Cary ActiveX Control

Version 3.03

Programmer's reference guide for the CaryXControl.ocx (beta)

1. Introduction

The CaryXControl was developed to allow easier access than the DLL (Cary32.dll) to a Cary instrument from programs other than the existing Varian applications. It features most of the DLL functions as methods, properties and events of the control.

The ActiveX control uses buffered communications with the instrument. Setting properties does not necessarily immediately change the state of the instrument, for this SetupInst or SubSetup should be used. This buffer is referred to as the *instrument record*.

1.1.1. Unused Methods

There are some methods that are generated by the ActiveX compiler that should be ignored by the end user as they have no effect on the Cary instrument. These are DrawTextBiDiModeFlagsReadingOnly, InitiateAction, SetSubComponent, UseRightToLeftReading and UseRightToLeftScrollBar.

2. Functionality

2.1. Properties

2.1.1. Averaging As Long

The UV/VIS averaging.

This property is deprecated, for clarity use AveragingUVVIS As Long instead.

2.1.2. AveragingUVVIS As Long

The UV/VIS averaging time (ms).

The time period, in milliseconds, over which the instruments collects data at a single point before returning the averaged data. This gets converted internally to a number of chopper cycles (1/80s for Cary 50 and 1/30s for other Cary instruments).

2.1.3. AveragingNIR As Long

The NIR averaging time (ms).

The time period, in milliseconds, over which the instruments collects data at a single point before returning the averaged data. This gets converted internally to a number of chopper cycles (1/80s for Cary 50 and 1/30s for other Cary instruments).

2.1.4. BeamMode As TBeamMode

```
Public Enum TBeamMode
      beamSingleFront = 0
      beamSingleRear = 1
      beamDoubleFixedSlit = 2
      beamDoubleFixedEnergy = 3
      beamDoubleAutoSelect = 4
      beamTestMode = 5
      beamDualSingle = 6
End Enum
```

Controls the instrument's beam arrangement.

2.1.5. CollectXMode As TCollectXMode

```
Public Enum TCollectXMode
      collectXWavelength = 0
      collectXTime = 1
      collectXPeaking = 2
      collectXWavelengthSNR = 3
      collectXTimeSNR = 4
      collectXIdle = 5
      collectXPmvPeak = 6
      collectXTCalibration = 7
      collectXWavenumber = 8
      collectXWavenumberSNR = 9
```

```
End Enum
```

Specifies the collection mode Collect is called.

2.1.6. DetectorChangeoverWavelength As Single

This is the wavelength at which the UV/VIS or NIR detectors are selected.

2.1.7. Grating As TGratingMode

```
Public Enum TGratingMode
     gratAuto = 0
     gratUVVis = 1
     gratNIR = 2
End Enum
```

Controls the behaviour of the grating, the default is gratAuto.

2.1.8. GratingChangeoverWavelength As Single

The wavelength (nm) at which the grating changes, the default value is 900.

2.1.9. Interval As Single

This is the UV/VIS wavelength interval.

This function is deprecated, for clarity use IntervalUVVIS As Single instead.

2.1.10. IntervalUVVIS As Single

The UVVIS interval.

The wavelength interval (nm) between consecutive points in a wavelength scan, the default value is -1.

2.1.11. IntervalNIR As Single

The NIR interval.

The wavelength interval (nm) between consecutive points in a wavelength scan, the default value is 0.

2.1.12. LampThird As Boolean

Controls whether the third source lamp (eg NIR) is on or off.

2.1.13. LampUVOn As Boolean

Controls whether the UV source lamp is on or off.

2.1.14. LampVISOn As Boolean

Controls whether the VIS source lamp is on or off.

2.1.15. Model As Long

```
modelOffline = -1
modelCary50 = 0
modelCary100 = 1
modelCary200 = 2
modelCary300 = 3
modelCary400 = 4
modelCary500 = 5
modelCary4000 = 6
modelCary5000 = 7
modelCary6000i = 8
modelDeepUV = 9
```

This property contains the model number. The value is -1 before the control has connected to the instrument, or if the instrument is offline.

2.1.16. SBW As Single

The UV/VIS spectral bandwidth or slit width.

This function is deprecated, use SlitUVVIS instead.

2.1.17. ScanStart As Single

The starting wavelength for a scan. For wavelength scans ScanStart should be in nm, for wavenumber scans the start should be in wavenumbers. The default value is 500.

2.1.18. ScanStop As Single

The stopping wavelength for a scan. For wavelength scans ScanStop should be in nm, for wavenumber scans the stop should be in wavenumbers. The default value is 400.

2.1.19. ShowInfo As Boolean

Determines if the ActiveX control should display information on the form. Version information, connection status and license information is displayed. This option is ignored before the control is registered (see 4.2 Registration).

2.1.20. SignalNoise As Single

The target signal to noise ratio. Default is 10,000.

2.1.21. SlitNIR As Single

This is the width (nm) of the slit for an NIR scan, the default value is 4.

2.1.22. SlitUVVIS As Single

This is the width (nm) of the slit for a UV/Vis scan, the default value is 1.

2.1.23. SourceChangeoverWavelength

The wavelength (nm) at which the source changes over.

2.1.24. Source As TSourceMode

```
Public Enum TSourceMode
      SrcAuto = 0
      SrcUV = 1
      SrcVis = 2
      SrcThird = 3
End Enum
```

The mode that controls the instrument source selection.

2.2. Methods

Many functions return a simple Boolean (True or False) indicating only that the call succeeded.

2.2.1. Sub AccWaitNotBusy

Waits until the accessory board is no longer busy.

2.2.2. Function AccyReset(item As Long) As Boolean

Resets a device on the accessory board as specified by ***item***.

Returns ***True*** if successful, else ***False***.

2.2.3. Function AccyStatusString(StatusCode As Long) As String

Returns the English of a set of status code flags, such as that returned by OnAccyStatus.

2.2.4. Function ARampAcc(Device As Long, Rate As Long) As Boolean

This command allows you to drive an accessory at a given rate from the current position to a specified limit or until the STOP command is executed. This command will listen for the "data point acquired" signal from the instrument before moving the accessory. This is the distinct from RAMPACC command.

Returns ***True*** if the accessory was driven to the correct position otherwise it returns ***False***.

2.2.5. Function AVASRASet(SampleAngle As Single, DetectorAngle As Single) As Boolean

Sets the angle of the AVASRA (UMA) accessory sample and detector angles (values as per WinUV software).

2.2.6. Function BusyInst As Boolean

Indicates if the instrument is busy setting itself up or collecting.

2.2.7. Function Calibrate(item As Long) As Boolean

This command is used to calibrate various CARY 4/5 functions. Note that some of the calibrations require NIR capability and so apply only to the Cary 5. The *item* can be one of the following:

Item	Meaning	Description
0	Calibrate signal processing	Calibrates the programmable gain amplifiers on the photomultiplier tube (PMT)
1	Calibrate wavelength UV/Vis	Uses the deuterium lamp to calibrate the instrument at 656.1 nm, 486.0 nm, and 0.0 nm. (This calibration is performed by the Calibrate wl command on the Instrument Test Page.)
2	Calibrate wavelength NIR	Uses the deuterium lamp to calibrate the instrument at 2624.4 nm, 1312.2 nm, and 0.0 nm. (This calibration is performed by the Calibrate wl command on the Instrument Test Page.)
3	Calibrate NIR memory 1	This sets up the instrument to compare the NIR zero error over the wavelength range. It is used in conjunction with the Calibrate(10) command.
4	Calibrate NIR linearity 1	This sets up the instrument to compare the NIR detector to the PMT detector. It is used in conjunction with the Calibrate(5) command.
5	Calibrate NIR linearity 2	This compares the NIR detector with the PMT detector. It is used in conjunction with the Calibrate(4) command. A 50% transmission filter should be placed in the sample compartment before issuing a Calibrate(5) command.
6	Check wavelength UV/Vis	This checks the calibration of the monochromator for the UV grating at 656.1 nm, 486.0 nm and 0.0 nm using the spectrometers UV (deuterium) lamp as the source. The values for the wavelength offsets are stored internally. This check is done by the Start command on the Instrument Test Page.
7	Check wavelength NIR	This checks the calibration of the monochromator for the NIR grating at 2624.0 nm, 1944.0 nm and 0.0 nm using the spectrometers' UV (deuterium) lamp as the source. The values for the wavelength offsets are stored internally. This check is done by the Start command on the Instrument Test Page.
8	Calibrate PM zero 1	This closes the shutter and calibrates the photomultiplier zero.
9	Calibrate PM zero 2	This opens the shutter and calibrates the photomultiplier zero.
10	Calibrate NIR memory 2	This corrects the NIR zero error over the wavelength range. It is used in conjunction with the Calibrate(3) command.

Note: The results of a CALIBRATE command are stored in instrument memory until another CALIBRATE command is issued or the Start or Calibrate wl commands are used. This memory is maintained even when the spectrophotometer is turned off.

2.2.8. Function CaryRead(var x As Single, y As Single, z As Single) As Boolean

Reads and removes single data values from the instrument's internal buffer.

x receives the abscissa data (eg wavelength).

y receives the front beam ordinate data.

z receives the rear beam ordinate data.

Returns ***True*** if data was retrieved, else ***False***.

2.2.9. Sub CheckAbort

Checks the Abort flag.

2.2.10. Sub Collect

Starts the instrument collecting data. Data will be returned in the **OnData** event. Same as StartInst except this method also calls StopInst and SetupInst internally before StartInst.

2.2.11. Function ConnectInstrument As Long

This function must be called before the data can be set in the instrument and collected. It connects the ActiveX component with the instrument.

It returns a value greater than zero if it succeeds.

2.2.12. Function Disconnect As Boolean

When your program is finished with the instrument you should call this function. It returns ***True*** if it successfully disconnects.

2.2.13. Function DisplaySPSStatusMessage(StatusCode As Long) As String

Returns the SPS status as an English string, as returned in Event OnSPSStatus(StatusCode As Long). It only returns "Cary SPS ready" or "SPS offline".

2.2.14. Function DriveAcc(Device As Long, Condition As Long, Device2 As Long, Value As Long, State As Long) As Boolean

This command allows you to drive an accessory at a given rate from the current position to a specified limit or until the STOP command is executed.

Device is the device number on the accessory board.

Condition is < 0 for reverse, > 0 for forward.

Device2 is the device whose condition is to be monitored.

Value is the trigger value.

State is either:

1 : detect device2 state > value

2 : detect device2 state < value

Returns ***True*** if successful, else ***False***.

2.2.15. Function ErrorString(ErrorCode As Long) As String

Returns an English description of an error code (such as returned by the Event OnCaryError(ErrorCode As Long) event).

2.2.16. Sub ForceConnect

Tries to forcibly connect to the instrument, overriding any other applications that may be connected.

This is no longer necessary with current drivers.

2.2.17. Function GetAcc(Device As Long) As Single

Returns the current value of an accessory device.

2.2.18. Sub GetDefaultSetup

Loads the default instrument parameters into the ActiveX control's instrument record. The values are determined by the type of instrument.

2.2.19. Function GoToCellAcc(CellNum As Long) As Boolean

Sets the cell changer (if present) to the appropriate cell.

2.2.20. Function GotoWavelength(wavelength As Single) As Boolean

Sends the instrument to go to the specified wavelength. This is the same as using InstSet with RESP_GOTO_WAVELENGTH and then SetupInst to send all instrument records to the instrument.

2.2.21. Function InstGet(item longint) As Single

This returns values from the instrument record. See Appendix 4.5 Get and Set ID's for a list of possible parameters.

2.2.22. Function InstSet(item longint; value As Single) As Boolean

This directly sets values in the instrument record. Use SetupInst or SubSetup to send these to the instrument. See Appendix 4.5 Get and Set ID's for a list of possible parameters.

2.2.23. Sub InstWaitNotBusy

Waits until the instrument is no longer busy.

2.2.24. Sub InstWaitNotScanning

Waits until the instrument is no longer scanning.

2.2.25. Function LimitAcc(Device As Long, LowValue As Long, HighValue As Long) As Boolean

Sets the upper and lower limits on an accessory. This command would normally be used to set limits for a stepper motor.

2.2.26. Function MeasureAcc(Device As Long) As Single

Reads the value of an accessory. This differs from GetAcc as this actually requests a new reading from the device, whereas GetAcc will just return the last requested value.

2.2.27. Function MonitorAcc(Dev1 As Long, Dev2 As Long, Dev3 As Long, Dev4 As Long, TimeVal As Long) As Boolean

Allows you to automatically update the database periodically from the accessory controller.

2.2.28. Function PolarizerSet(PolarizerAngle As Single) As Boolean;

Sets the angle of the polarizer – the value should be from 0 to 360 (degrees).

2.2.29. Function RampAcc(Device As Long, Rate As Long) As Boolean

Drives an accessory at a given rate from the current position to a specified limit or until the STOP command is executed.

2.2.30. Function Read(Wavelength As Single) As Single

Drives the instrument to the specified wavelength and return a value that is the ordinate at that wavelength. This Ordinate value will already be translated into the units of the current Ordinate mode.

2.2.31. Function RequestPolarizerCount As Boolean

The information will be returned as a status event.

2.2.32. Function RequestUMAVersion As Boolean

The version will be returned as a status event.

2.2.33. Function ResetSlide As Boolean

Resets the Sample Transport to the home position.

2.2.34. Function ResetVasra As Boolean

Resets the Cary 4/5 Variable Angle Specular Reflectance accessory to the home position.

2.2.35. Function ResetUMA As Boolean

Resets the position of the UMA (AVASRA) to the home position.

2.2.36. Function ScanningInst As Boolean

Indicates if a scan is active.

2.2.37. Sub SetAbort(state As Boolean)

Sets the abort flag to ***state***.

2.2.38. Function SetAcc(Device As Long; Value As Single) As Boolean

Sets an accessory device to a given value.

2.2.39. Function SetRba(Value As Single) As Boolean

Moves the Rear Beam Attenuator to a new position. Similar to SETACC except SETRBA knows to use device number 2 and the value parameter is in degrees (not motor steps).

2.2.40. Function SetupInst As Boolean

This command sends all the database parameters to the hardware and sets the hardware up.

2.2.41. Sub StartInst

Starts the instrument collecting data. Data will be returned in the **OnData** event. This is a low level call and the Collect method should be used in most cases.

2.2.42. Function StatusString(StatusCode As Long) As String

Returns the status in English from the provided status flags, such as that from Event OnCaryStatus(StatusCode As Long). The following table lists the status codes. Note the StatusCode is a set of binary flags. If none of the flags are defined it returns “Cary ready”.

Flag	Description
00000001	Busy
00000002	Resetting
00000004	Collecting
00000008	Slewing
00000010	Driving slit
00000020	Calibrating PGA
00000040	Waiting for user
00000100	Lighting UV lamp
00000200	Lighting VIS lamp
00000400	Changing grating
00000800	Changing source
00001000	Changing filter
00002000	Cary offline

00010000	Changing detector
00020000	Init wavelength
00040000	Init mono
00080000	Init slit
00100000	Cal wavelength
00200000	Cal PM zero
00800000	SNR timeout
01000000	Cal NIR detector
02000000	Lid open
04000000	Over heat
08000000	Waiting for chopper
10000000	Data over-range
20000000	Data under-range

2.2.43. Function StopAcc As Boolean

Stops the current accessory operations.

2.2.44. Function StopInst As Boolean

Stops a collection.

2.2.45. Function SubSetup(item As Long) As Boolean

Send a single parameter to the instrument where item is one of the following:

Name	Value
SUB_INIT	1
SUB_BEAM_MODE	2
SUB_GAIN	3
SUB_SLITS	4
SUB_CYCLES	5
SUB_SLIT_HEIGHT	6
SUB_REF_LEVELS	7
SUB_GRAT_WL	8
SUB_DET_WL	9
SUB_DETECTOR	10
SUB_PGA	11
SUB_SOURCE_WL	12
SUB_SOURCE	13
SUB_FILTER	14
SUB_SNR	15
SUB_WN_INT	16
SUB_GOTO_WL	17
SUB_GRATING	18

SUB_SOURCE_POWER	19
SUB_SHUTTERS	20
SUB_WRITE	23
SUB_COLLECT	25
SUB_STOP	26
SUB_CALIBRATE	27
SUB_PROM_MODEL	28
SUB_STATUS	29

2.2.46. Function Version As Single

Returns the version of the ActiveX control.

2.2.47. Function VersionComms As Single

Returns the version of the communications DLL (cary32.dll). This may not return a valid value with some drivers.

2.3. Events

2.3.1. Event OnAccyData(x, y, z As Long)

Occurs when an accessory triggers and event. The first parameter is the accessory ID that caused the event. For example, the foot switch is 52.

This event occurs when the accessory device returns data. See Event OnAccyError(ErrorCode As Long).

2.3.2. Event OnAccyError(ErrorCode As Long)

This event is raised when an accessory error occurs.

2.3.3. Event OnAccyStatus(StatusCode As Long)

This event occurs when the accessory status changes. Use Function AccyStatusString(StatusCode As Long) As String to get an English description.

2.3.4. Event OnCaryData(wl, t1, t2 As Single)

This event is raised when data is received from the Cary instrument.

wl is the wavelength.

t1 is the ordinate value of the front beam (in T).

t2 is the ordinate value of the rear beam (in T).

There is also an Accessory and SPS data event.

2.3.5. Event OnCaryError(ErrorCode As Long)

Is triggered when an error in the Cary occurs. Use Function ErrorString(ErrorCode As Long) As String for an English description of the error.

2.3.6. Event OnCaryStatus(StatusCode As Long)

Occurs when a status message is sent from the Cary. Use Function StatusString(StatusCode As Long) As String to return an English description.

2.3.7. Event OnCaryValue(Response As Long, Value As Long)

Occurs when values in the Cary change. For example, when Response is 79 (RESP_GOTO_WAVELENGTH) the Value will contain the current wavelength.

2.3.8. Event OnOnOff(InstrumentIsOn As Boolean)

Occurs when the instrument is turned on or off. The parameter indicates the new state.

2.3.9. Event OnSPSData(wl, t1, t2 As Single)

This event occurs when the SPS device returns data. See Event OnCaryData(wl, t1, t2 As Single).

2.3.10. Event OnSPSError(ErrorCode As Long)

This event is raised when an SPS error occurs.

2.3.11. Event OnSPSStatus(StatusCode As Long)

This event occurs when the SPS status changes. Use Function DisplaySPSStatusMessage(StatusCode As Long) As String to get an English description of the status code.

3. Using the Control

The CaryXControl was designed to simplify the control of Cary instruments from custom software.

Note that only one program can connect to an instrument at any time. Calling ConnectInstrument will try to forcibly disconnect any other application that may be accessing the instrument.

Implementation is dependent on the compiler and development environment and you should consult documentation for that on how to use or import an ActiveX control.

Be aware of possible different versions of cary32.dll on a system (for example, in C:\Windows\System32 and C:\Program Files\Agilent\Cary WinUV). It may be best to distribute a specific cary32.dll with a project executable.

3.1. ActiveX Control Registration

ActiveX controls need to be "registered" within Windows for them to be accessible. Some development environments (like Delphi) allow you to register and unregister them via the interface.

To register (or unregister) the control, call regsvr32.exe in administrator mode. This could be done by an installer or within the program itself (it would need to be run as Administrator the first time). The following can be used from the command line and display a dialog of the result – add an additional "/s" parameter for silent execution (no dialog).

3.1.1. Register Control

regsvr32.exe *[path]*CaryXControl.ocx

3.1.2. Unregister Control

regsvr32.exe /u *[path]*CaryXControl.ocx

4. Appendices

4.1. System Requirements

A computer running Microsoft® Windows® with a base installation of Cary WinUV software (drivers). This version has been tested only with 32-bit Windows 7 but it should also work with other versions as per Agilent Cary compatibility.

4.2. Registration

For the ActiveX control to be used indefinitely it needs to be registered with Startek. After first time use there is a 30 day trial period during which you should acquire the registration information by contacting Startek. For registration and during the trial period any program using the control must be run as Administator.

Please supply your company and department names when registering.

4.3. Contact Information

ABN 21 074 816 336

Startek Technology Pty Ltd
8 / 18 Floriston Rd
Boronia VIC 3155
AUSTRALIA

http://www.star-tek.com.au/
admin@star-tek.com.au

Telephone: +61 3 9761 3880
Fax: +61 3 9761 3811

For bugs and suggestions email us on: caryx.control@star-tek.com.au

4.4. Version History

Version	Date Changes	Description
3.03	Jan 2016	
		Reimplemented some initialisation that had been removed since v2.51 but caused issues, particularly with Cary ‘000 series instruments.
		Changed AVASRASet to accept angles in degrees (as a Single type) in line with WinUV software.
		Added functions RequestUMAVersion and RequestPolarizerCount.
3.01	Jan 2016	
		Added property CollectXMode and sets this mode in the instrument on a call to Collect to fix problems with Cary ‘000 series instruments not responding.
3.00	Dec 2015	
		Added methods AVASRASet, PolarizerSet and ResetUMA

2.51 July 2005

Averaging, Interval & SBW/SlitWidth now use UV/VIS values

In the previous version these properties returned their UV/VIS values but set both UV/VIS and NIR values. When used as an ActiveX this resulted in both also being set when the new individual properties were set.

Changed CaryValueIn to return integer data.

Previously this event returned Long integer values that were wrong because of the way data is sometimes returned as a Single at a lower level. The ActiveX now converts this value to a Long before being passed to the user.

No longer auto-connects

Previous versions automatically tried to connect to an instrument immediately the control was placed on a form. This is no longer the case and programmers will have to manually connect using the ConnectInstrument method. This overcomes some issues like the control connecting to the instrument at design time when it usually isn't needed.

2.50 May 2005

Separated Averaging, Interval & SlitWidth

These properties have been separated for UV/VIS and NIR.

Added DetectorChangeover

This property has been included to allow the setting of the detector changeover wavelength

2.49 April 2004

Improved Cary message handling

2.48 March 2003

Added source changeover & UV/Vis lamps

2.47 August 2002

Included message handling as required by new driver

2.46 January 2002

Added Attached, VersionComms functions

2.45 2001

Changed chopper cycles for Cary 50 to 1/80

Added Version function

2.44 June 2001

2.41 May 2001

2.40 April 2001

2.31 March 2001

2.20 July 2000

4.5. Get and Set ID's

Below is a list of parameter values that can be used with InstSet and InstGet. The actual values are stored in the instrument record. Some can be sent to the instrument using SubSetup or SetupInst.

RESP_ABSCISSA_INTERVAL	0
RESP_ABSCISSA_START	1
RESP_BEAM_MODE	2
RESP_COLLECT_PENDING	3
RESP_DETECTOR	4
RESP_DETECTOR_CHANGEOVER_WAVELENGTH	5
RESP_FILTER	6
RESP_GAIN	7
RESP_GRATING	8
RESP_GRATING_CHANGEOVER_WAVELENGTH	9
RESP_LAST_TRANSMISSION_VALUE	10
RESP_LAST_PMV_GAIN_INDEX	11
RESP_LAST_SLIT_WIDTH	12
RESP_LAST_REART_VALUE	13
RESP_MODEL	14
RESP_NIR_AVERAGING	15
RESP_NIR_DETECTOR_CALIBRATION	16
RESP_NIR_INTERVAL	22
RESP_NIR_PEAK_WAVELENGTHS	23
RESP_NIR_REF_LEVEL	26
RESP_NIR_SLIT_WIDTH	27
RESP_NIR_WAVELENGTH_CALIB_A	28
RESP_NIR_WAVELENGTH_CALIB_B	29
RESP_NIR_WAVELENGTH_CALIB_C	30
RESP_NIR_WAVENUMBER	31
RESP_PGA_GAIN_INDICES_FRONT	32
RESP_PGA_GAIN_INDICES_REAR	33
RESP_PGA_GAIN_INDICES_DARK	34
RESP_PGA_MODE	35
RESP_SHUTTER_FRONT	36
RESP_SHUTTERS_OPEN_CLOSED_FRONT	37
RESP_SHUTTERS_OPEN_CLOSED_REAR	38
RESP_SHUTTER_REAR	39
RESP_SIG_PROCC_CALIB_OFFSET	40
RESP_SIG_PROCC_CALIB_GAIN	41
RESP_SLIT_HEIGHT	49
RESP_SOURCE	50
RESP_SOURCE_CHANGEOVER_WAVELENGTH	51
RESP_SOURCE_ON_OFF_UV	52
RESP_SOURCE_ON_OFF_VIS	53
RESP_SOURCE_ON_OFF_THIRD	54
RESP_STATUS	55
RESP_TARGET_SIGNAL_NOISE	56
RESP_UVVIS_AVERAGING	57
RESP_UVVIS_INTERVAL	58
RESP_UVVIS_PEAK_WAVELENGTHS	59
RESP_UVVIS_REF_LEVEL	62

RESP_UVVIS_SLIT_WIDTH	63
RESP_UVVIS_WAVELENGTH_CALIB_A	64
RESP_UVVIS_WAVELENGTH_CALIB_B	65
RESP_UVVIS_WAVELENGTH_CALIB_C	66
RESP_UVVIS_WAVENUMBER	67
RESP_VERSION_MAIN	68
RESP_VERSION_MONO_DRIVE	69
RESP_VERSION_SLIT_DRIVE	70
RESP_WAVELENGTH	71
RESP_SCAN_MODE	72
RESP_SCAN_START	73
RESP_SCAN_STOP	74
RESP_SCAN_POINTS	75
RESP_CAL_MODE	76
RESP_UVVIS_WAVENUMBER_INTERVAL	77
RESP_NIR_WAVENUMBER_INTERVAL	78
RESP_GOTO_WAVELENGTH	79
RESP_BUSY	80
RESP_RESETTING	81
RESP_COLLECTING	82
RESP_OFFLINE	83
RESP_LAST_RD_VALUE	84
RESP_LAST_D2PEAK_WAVELENGTH	85
RESP_LAST_TIME	86
RESP_INSTRUMENT_TIME	87
RESP_LAMP1_TIME	88
RESP_LAMP2_TIME	89
RESP_LAMP3_TIME	90
RESP_TIMER	91
RESP_CURRENT_SOURCE	92
RESP_CURRENT_FILTER	93
RESP_CURRENT_GRATING	94
RESP_CURRENT_DETECTOR	95