingRow:

5.2.2.27.1.Semantics

Returns the starting row of the exposure in pixels.

The value is in logical pixels; i.e., units of the row binning setting.

5.2.2.27.2.Declaration

```
- (QSIStatus) exposureStartingRow : (out uint16_t *) aRow;
```

5.2.2.27.3.Parameters

Name	In/Out	Usage
aRow	Out	The exposure's starting row in pixels.

Copyright © 2012, Joe Shimkus

5.2.2.27.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.

5.2.2.27.5.Notes

5.2.2.28.exposureWidth:

5.2.2.28.1.Semantics

Returns the exposure width (i.e., number of columns) setting.

The value is in logical pixels; i.e., units of the column binning setting.

5.2.2.28.2.Declaration

```
- (QSIStatus) exposureWidth : (out uint16_t *) aWidth;
```

5.2.2.28.3.Parameters

Name	In/Out	Usage
aWidth	Out	The exposure width setting.

5.2.2.28.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.

5.2.2.28.5.Notes

5.2.2.29.fanMode:

5.2.2.29.1.Semantics

Reports the current mode of the camera's fans.

5.2.2.29.2.Declaration

```
- (QSIStatus) fanMode : (out QSICameraFanMode *) aFanMode;
```

5.2.2.29.3.Parameters

Name	In/Out	Usage
aFanMode	Out	The current mode of the camera's fans.

Copyright © 2012, Joe Shimkus

5.2.2.29.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.29.5.Notes

5.2.2.30.filterCount:

5.2.2.30.1.Semantics

Returns the number of filter positions the camera has.

5.2.2.30.2.Declaration

```
- (QSIStatus) filterCount : (out uint8_t *) aFilterCount;
```

5.2.2.30.3.Parameters

Name	In/Out	Usage
aFilterCount	Out	The number of filter positions.

5.2.2.30.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.30.5.Notes

5.2.2.31.filterPosition:

5.2.2.31.1.Semantics

Returns which filter position is currently in place before the CCD.

The position is 1-relative.

5.2.2.31.2.Declaration

```
- (QSIStatus) filterPosition : (out uint8_t *) aFilterPosition;
```

5.2.2.31.3.Parameters

Name	In/Out	Usage
aFilterPosition	Out	The filter position currently in place before the CCD.

Copyright © 2012, Joe Shimkus

5.2.2.31.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINoFilter`	The camera has no filters.

5.2.2.31.5.Notes

5.2.2.32.filterWheelConnected:

5.2.2.32.1.Semantics

Returns if there is a filter wheel connected.

To be connected, a connection to the camera must have been established and the camera must have a filter wheel.

5.2.2.32.2.Declaration

```
- (QSIStatus) filterWheelConnected :
                                    (out bool *) aFilterWheelConnected;
```

5.2.2.32.3.Parameters

Name	In/Out	Usage
aFilterWheel Connected	Out	`YES`, a connection exists to the camera and the camera has a filter wheel. `NO`, one (or both) of the following: • there is no connection to the camera • the camera has no filter wheel

5.2.2.32.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.

5.2.2.32.5.Notes

5.2.2.33.focusOffset:forFilter:

5.2.2.33.1.Semantics

Reports the focus offset adjustment for the filter at the specified position.

The position is 1-relative.

Copyright © 2012, Joe Shimkus

5.2.2.33.2.Declaration

```
- (QSIStatus) focusOffset : (out int32_t *) aFocusOffset
                forFilter : (in uint8_t) aFilterPosition;
```

5.2.2.33.3.Parameters

Name	In/Out	Usage
aFocusOffset	Out	The filter's focus offset adjustment.
aFilterPosition	In	The position of the filter in question.

5.2.2.33.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINoFilter`	The camera has no filters.
`QSIInvalid Parameter`	The specified filter position is out of range.

5.2.2.33.5.Notes

5.2.2.34.fullWellCapacity:

5.2.2.34.1.Semantics

Reports the full well capacity of the camera's CCD.

5.2.2.34.2.Declaration

```
- (QSIStatus) fullWellCapacity : (out double *) aFullWellCapacity;
```

5.2.2.34.3.Parameters

Name	In/Out	Usage
aFullWellCapacity	Out	The CCD's full well capacity.

5.2.2.34.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

Copyright © 2012, Joe Shimkus

5.2.2.34.5.Notes

5.2.2.35.getName:forFilter:

5.2.2.35.1.Semantics

Reports the name assigned to the filter at the specified position.

The position is 1-relative.

5.2.2.35.2.Declaration

```
- (QSIStatus) getName : (out NSString * *) aName
             forFilter : (in uint8_t) aFilterPosition;
```

5.2.2.35.3.Parameters

Name	In/Out	Usage
aName	Out	The filter's assigned name.
aFilterPosition	In	The position of the filter in question.

5.2.2.35.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINoFilter`	The camera has no filters.
`QSIInvalid Parameter`	The specified filter position is out of range.

5.2.2.35.5.Notes

5.2.2.36.hasFilterWheel:

5.2.2.36.1.Semantics

Returns whether or not the camera has a filter wheel.

5.2.2.36.2.Declaration

```
- (QSIStatus) hasFilterWheel : (out bool *) aHasFilterWheel;
```

5.2.2.36.3.Parameters

Name	In/Out	Usage
aHasFilterWheel	Out	`YES`, the camera has a filter wheel. `NO`, the camera does not have a filter wheel.

Copyright © 2012, Joe Shimkus

5.2.2.36.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.36.5.Notes

5.2.2.37.hasShutter:

5.2.2.37.1.Semantics

Returns whether or not the camera has a shutter.

5.2.2.37.2.Declaration

```
- (QSIStatus) hasShutter : (out bool *) aHasShutter;
```

5.2.2.37.3.Parameters

Name	In/Out	Usage
aHasShutter	Out	YES, the camera has a shutter. NO, the camera does not have a shutter.

5.2.2.37.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.37.5.Notes

5.2.2.38.heatsinkTemperature:

5.2.2.38.1.Semantics

Returns the temperature (°C) of the camera's heatsink.

5.2.2.38.2.Declaration

```
- (QSIStatus) heatsinkTemperature :
                                (out double *) aHeatsinkTemperature;
```

5.2.2.38.3.Parameters

Name	In/Out	Usage
aHeatsinkTemperature	Out	The camera's heatsink's temperature (°C).

Copyright © 2012, Joe Shimkus

5.2.2.38.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.38.5.Notes

5.2.2.39.imageArray:

5.2.2.39.1.Semantics

Returns the image data taken by the last exposure.

If auto-zeroing is enabled, the returned image data has been auto-zeroed; the data as read from the CCD is not modified.

5.2.2.39.2.Declaration

```
- (QSIStatus) imageArray : (inout NSMutableData *) anImageArray;
```

5.2.2.39.3.Parameters

Name	In/Out	Usage
anImageArray	Out	The image.

5.2.2.39.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINoImage`	There is no image available. Failing some other error, if an exposure has been started this is a temporary situation.

5.2.2.39.5.Notes

The provided `NSMutableData` destination for the image data is presumed to be large enough to hold the image. See `imageArrayWidth:height:andElementSize:`.

5.2.2.40.imageArrayWidth:height:andElementSize:

5.2.2.40.1.Semantics

Returns the particulars (width, height and bytes/pixel) of the last exposed image.

Use these values to determine the necessary size (width * height * bytes/pixel) of the `NSMutableData` object passed to `imageArray:`.

Copyright © 2012, Joe Shimkus

5.2.2.40.2.Declaration

```
- (QSIStatus) imageArrayWidth : (out uint16_t *) aWidth
                       height : (out uint16_t *) aHeight
                andElementSize : (out uint16_t *) anElementSize;
```

5.2.2.40.3.Parameters

Name	In/Out	Usage
aWidth	Out	The image width in pixels.
aHeight	Out	The image height in pixels.
anElementSize	Out	The per pixel data size.

5.2.2.40.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINoImage`	There is no image available. Failing some other error, if an exposure has been started this is a temporary situation.

5.2.2.40.5.Notes

5.2.2.41.imageIsReady:

5.2.2.41.1.Semantics

Returns if an image is ready to be transferred from the camera.

If the output value is `YES`, the data will have already been read from the CCD into a holding area of the camera object.

5.2.2.41.2.Declaration

```
- (QSIStatus) imageIsReady : (out bool *) anImageIsReady;
```

5.2.2.41.3.Parameters

Name	In/Out	Usage
anImageIsReady	Out	`YES`, an image exists ready for transfer from the camera. `NO`, there is no image ready to be transferred from the camera.

Copyright © 2012, Joe Shimkus

5.2.2.41.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.41.5.Notes

Assuming an exposure has been successfully initiated, the most basic code for waiting for the image to be ready and transferring it from the camera would be:

```
QSIStatus   status;
bool        imageReady;

do
{
  status = [camera imageIsReady:&imageReady];
} while ((status == QSISuccess) && (! imageReady));

uint16_t  imageWidth;
uint16_t  imageHeight;
uint16_t  elementSize;

if ((status == QSISuccess) && (imageReady))
{
  status = [camera imageArrayWidth:&imageWidth
                            height:&imageHeight
                     andElementSize:&elementSize];
}

NSMutableData *imageData;
if (status == QSISuccess)
{
  imageData = [[[NSMutableData alloc]
                 initWithLength:(imageWidth * imageHeight *
                                 elementSize)] autorelease];
  if (imageData == nil)
  {
    status = QSIAllocationFailed;
  }
}

if (status == QSISuccess)
{
  status = [camera imageArray:imageData];
}
```

5.2.2.42.isCoolerOn:

5.2.2.42.1.Semantics

Returns whether or not the camera's cooler is operating.

Copyright © 2012, Joe Shimkus

5.2.2.42.2.Declaration

```
- (QSIStatus) isCoolerOn : (out bool *) aIsCoolerOn;
```

5.2.2.42.3.Parameters

Name	In/Out	Usage
aIsCoolerOn	Out	YES, the cooler is operating. NO, the cooler is not operating.

5.2.2.42.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.42.5.Notes

5.2.2.43.isLEDEnabled:

5.2.2.43.1.Semantics

Returns whether or not the camera's LED status indicator is enabled.

5.2.2.43.2.Declaration

```
- (QSIStatus) isLEDEnabled : (out bool *) anLEDEnabled;
```

5.2.2.43.3.Parameters

Name	In/Out	Usage
anLEDEnabled	Out	YES, the LED status indicator is enabled. NO, the LED status indicator is not enabled.

5.2.2.43.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.43.5.Notes

Copyright © 2012, Joe Shimkus

5.2.2.44.isMainCamera:

5.2.2.44.1.Semantics

Returns whether the camera is operating as the main camera or as a guider.

5.2.2.44.2.Declaration

```
- (QSIStatus) isMainCamera : (out bool *) aMainCamera;
```

5.2.2.44.3.Parameters

Name	In/Out	Usage
aMainCamera	Out	YES, the camera is operating as the main camera. NO, the camera is operating as a guider.

5.2.2.44.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.

5.2.2.44.5.Notes

5.2.2.45.isPulseGuiding:

5.2.2.45.1.Semantics

Returns whether or not the camera is actively performing a guide operation.

5.2.2.45.2.Declaration

```
- (QSIStatus) isPulseGuiding : (out bool *) anIsPulseGuiding;
```

5.2.2.45.3.Parameters

Name	In/Out	Usage
anIsPulseGuiding	Out	YES, the camera is actively performing a guide operation. NO, the camera is not actively performing a guide operation.

5.2.2.45.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

Copyright © 2012, Joe Shimkus

5.2.2.45.5.Notes

5.2.2.46.isSoundEnabled:

5.2.2.46.1.Semantics

Returns whether or not the camera's sound status indicator is enabled.

5.2.2.46.2.Declaration

```
- (QSIStatus) isSoundEnabled : (out bool *) aSoundEnabled;
```

5.2.2.46.3.Parameters

Name	In/Out	Usage
aSoundEnabled	Out	`YES`, the sound status indicator is enabled. `NO`, the sound status indicator is not enabled.

5.2.2.46.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.46.5.Notes

5.2.2.47.maxADU:

5.2.2.47.1.Semantics

Returns the CCD's maximum ADU value.

5.2.2.47.2.Declaration

```
- (QSIStatus) maxADU : (out uint32_t *) aMaxADU;
```

5.2.2.47.3.Parameters

Name	In/Out	Usage
aMaxADU	Out	The CCD's maximum ADU value.

5.2.2.47.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

Copyright © 2012, Joe Shimkus

5.2.2.47.5.Notes

5.2.2.48.maxColumnBinning:

5.2.2.48.1.Semantics

Returns the camera's maximum supported column binning.

5.2.2.48.2.Declaration

```
- (QSIStatus) maxColumnBinning : (out uint16_t *) aBinning;
```

5.2.2.48.3.Parameters

Name	In/Out	Usage
aBinning	Out	The maximum supported column binning.

5.2.2.48.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.48.5.Notes

5.2.2.49.maxPixelsPerBlock:

5.2.2.49.1.Semantics

Returns the maximum number of pixels the camera will read in a single operation from the CCD.

5.2.2.49.2.Declaration

```
- (QSIStatus) maxPixelsPerBlock :
                                    (out int32_t *) aMaxPixelsPerBlock;
```

5.2.2.49.3.Parameters

Name	In/Out	Usage
aMaxPixelsPerBlock	Out	The maximum number of pixels the camera will read from the CCD in a single operation.

5.2.2.49.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.

Copyright © 2012, Joe Shimkus

5.2.2.49.5.Notes

5.2.2.50.maxRowBinning:

5.2.2.50.1.Semantics

Returns the camera's maximum supported row binning.

5.2.2.50.2.Declaration

```
- (QSIStatus) maxRowBinning : (out uint16_t *) aBinning;
```

5.2.2.50.3.Parameters

Name	In/Out	Usage
aBinning	Out	The maximum supported row binning.

5.2.2.50.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.50.5.Notes

5.2.2.51.modelName:

5.2.2.51.1.Semantics

Returns the camera's model name.

5.2.2.51.2.Declaration

```
- (QSIStatus) modelName : (out NSString * *) aModelName;
```

5.2.2.51.3.Parameters

Name	In/Out	Usage
aModelName	Out	The model name of the camera.

5.2.2.51.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

Copyright © 2012, Joe Shimkus

5.2.2.51.5.Notes

5.2.2.52.modelNumber:

5.2.2.52.1.Semantics

Returns the camera's model number.

5.2.2.52.2.Declaration

```
- (QSIStatus) modelNumber : (out NSString * *) aModelNumber;
```

5.2.2.52.3.Parameters

Name	In/Out	Usage
aModelNumber	Out	The model number of the camera.

5.2.2.52.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.52.5.Notes

5.2.2.53.pixelHeight:

5.2.2.53.1.Semantics

The vertical size of the physical pixels of the CCD in microns (μm).

5.2.2.53.2.Declaration

```
- (QSIStatus) pixelHeight : (out double *) aHeight;
```

5.2.2.53.3.Parameters

Name	In/Out	Usage
aHeight	Out	The CCD's physical pixel vertical size.

5.2.2.53.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

Copyright © 2012, Joe Shimkus

5.2.2.53.5.Notes

5.2.2.54.pixelWidth:

5.2.2.54.1.Semantics

The horizontal size of the physical pixels of the CCD in microns (μm).

5.2.2.54.2.Declaration

```
- (QSIStatus) pixelWidth : (out double *) aWidth;
```

5.2.2.54.3.Parameters

Name	In/Out	Usage
aWidth	Out	The CCD's physical pixel horizontal size.

5.2.2.54.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.54.5.Notes

5.2.2.55.powerOfTwoBinning:

5.2.2.55.1.Semantics

Returns whether the camera bins in powers of two or in units of one.

5.2.2.55.2.Declaration

```
- (QSIStatus) powerOfTwoBinning : (out bool *) aPowerOfTwoBinning;
```

5.2.2.55.3.Parameters

Name	In/Out	Usage
aPowerOfTwoBinning	Out	`YES`, the camera bins in powers of two. `NO`, the camera bins in units of one.

5.2.2.55.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

Copyright © 2012, Joe Shimkus

5.2.2.55.5.Notes

5.2.2.56.pulseGuide:withDuration:

5.2.2.56.1.Semantics

Instructs the camera to perform a guiding operation in the specified direction with the specified duration (in milliseconds) or to abort any in-progress guiding operation.

If the camera has an in-progress guiding operation when this method is invoked, the in-progress operation is aborted and the new guiding operation begun.

5.2.2.56.2.Declaration

```
- (QSIStatus) pulseGuide : (in QSICameraGuiding) aGuideDirection
             withDuration : (in uint16_t) aDuration;
```

5.2.2.56.3.Parameters

Name	In/Out	Usage
aGuideDirection	In	The direction in which to guide or an abort request.
aDuration	In	The duration of the guiding operation in milliseconds. This value is ignored if an abort has been requested.

5.2.2.56.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINotSupported`	The requested functionality is not supported by the camera.

5.2.2.56.5.Notes

5.2.2.57.readoutSpeed:

5.2.2.57.1.Semantics

Returns the current readout speed.

5.2.2.57.2.Declaration

```
- (QSIStatus) readoutSpeed :
                        (out QSICameraReadoutSpeed *) aReadoutSpeed;
```

5.2.2.57.3.Parameters

Name	In/Out	Usage
aReadoutSpeed	Out	The current readout speed.

Copyright © 2012, Joe Shimkus

5.2.2.57.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.57.5.Notes

5.2.2.58.rowBinning:

5.2.2.58.1.Semantics

Returns the current row binning value.

5.2.2.58.2.Declaration

```
- (QSIStatus) rowBinning : (out uint16_t *) aBinning;
```

5.2.2.58.3.Parameters

Name	In/Out	Usage
aBinning	Out	The current row binning value.

5.2.2.58.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.

5.2.2.58.5.Notes

5.2.2.59.serialNumber:

5.2.2.59.1.Semantics

Returns the camera's serial number.

5.2.2.59.2.Declaration

```
- (QSIStatus) serialNumber : (out NSString * *) aSerialNumber;
```

5.2.2.59.3.Parameters

Name	In/Out	Usage
aSerialNumber	Out	The serial number of the camera.

Copyright © 2012, Joe Shimkus

5.2.2.59.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.59.5.Notes

5.2.2.60.setAntiBlooming:

5.2.2.60.1.Semantics

Set's the anti-blooming state of the camera.

If successful, the value is stored in the camera's persistent settings.

5.2.2.60.2.Declaration

```
- (QSIStatus) setAntiBlooming :
                                (in QSICameraAntiBloom) anAntiBlooming;
```

5.2.2.60.3.Parameters

Name	In/Out	Usage
anAntiBlooming	In	The desired anti-blooming state.

5.2.2.60.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.
QSINotSupported	The requested functionality is not supported by the camera.

5.2.2.60.5.Notes

5.2.2.61.setCCDTargetTemperature:

5.2.2.61.1.Semantics

Set's the CCD temperature (°C) that the cooler should attempt to achieve.

If successful, the value is stored in the camera's persistent settings.

5.2.2.61.2.Declaration

```
- (QSIStatus) setCCDTargetTemperature :
                                    (in double) aTargetTemperature;
```


Copyright © 2012, Joe Shimkus

5.2.2.61.3.Parameters

Name	In/Out	Usage
aTargetTemperature	In	The desired CCD temperature (°C).

5.2.2.61.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINotSupported`	The requested functionality is not supported by the camera.
`QSIInvalid Parameter`	The specified temperature is out of the supported range.

5.2.2.61.5.Notes

The range checking of this method limits the target temperature to between -100 °C and +100 °C (both exclusive); not that either of these numbers are achievable.

5.2.2.62.setColumnBinning:

5.2.2.62.1.Semantics

Sets the camera's column binning value.

5.2.2.62.2.Declaration

```
- (QSIStatus) setColumnBinning : (in uint16_t) aBinning;
```

5.2.2.62.3.Parameters

Name	In/Out	Usage
aBinning	In	The column binning value to use.

5.2.2.62.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSIInvalid Parameter`	The specified binning value is out of the supported range.

5.2.2.62.5.Notes

Copyright © 2012, Joe Shimkus

5.2.2.63.setCoolerOn:

5.2.2.63.1.Semantics

Turns the camera's cooler on and off.

If successful, the value is stored in the camera's persistent settings.

5.2.2.63.2.Declaration

```
- (QSIStatus) setCoolerOn : (in bool) aCoolerOn;
```

5.2.2.63.3.Parameters

Name	In/Out	Usage
aCoolerOn	In	`YES`, turns the cooler on. `NO`, turns the cooler off.

5.2.2.63.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.63.5.Notes

5.2.2.64.setExposureHeight:

5.2.2.64.1.Semantics

Set's the exposure height in logical pixels (i.e., in units of the row binning setting)

5.2.2.64.2.Declaration

```
- (QSIStatus) setExposureHeight : (in uint16_t) aHeight;
```

5.2.2.64.3.Parameters

Name	In/Out	Usage
aHeight	In	The exposure height.

5.2.2.64.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.

5.2.2.64.5.Notes

Copyright © 2012, Joe Shimkus

5.2.2.65.setExposureStartingColumn:

5.2.2.65.1.Semantics

Sets the exposure beginning column in pixels.

The value is in logical pixels; i.e., units of the column binning setting.

5.2.2.65.2.Declaration

```
- (OSIStatus) setExposureStartingColumn : (in uint16_t) aColumn;
```

5.2.2.65.3.Parameters

Name	In/Out	Usage
aColumn	In	The exposure beginning column.

5.2.2.65.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.

5.2.2.65.5.Notes

5.2.2.66.setExposureStartingRow:

5.2.2.66.1.Semantics

Sets the exposure beginning row in pixels.

The value is in logical pixels; i.e., units of the row binning setting.

5.2.2.66.2.Declaration

```
- (OSIStatus) setExposureStartingRow : (in uint16_t) aRow;
```

5.2.2.66.3.Parameters

Name	In/Out	Usage
aRow	In	The exposure beginning row.

5.2.2.66.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.

5.2.2.66.5.Notes

Copyright © 2012, Joe Shimkus

5.2.2.67.setExposureWidth:

5.2.2.67.1.Semantics

Set's the exposure width in logical pixels (i.e., in units of the column binning setting)

5.2.2.67.2.Declaration

```
- (OSIStatus) setExposureWidth : (in uint16_t) aWidth
```

5.2.2.67.3.Parameters

Name	In/Out	Usage
aWidth	In	The exposure width.

5.2.2.67.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.

5.2.2.67.5.Notes

5.2.2.68.setFanMode:

5.2.2.68.1.Semantics

Sets the mode of the camera's fans.

If successful, the value is stored in the camera's persistent settings.

5.2.2.68.2.Declaration

```
- (OSIStatus) setFanMode : (in OSICameraFanMode) aFanMode;
```

5.2.2.68.3.Parameters

Name	In/Out	Usage
aFanMode	In	The fan mode to set.

5.2.2.68.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.
QSINotSupported	The requested functionality is not supported by the camera.

5.2.2.68.5.Notes

Copyright © 2012, Joe Shimkus

5.2.2.69.setFilterPosition:

5.2.2.69.1.Semantics

Sets the filter to position in front of the CCD.

The position is 1-relative.

The filter is moved into position immediately.

5.2.2.69.2.Declaration

```
- (OSIStatus) setFilterPosition : (in uint8_t) aFilterPosition;
```

5.2.2.69.3.Parameters

Name	In/Out	Usage
aFilterPosition	In	The filter to position in front of the CCD.

5.2.2.69.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINoFilter`	The camera has no filters.
`QSIInvalid Parameter`	The specified filter position is out of range.

5.2.2.69.5.Notes

5.2.2.70.setFocusOffset:forFilter:

5.2.2.70.1.Semantics

Sets the focus offset adjustment for the specified filter.

The position is 1-relative.

5.2.2.70.2.Declaration

```
- (OSIStatus) setFocusOffset : (in int32_t) aFocusOffset
                    forFilter : (in uint8_t) aFilterPosition;
```

5.2.2.70.3.Parameters

Name	In/Out	Usage
aFocusOffset	In	The filter's focus offset adjustment.
aFilterPosition	In	The position of the filter in question.

Copyright © 2012, Joe Shimkus

5.2.2.70.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINoFilter`	The camera has no filters.
`QSIInvalid Parameter`	The specified filter position is out of range.

5.2.2.70.5.Notes

5.2.2.71.setGain:

5.2.2.71.1.Semantics

Sets the camera's gain.

If successful, the value is stored in the camera's persistent settings.

5.2.2.71.2.Declaration

```
- (QSIStatus) setGain : (in QSICameraGain) aGain;
```

5.2.2.71.3.Parameters

Name	In/Out	Usage
aGain	In	The gain to set.

5.2.2.71.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINotSupported`	The requested functionality is not supported by the camera.

5.2.2.71.5.Notes

5.2.2.72.setLEDEnabled:

5.2.2.72.1.Semantics

Turns the camera's LED status indicator on and off.

If successful, the value is stored in the camera's persistent settings.

Copyright © 2012, Joe Shimkus

5.2.2.72.2.Declaration

```
- (QSIStatus) setLEDEnabled : (in bool) anLEDEnabled;
```

5.2.2.72.3.Parameters

Name	In/Out	Usage
anLEDEnabled	In	YES, the LED status indicator is turned on. NO, the LED status indicator is turned off.

5.2.2.72.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.
QSINotSupported	The requested functionality is not supported by the camera.

5.2.2.72.5.Notes

5.2.2.73.setMainCamera:

5.2.2.73.1.Semantics

Set the camera to serving as either the main camera or a guiding camera.

In order to change the camera from main to guiding, or vice-versa, there must not be an extant connection to the camera.

5.2.2.73.2.Declaration

```
- (QSIStatus) setMainCamera : (in bool) aMainCamera;
```

5.2.2.73.3.Parameters

Name	In/Out	Usage
aMainCamera	In	YES, set the camera to be the main camera. NO, set the camera to be a guiding camera.

5.2.2.73.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSIConnected	There is a connection to the camera.

Copyright © 2012, Joe Shimkus

5.2.2.73.5.Notes

5.2.2.74.setMaxPixelsPerBlock:

5.2.2.74.1.Semantics

Sets the maximum number of pixels the camera will read from the CCD in a single operation.

5.2.2.74.2.Declaration

```
- (QSIStatus) setMaxPixelsPerBlock :
                                    (in int32_t) aMaxPixelsPerBlock;
```

5.2.2.74.3.Parameters

Name	In/Out	Usage
aMaxPixelsPerBlock	In	The maximum number of pixels the camera will read in a single operation from the CCD.

5.2.2.74.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.74.5.Notes

5.2.2.75.setName:forFilter:

5.2.2.75.1.Semantics

Sets the name for the specified filter.

The position is 1-relative.

5.2.2.75.2.Declaration

```
- (QSIStatus) setName : (in NSString *) aName
             forFilter : (in uint8_t) aFilterPosition;
```

5.2.2.75.3.Parameters

Name	In/Out	Usage
aName	In	The filter's name.
aFilterPosition	In	The position of the filter in question.

Copyright © 2012, Joe Shimkus

5.2.2.75.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.
QSINoFilter	The camera has no filters.
QSIInvalid Parameter	The specified filter position is out of range.

5.2.2.75.5.Notes

5.2.2.76.setPreExposureFlush:

5.2.2.76.1.Semantics

Sets the camera's pre-exposure flush setting.

If successful, the value is stored in the camera's persistent settings.

5.2.2.76.2.Declaration

```
- (QSIStatus) setPreExposureFlush :
                (in QSICameraPreExposureFlush) aPreExposureFlush;
```

5.2.2.76.3.Parameters

Name	In/Out	Usage
aPreExposureFlush	In	The pre-exposure flush setting to set.

5.2.2.76.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.
QSINotSupported	The requested functionality is not supported by the camera.

5.2.2.76.5.Notes

5.2.2.77.setReadoutSpeed:

5.2.2.77.1.Semantics

Sets the camera's readout speed.

If successful, the value is stored in the camera's persistent settings.

Copyright © 2012, Joe Shimkus

5.2.2.77.2.Declaration

```
- (QSIStatus) setReadoutSpeed :
                                    (in QSICameraReadoutSpeed) aReadoutSpeed;
```

5.2.2.77.3.Parameters

Name	In/Out	Usage
aReadoutSpeed	Out	The readout speed to which to set the camera.

5.2.2.77.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINotSupported`	The requested functionality is not supported by the camera.

5.2.2.77.5.Notes

5.2.2.78.setRowBinning:

5.2.2.78.1.Semantics

Sets the camera's row binning value.

5.2.2.78.2.Declaration

```
- (QSIStatus) setRowBinning : (in uint16_t) aBinning;
```

5.2.2.78.3.Parameters

Name	In/Out	Usage
aBinning	In	The row binning value to use.

5.2.2.78.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSIInvalid Parameter`	The specified binning value is out of the supported range.

5.2.2.78.5.Notes

Copyright © 2012, Joe Shimkus

5.2.2.79.setShutterPriority:

5.2.2.79.1.Semantics

Set's the camera's shutter priority.

If successful, the value is stored in the camera's persistent settings.

5.2.2.79.2.Declaration

```
- (OSIStatus) setShutterPriority :
                        (in OSICameraShutterPriority) aShutterPriority;
```

5.2.2.79.3.Parameters

Name	In/Out	Usage
aShutterPriority	In	The shutter priority to set.

5.2.2.79.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.
`QSINotSupported`	The requested functionality is not supported by the camera.

5.2.2.79.5.Notes

5.2.2.80.setSoundEnabled:

5.2.2.80.1.Semantics

Turns the camera's sound status indicator on and off.

If successful, the value is stored in the camera's persistent settings.

5.2.2.80.2.Declaration

```
- (OSIStatus) setSoundEnabled : (in bool) aSoundEnabled;
```

5.2.2.80.3.Parameters

Name	In/Out	Usage
aSoundEnabled	In	`YES`, the sound status indicator is turned on. `NO`, the sound status indicator is turned off.

5.2.2.80.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.

Copyright © 2012, Joe Shimkus

Value	Significance
QSINotConnected	There is no connection to the camera.
QSINotSupported	The requested functionality is not supported by the camera.

5.2.2.80.5.Notes

5.2.2.81.supportsAsymmetricBinning:

5.2.2.81.1.Semantics

Returns whether or not the camera supports asymmetric binning.

5.2.2.81.2.Declaration

```
- (QSIStatus) supportsAsymmetricBinning :
                                (out bool *) aSupportsAsymmetricBinning;
```

5.2.2.81.3.Parameters

Name	In/Out	Usage
aSupportsAsymmetric Binning	Out	YES, the camera supports asymmetric binning. NO, the camera does not support asymmetric binning.

5.2.2.81.4.Return Values

Value	Significance
QSISuccess	The method completed successfully.
QSINotConnected	There is no connection to the camera.

5.2.2.81.5.Notes

5.2.2.82.temperature:

5.2.2.82.1.Semantics

Returns the camera's temperature information.

The returned object has been auto-released.

5.2.2.82.2.Declaration

```
- (QSIStatus) temperature :
                          (out QSICameraTemperature * *) aTemperature;
```


Copyright © 2012, Joe Shimkus

5.2.2.82.3.Parameters

Name	In/Out	Usage
aTemperature	Out	The camera's temperature information.

5.2.2.82.4.Return Values

Value	Significance
`QSISuccess`	The method completed successfully.
`QSINotConnected`	There is no connection to the camera.

5.2.2.82.5.Notes

Copyright © 2012, Joe Shimkus

5.3. QSICamera

5.3.1. Semantics

Implements the `QSICamera` interface.

Provides access to camera status and control.

5.3.2. Class APIs

5.3.2.1. simulatedCameras

5.3.2.1.1. Semantics

This method returns an array of cameras which, while not allowing soup-to-nuts testing of QSICamera, allows testing of its higher-level functionality.

More usefully, these simulated cameras allow the development of applications incorporating QSICamera without requiring the actual connection of a camera from QSI.

5.3.2.1.2. Declaration

```
+ (NSArray *) simulatedCameras
```

5.3.2.1.3. Parameters

Name	In/Out	Usage
n/a	n/a	n/a

5.3.2.1.4. Return Values

Value	Significance
`NSArray instance`	Array containing instances of simulated camera objects

5.3.2.1.5. Notes

Copyright © 2012, Joe Shimkus

5.4. QSICameraAdvancedDetails

5.4.1. Semantics

Used to report the advanced details of the camera.

5.4.2. Definition

```
@interface QSICameraAdvancedDetails : QSIObject <NSCopying>
{
@private
  bool                      _antiBloomingEnabled;
  QSICameraAntiBloom        _antiBloomingIndex;

  bool                      _cameraGainEnabled;
  QSICameraGain             _cameraGainIndex;

  bool                      _fanModeEnabled;
  QSICameraFanMode          _fanModeIndex;

  bool                      _ledIndicatorDefault;
  bool                      _ledIndicatorEnabled;

  bool                      _optimizationsEnabled;
  bool                      _optimizeReadoutSpeed;

  bool                      _preExposureFlushEnabled;
  QSICameraPreExposureFlush _preExposureFlushIndex;

  bool                      _showDownloadProgressDefault;
  bool                      _showDownloadProgressEnabled;

  bool                      _shutterPriorityEnabled;
  QSICameraShutterPriority  _shutterPriorityIndex;

  bool                      _soundOnDefault;
  bool                      _soundOnEnabled;
}

@property (nonatomic, assign) bool  _antiBloomingEnabled;
@property (nonatomic, assign) QSICameraAntiBloom
                                    _antiBloomingIndex;

@property (nonatomic, assign) bool  _cameraGainEnabled;
@property (nonatomic, assign) QSICameraGain
                                    _cameraGainIndex;

@property (nonatomic, assign) bool  _fanModeEnabled;
@property (nonatomic, assign) QSICameraFanMode
                                    _fanModeIndex;

@property (nonatomic, assign) bool  _ledIndicatorDefault;
@property (nonatomic, assign) bool  _ledIndicatorEnabled;

@property (nonatomic, assign) bool  _optimizationsEnabled;
```


Copyright © 2012, Joe Shimkus

```
    @property (nonatomic, assign) bool  _optimizeReadoutSpeed;

    @property (nonatomic, assign) bool  _preExposureFlushEnabled;
    @property (nonatomic, assign) OSICameraPreExposureFlush
                                        _preExposureFlushIndex;

    @property (nonatomic, assign) bool  _showDownloadProgressDefault;
    @property (nonatomic, assign) bool  _showDownloadProgressEnabled;

    @property (nonatomic, assign) bool  _shutterPriorityEnabled;
    @property (nonatomic, assign) OSICameraShutterPriority
                                        _shutterPriorityIndex;

    @property (nonatomic, assign) bool  _soundOnDefault;
    @property (nonatomic, assign) bool  _soundOnEnabled;

    @end
```

5.4.3. Notes

Copyright © 2012, Joe Shimkus

5.5. QSICameraAdvancedSettingsParameters

5.5.1. Semantics

Used to set the camera's advanced settings.

5.5.2. Definition

```
@interface QSICameraAdvancedSettingsParameters : QSIObject <NSCopying>
{
@private
  QSICameraAntiBloom        _antiBloomingIndex;
  QSICameraGain             _cameraGainIndex;
  QSICameraFanMode          _fanModeIndex;
  bool                      _ledIndicatorOn;
  bool                      _optimizeReadoutSpeed;
  QSICameraPreExposureFlush _preExposureFlushIndex;
  bool                      _showDownloadProgress;
  QSICameraShutterPriority  _shutterPriorityIndex;
  bool                      _soundOn;
}

@property (nonatomic, assign) QSICameraAntiBloom
                                    _antiBloomingIndex;
@property (nonatomic, assign) QSICameraGain
                                    _cameraGainIndex;
@property (nonatomic, assign) QSICameraFanMode
                                    _fanModeIndex;
@property (nonatomic, assign) bool  _ledIndicatorOn;
@property (nonatomic, assign) bool  _optimizeReadoutSpeed;
@property (nonatomic, assign) QSICameraPreExposureFlush
                                    _preExposureFlushIndex;
@property (nonatomic, assign) bool  _showDownloadProgress;
@property (nonatomic, assign) QSICameraShutterPriority
                                    _shutterPriorityIndex;
@property (nonatomic, assign) bool  _soundOn;

@end
```

5.5.3. Notes

Copyright © 2012, Joe Shimkus

5.6. QSICameraAntiBloom

5.6.1. Semantics

Used to report and set the camera's anti-blooming setting.

5.6.2. Definition

```
typedef enum
{
  antiBloomNormal = 0,
  antiBloomHigh   = 1
} QSICameraAntiBloom;
```

5.6.3. Notes

Copyright © 2012, Joe Shimkus

5.7. QSICameraAutoZero

5.7.1. Semantics

Used to report the auto-zero values of the camera.

5.7.2. Definition

```
@interface QSICameraAutoZero : QSIObject <NSCopying>
{
@private
  uint16_t  _pixelCount;
  bool      _zeroEnabled;
  uint16_t  _zeroLevel;
}

@property (nonatomic, assign) uint16_t  _pixelCount;
@property (nonatomic, assign) bool      _zeroEnabled;
@property (nonatomic, assign) uint16_t  _zeroLevel;

@end
```

5.7.3. Notes

Copyright © 2012, Joe Shimkus

5.8. QSICameraAutoZeroControl

5.8.1. Semantics

Used to report and set the user's auto-zero control parameters.

5.8.2. Definition

```
@interface QSICameraAutoZeroControl : QSIObject <NSCopying>
{
@private
  bool      _autoZeroEnabled;
  int32_t   _autoZeroMaxADU;
  bool      _autoZeroMedian;
  int32_t   _autoZeroSaturationThreshold;
  int32_t   _autoZeroSkipEndPixels;
  int32_t   _autoZeroSkipStartPixels;
}

@property (nonatomic, assign) bool      _autoZeroEnabled;
@property (nonatomic, assign) int32_t   _autoZeroMaxADU;
@property (nonatomic, assign) bool      _autoZeroMedian;
@property (nonatomic, assign) int32_t   _autoZeroSaturationThreshold;
@property (nonatomic, assign) int32_t   _autoZeroSkipEndPixels;
@property (nonatomic, assign) int32_t   _autoZeroSkipStartPixels;

@end
```

5.8.3. Notes

Copyright © 2012, Joe Shimkus

5.9. QSICameraCCDSpecs

5.9.1. Semantics

Used to report the characteristics of the camera's CCD chip.

5.9.2. Definition

```
@interface QSICameraCCDSpecs : QSIObject <NSCopying>
{
@private
  double    _electronsPerADUHigh;
  double    _electronsPerADULow;
  double    _fullWell;
  uint16_t  _maxADU;
  double    _maxExposure; // in seconds
  double    _minExposure; // in seconds
}

@property (nonatomic, assign) double    _electronsPerADUHigh;
@property (nonatomic, assign) double    _electronsPerADULow;
@property (nonatomic, assign) double    _fullWell;
@property (nonatomic, assign) uint16_t  _maxADU;
@property (nonatomic, assign) double    _maxExposure;
@property (nonatomic, assign) double    _minExposure;

@end
```

5.9.3. Notes

Copyright © 2012, Joe Shimkus

5.10.QSICameraCombinedDetails

5.10.1.Semantics

Provides “one stop” access to both the camera details and CCD characteristics.

5.10.2.Definition

```
@interface QSICameraCombinedDetails : QSIObject <NSCopying>
{
@private
  QSICameraCCDSpecs *_ccdSpecs;
  QSICameraDetails  *_details;
}

@property (nonatomic, readonly) QSICameraCCDSpecs *_ccdSpecs;
@property (nonatomic, readonly) QSICameraDetails  *_details;

@end
```

5.10.3.Notes

Copyright © 2012, Joe Shimkus

5.11.QSICameraCoolerState

5.11.1.Semantics

Used to report the camera's cooler's state.

5.11.2.Definition

```
typedef enum
{
  coolerOff,            // Cooler is off
  coolerOn,             // Cooler is on
  coolerAtAmbient,      // Cooler is on and regulating at ambient
                        //    temperature
  coolerGoToAmbient,    // Cooler is on and ramping to ambient
  coolerNoControl,      // Cooler cannot be controlled on this camera
  coolerInitializing,   // Cooler control is initializing
  coolerIncreasing,     // Cooler temperature is going up
  coolerDecreasing,     // Cooler temperature is going down
  coolerBrownout        // Cooler brownout condition
} QSICameraCoolerState;
```

5.11.3.Notes

Copyright © 2012, Joe Shimkus

5.12.QSICameraDetails

5.12.1.Semantics

Used to report the camera details.

5.12.2.Definition

```
@interface QSICameraDetails : QSIObject <NSCopying>
{
@private
  uint16_t  _arrayColumns;
  uint16_t  _arrayRows;
  bool      _hasCamera;
  bool      _hasFilterWheel;
  bool      _hasRelays;
  bool      _hasShutter;
  bool      _hasTemperatureRegulator;
  uint8_t   _maxHorizontalBinning;
  uint8_t   _maxVerticalBinning;
  NSString  *_modelName;
  NSString  *_modelNumber;
  uint8_t   _numberOfFilters;
  uint16_t  _numberOfRowsPerBlock;
  bool      _perBlockControl;
  double    _pixelHeight;
  double    _pixelWidth;
  bool      _powerOfTwoBinning;
  NSString  *_serialNumber;
  bool      _supportsAsymmetricBinning;
}

@property (nonatomic, assign) uint16_t  _arrayColumns;
@property (nonatomic, assign) uint16_t  _arrayRows;
@property (nonatomic, assign) bool      _hasCamera;
@property (nonatomic, assign) bool      _hasFilterWheel;
@property (nonatomic, assign) bool      _hasRelays;
@property (nonatomic, assign) bool      _hasShutter;
@property (nonatomic, assign) bool      _hasTemperatureRegulator;
@property (nonatomic, assign) uint8_t   _maxHorizontalBinning;
@property (nonatomic, assign) uint8_t   _maxVerticalBinning;
@property (nonatomic, retain) NSString  *_modelName;
@property (nonatomic, retain) NSString  *_modelNumber;
@property (nonatomic, assign) uint8_t   _numberOfFilters;
@property (nonatomic, assign) uint16_t  _numberOfRowsPerBlock;
@property (nonatomic, assign) bool      _perBlockControl;
@property (nonatomic, assign) double    _pixelHeight;
@property (nonatomic, assign) double    _pixelWidth;
@property (nonatomic, assign) bool      _powerOfTwoBinning;
@property (nonatomic, retain) NSString  *_serialNumber;
@property (nonatomic, assign) bool      _supportsAsymmetricBinning;

@end
```


Copyright © 2012, Joe Shimkus

5.12.3.Notes

Copyright © 2012, Joe Shimkus

5.13.QSICameraExposureParameters

5.13.1.Semantics

Used to set the parameters for an exposure.

5.13.2.Definition

```
@interface QSICameraExposureParameters : QSIObject <NSCopying>
{
@private
  uint16_t  _columnBinning;
  uint32_t  _duration;
  uint8_t   _durationUsec;
  bool      _fastReadout;
  uint16_t  _height;
  bool      _holdShutterOpen;
  bool      _openShutter;
  bool      _probeForImplemented;
  uint16_t  _repeatCount;
  uint16_t  _rowBinning;
  uint16_t  _startingColumn;
  uint16_t  _startingRow;
  bool      _strobeShutterOutput;
  bool      _useExternalTrigger;
  uint16_t  _width;
}

@property (nonatomic, assign) uint16_t  _columnBinning;
@property (nonatomic, assign) uint32_t  _duration;
@property (nonatomic, assign) uint8_t   _durationUsec;
@property (nonatomic, assign) bool      _fastReadout;
@property (nonatomic, assign) uint16_t  _height;
@property (nonatomic, assign) bool      _holdShutterOpen;
@property (nonatomic, assign) bool      _openShutter;
@property (nonatomic, assign) bool      _probeForImplemented;
@property (nonatomic, assign) uint16_t  _repeatCount;
@property (nonatomic, assign) uint16_t  _rowBinning;
@property (nonatomic, assign) uint16_t  _startingColumn;
@property (nonatomic, assign) uint16_t  _startingRow;
@property (nonatomic, assign) bool      _strobeShutterOutput;
@property (nonatomic, assign) bool      _useExternalTrigger;
@property (nonatomic, assign) uint16_t  _width;

@end
```

5.13.3.Notes

Copyright © 2012, Joe Shimkus

5.14.QSICameraExposureRequest

5.14.1.Semantics

Used to set all the camera's parameters that impact exposure.

5.14.2.Definition

```
@interface QSICameraExposureRequest : QSIObject
{
@private
  QSICameraAntiBloom        _antiBlooming;
  QSICameraGain             _cameraGain;
  QSICameraPreExposureFlush _preExposureFlush;
  QSICameraReadoutSpeed     _readoutSpeed;
  QSICameraShutterPriority  _shutterPriority;

  uint16_t                  _columnBinning;
  uint32_t                  _duration;
  uint16_t                  _height;
  bool                      _openShutter;
  uint16_t                  _rowBinning;
  uint16_t                  _startingColumn;
  uint16_t                  _startingRow;
  uint16_t                  _width;

  uint8_t                   _filterPosition;
}

@property (assign)  QSICameraAntiBloom        _antiBlooming;
@property (assign)  QSICameraGain             _cameraGain;
@property (assign)  QSICameraPreExposureFlush _preExposureFlush;
@property (assign)  QSICameraReadoutSpeed     _readoutSpeed;
@property (assign)  QSICameraShutterPriority  _shutterPriority;

@property (assign)  uint16_t                  _columnBinning;
@property (assign)  uint32_t                  _duration;
@property (assign)  uint16_t                  _height;
@property (assign)  bool                      _openShutter;
@property (assign)  uint16_t                  _rowBinning;
@property (assign)  uint16_t                  _startingColumn;
@property (assign)  uint16_t                  _startingRow;
@property (assign)  uint16_t                  _width;

@property (assign)  uint8_t                   _filterPosition;
@end
```

5.14.3.Notes

Copyright © 2012, Joe Shimkus

5.15.QSICameraFanMode

5.15.1.Semantics

Used to report and set the camera's fan operating mode.

5.15.2.Definition

```
typedef enum
{
  fanOff    = 0,
  fanQuiet  = 1,
  fanFull   = 2
} QSICameraFanMode;
```

5.15.3.Notes

Copyright © 2012, Joe Shimkus

5.16.QSICameraGain

5.16.1.Semantics

Used to report and set the camera's gain setting.

5.16.2.Definition

```
typedef enum
{
  gainHigh  = 0,
  gainLow   = 1
} QSICameraGain;
```

5.16.3.Notes

Copyright © 2012, Joe Shimkus

5.17.QSICameraGuiding

5.17.1.Semantics

Used to control the camera guiding operation.

5.17.2.Definition

```
typedef enum
{
  guidingAbort    = 0,
  guidingNorth    = 1,
  guidingSouth    = 2,
  guidingEast     = 3,
  guidingWest     = 4
} QSICameraGuiding;
```

5.17.3.Notes

Copyright © 2012, Joe Shimkus

5.18.QSICameraPreExposureFlush

5.18.1.Semantics

Used to report and set the camera's pre-exposure flush setting.

5.18.2.Definition

```
typedef enum
{
  preExposureFlushNone              = 0,
  preExposureFlushModest            = 1,
  preExposureFlushNormal            = 2,
  preExposureFlushAggressive        = 3,
  preExposureFlushVeryAggressive    = 4,
} QSICameraPreExposureFlush;
```

5.18.3.Notes

Copyright © 2012, Joe Shimkus

5.19.QSICameraReadoutSpeed

5.19.1.Semantics

Used to report and set the camera's readout speed.

5.19.2.Definition

```
typedef enum
{
  readoutSpeedHighQuality     = 0,
  readoutSpeedFast            = 1
} QSICameraReadoutSpeed;
```

5.19.3.Notes

Copyright © 2012, Joe Shimkus

5.20.QSICameraShutterPriority

5.20.1.Semantics

Used to report and set the camera's shutter priority setting.

5.20.2.Definition

```
typedef enum
{
  shutterPriorityMechanical   = 0,
  shutterPriorityElectronic   = 1
} QSICameraShutterPriority;
```

5.20.3.Notes

Copyright © 2012, Joe Shimkus

5.21. QSICameraState

5.21.1. Semantics

Used to report the camera's current state.

5.21.2. Definition

```
typedef enum
{
  stateIdle             = 0,  // Available to start exposure
  stateWaiting          = 1,  // Exposure started but waiting
  stateExposing         = 2,  // Exposure in progress
  stateReading          = 3,  // Reading CCD
  stateDownloading      = 4,  // Downloading data to computer
  stateError            = 5   // Error, no further operations tenable
} QSICameraState;
```

5.21.3. Notes

Copyright © 2012, Joe Shimkus

5.22. QSICameraTemperature

5.22.1. Semantics

Used to report the camera's temperature data.

5.22.2. Definition

```
@interface QSICameraTemperature : QSIObject <NSCopying>
{
@private
  double                _ambientTemperature;
  uint16_t              _coolerPower;
  QSICameraCoolerState  _coolerState;
  double                _coolerTemperature;
}

@property (nonatomic, assign) double      _ambientTemperature;
@property (nonatomic, assign) uint16_t    _coolerPower;
@property (nonatomic, assign) QSICameraCoolerState
                                          _coolerState;
@property (nonatomic, assign) double      _coolerTemperature;

@end
```

5.22.3. Notes

Copyright © 2012, Joe Shimkus

5.23. QSICameraTemperatureParameters

5.23.1. Semantics

Used to control the camera's temperature.

5.23.2. Definition

```
@interface QSICameraTemperatureParameters : QSIObject <NSCopying>
{
@private
  bool    _coolerOn;
  bool    _goToAmbient;
  double  _setPoint;
}

@property (nonatomic, assign) bool    _coolerOn;
@property (nonatomic, assign) bool    _goToAmbient;
@property (nonatomic, assign) double  _setPoint;

@end
```

5.23.3. Notes

Copyright © 2012, Joe Shimkus

5.24.QSIFilter

5.24.1.Semantics

5.24.2.APIs

5.24.2.1.focusOffset

5.24.2.1.1.Semantics

Returns the focus adjustment for the filter.

5.24.2.1.2.Declaration

```
- (int32_t) focusOffset;
```

5.24.2.1.3.Parameters

Name	In/Out	Usage
n/a	n/a	n/a

5.24.2.1.4.Return Values

Value	Significance
`any`	The filter's focus offset

5.24.2.1.5.Notes

5.24.2.2.name

5.24.2.2.1.Semantics

Returns the name of the filter.

5.24.2.2.2.Declaration

```
- (NSString *) name;
```

5.24.2.2.3.Parameters

Name	In/Out	Usage
n/a	n/a	n/a

5.24.2.2.4.Return Values

Value	Significance
`NSString *`	The filter's name

5.24.2.2.5.Notes

Copyright © 2012, Joe Shimkus

5.24.2.3.setFocusOffset

5.24.2.3.1.Semantics

Sets the focus adjustment for the filter.

5.24.2.3.2.Declaration

```
- (void) setFocusOffset : (int32_t) aFocusOffset;
```

5.24.2.3.3.Parameters

Name	In/Out	Usage
aFocusOffset	In	The value to set for the filter's offset

5.24.2.3.4.Return Values

Value	Significance
`n/a`	n/a

5.24.2.3.5.Notes

5.24.2.4.setName

5.24.2.4.1.Semantics

Sets the name of the filter.

5.24.2.4.2.Declaration

```
- (void) setName : (NSString *) aName;
```

5.24.2.4.3.Parameters

Name	In/Out	Usage
aName	In	The value to set for the filter's name

5.24.2.4.4.Return Values

Value	Significance
`n/a`	n/a

5.24.2.4.5.Notes

5.24.2.5.setTrim

5.24.2.5.1.Semantics

Sets the trim adjustment for the filter.

Copyright © 2012, Joe Shimkus

5.24.2.5.2.Declaration

```
- (void) setTrim : (int16_t) aTrim;
```

5.24.2.5.3.Parameters

Name	In/Out	Usage
aTrim	In	The value to set for the filter's trim adjustment

5.24.2.5.4.Return Values

Value	Significance
`n/a`	n/a

5.24.2.5.5.Notes

5.24.2.6.trim

5.24.2.6.1.Semantics

Returns the trim adjustment for the filter.

5.24.2.6.2.Declaration

```
- (int16_t) trim;
```

5.24.2.6.3.Parameters

Name	In/Out	Usage
n/a	n/a	n/a

5.24.2.6.4.Return Values

Value	Significance
`any`	The filter's trim adjustment

5.24.2.6.5.Notes

Copyright © 2012, Joe Shimkus