



Document Type: Interface Defining Specification
Document Reference: ASCII Protocol.doc
Protocol Version: 1.80
Date: 08 May 2014
Author: M Stephens

Changes:

Date	Description of change
19-Apr-04	Commands list fully up to date.
20-Apr-04	Updated to protocol version 1.1 which removes RolloverCorrection parameter from all partial pressure measurement creation commands and adds MeasurementRolloverCorrection command. Info command also indicated whether sensor supports rollover correction or not.
13-Jul-04	Updates to protocol version 1.2. This remains compatible with 1.1 but adds additional commands for MV+ and IP units
13-Jun-05	All commands now have response format and minimal description filled out.
4-Oct-05	Made alteration to document to highlight the protocol version 1.3 ScanRestart command.
25-July-07	Added new commands for multiplier life enhancements. Protocol becomes 1.4 but remains compatible with the 1.2 version.
28-Aug-08	Protocol version 1.5 adds the AcceptProtocol command so clients can accept new format for existing messages, if the command is not used then the protocol remains unchanged for compatibility with existing pre1.5 code. By accepting protocol 1.5 or higher the client receives timestamp data with digital change notifications and RVC general purpose digital input change notifications.
21-Apr-10	Protocol version 1.6 adds features for the Microvision2 control unit such as interpolated tuning, diagnostic inputs, and per-port digital output timeout mask and times.
25-Oct-10	Protocol version 1.7 adds InletSelect command and changes InletInfo to indicate if inlets are automatic or manual.
25-Sep-13	Protocol version 1.8 adds DefaultZeroBeamOff state to the Info response data

RGA Ascii Protocol.....	5
Document conventions	5
Commands from client to sensor	5
Messages sent from the server to the client application	5
Responses from the sensor to commands issued by the client application	5
Asynchronous notification messages from the sensor to the client	6
Initial connection	6
Controlling message formatting	6
FormatWithTab	6
AcceptProtocol [protocol 1.5]	8
Sensor Information Commands	9
Sensors	9
Select	10

SensorState	11
Info [updated in protocol 1.6]	12
EGains	14
InletInfo	15
RFInfo	16
MultiplierInfo [updated in protocol 1.6]	17
SourceInfo [updated in protocol 1.6]	18
DetectorInfo	19
FilamentInfo	21
TotalPressureInfo [protocol 1.2, 1.6]	23
AnalogInputInfo [protocol 1.2, 1.6]	24
AnalogOutputInfo [protocol 1.2]	25
DigitalInfo [protocol 1.2, 1.6]	26
RolloverInfo [protocol 1.2]	28
RVCInfo [protocol 1.2]	29
CirrusInfo [protocol 1.2]	30
PECal_Info [protocol 1.2]	31
Sensor Information Commands In Protocol 1.6	32
SourceAlignmentInfo [protocol 1.6]	32
SourceResolutionInfo [protocol 1.6]	33
SourceTuningInfo [protocol 1.6]	34
DiagnosticInputInfo [protocol 1.6]	35
Gaining control of a sensor	36
Control	36
Release	37
Sensor Control Commands	38
FilamentControl	38
FilamentSelect	39
FilamentOnTime	40
AddAnalog	41
AddBarchart	43
AddPeakJump	45
AddSinglePeak	46
MeasurementAccuracy	47
MeasurementAddMass	48
MeasurementChangeMass	49
MeasurementDetectorIndex	50
MeasurementEGainIndex	51
MeasurementFilterMode	52
MeasurementMass	53
MeasurementPointsPerPeak	54
MeasurementRemoveMass	55
MeasurementSourceIndex	56
MeasurementRolloverCorrection	57
MeasurementZeroBeamOff	58
MeasurementZeroBufferDepth	59
MeasurementZeroBufferMode	60
MeasurementZeroReTrigger	61
MeasurementZeroMass	62
MultiplierProtect	63
RunDiagnostics	64
SetTotalPressure [protocol 1.2]	65
TotalPressureCalFactor [protocol 1.2]	66
TotalPressureCalDate [protocol 1.2]	67
CalibrationOptions	68
DetectorFactor	69

DetectorCalDate [protocol 1.2].....	70
DetectorVoltage	71
InletFactor	72
ScanAdd.....	73
ScanStart	74
ScanStop.....	75
ScanResume.....	76
ScanRestart [protocol 1.3]	77
MeasurementSelect	78
MeasurementStartMass	79
MeasurementEndMass	80
MeasurementMultSkipAutoProtect [Protocol 1.4]	81
MeasurementMultSkipMassAdd [Protocol 1.4].....	82
MeasurementMultSkipMassRemove [Protocol 1.4].....	83
MeasurementMultSkipMassRemoveAll [Protocol 1.4].....	84
MeasurementRemoveAll.....	85
MeasurementRemove.....	86
SourceIonEnergy	87
SourceEmission	88
SourceExtract.....	89
SourceElectronEnergy	90
SourceLowMassResolution	91
SourceLowMassAlignment	92
SourceHighMassAlignment.....	93
SourceHighMassResolution.....	94
AnalogInputAverageCount [protocol 1.2].....	95
AnalogInputEnable [protocol 1.2].....	96
AnalogInputInterval [protocol 1.2]	97
AnalogOutput [protocol 1.2]	98
AudioFrequency [protocol 1.2]	99
AudioMode [protocol 1.2]	100
CirrusCapillaryHeater [protocol 1.2].....	101
CirrusHeater [protocol 1.2].....	102
CirrusPump [protocol 1.2]	103
CirrusValvePosition [protocol 1.2].....	104
DigitalMaxPB67OnTime [protocol 1.2].....	105
DigitalOutput [protocol 1.2]	106
PECal_DateMsg [protocol 1.2].....	107
PECal_Flush [protocol 1.2]	108
PECal_Inlet [protocol 1.2]	109
PECal_MassMethodContribution [protocol 1.2].....	110
PECal_Pressures [protocol 1.2].....	111
PECal_Select [protocol 1.2]	112
RolloverScaleFactor [protocol 1.2].....	113
RolloverVariables [protocol 1.2]	114
RVCAAlarm [protocol 1.2]	115
RVCCloseAllValves [protocol 1.2].....	116
RVCH Heater [protocol 1.2].....	117
RVC Pump [protocol 1.2]	118
RVCValveControl [protocol 1.2]	119
RVCValveMode [protocol 1.2].....	120
SaveChanges [protocol 1.2].....	121
StartDegas [protocol 1.2]	122
StopDegas [protocol 1.2]	123
Sensor Control Commands In Protocol 1.6	124
SourceAlignmentCopyToAll [protocol 1.6]	124

SourceAlignmentUpdate [protocol 1.6]	125
SourceAlignmentRemove [protocol 1.6]	126
SourceResolutionCopyToAll [protocol 1.6]	127
SourceResolutionUpdate [protocol 1.6]	128
SourceResolutionRemove [protocol 1.6]	129
SourcePoleBias [protocol 1.6]	130
DiagnosticInputAverageCount [protocol 1.6]	131
DiagnosticInputEnable [protocol 1.6]	132
DiagnosticInputInterval [protocol 1.6]	133
Sensor Control Commands In Protocol 1.7	134
InletSelect [protocol 1.7]	134
Asynchronous Sensor Notifications	135
MKSRGA	135
FilamentStatus [updated in protocol 1.6]	136
FilamentTimeRemaining	137
StartingScan	138
StartingMeasurement	139
ZeroReading	140
MassReading [changed in protocol 1.4]	141
MultAutoSkip [Protocol 1.4]	142
MultiplierStatus [updated in protocol 1.6]	143
RFTripState	144
InletChange [protocol 1.2]	145
AnalogInput [protocol 1.2, 1.6]	146
TotalPressure [protocol 1.2, 1.6]	147
DigitalPortChange [protocol 1.2]	148
RVCPumpStatus [protocol 1.2]	149
RVHeaterStatus [protocol 1.2]	150
RVValveStatus [protocol 1.2]	151
RVInterlocks [protocol 1.2]	152
RVStatus [protocol 1.2]	153
RVCDigitalInput [protocol 1.2]	154
LinkDown [protocol 1.2]	155
VSCEvent [protocol 1.2]	156
DegasReading [protocol 1.2]	157
DiagnosticInput [protocol 1.6]	158

RGAA Ascii Protocol

Document conventions

In the description of the protocol that follows any examples of what is transmitted or received will appear in a `monospace` font. Any optional characters will be shown inside square braces []. To indicate whitespace, carriage return and linefeed characters the following characters are used in *italicized monospace* font:

`<ws>` Whitespace – one or more tab (ascii 9) or space (ascii 32) characters.
`<cr / lf>` Carriage return (ascii 13) OR Line feed (ascii 10) character.
`<crlf>` Line end – carriage return (ascii 13) followed by line feed (ascii 10).
`<rcrcr>` Message end – 2 successive carriage return (ascii 13) characters.

Where a parameter in the command/response may have different values that need clarifying the parameter name will be enclosed in curly braces {} and the description will indicate what valid values are for the parameter.

Commands from client to sensor

All commands sent from a client application to the sensor go on one line that is terminated by a carriage return, line feed or carriage return/line feed pair. On the line, items are separated by tabs or spaces and any items that contain spaces themselves should be enclosed in double quotes. The general format is as follows:

Command with no parameters:

```
command<cr / lf>
```

Command with parameters:

```
command<ws>["]parameter1["]<ws>["]parameter2["]<cr / lf>
```

Messages sent from the server to the client application

All messages sent from the sensor are terminated by 2 carriage return characters. The message may be made up of one or more lines which are terminated by a carriage return/line feed pair but the message itself will always end with 2 carriage returns in a row. The reason for this is that the protocol formats messages in a way that can be easily viewed in telnet/hyperterminal, using two carriage returns in a row does not effect the cursor position in those applications but provides an easy way to handle message termination in a client application when messages may be formatted across one or more lines. A simple client application will receive data from the sensor until it sees two consecutive carriage returns and then process the message.

Responses from the sensor to commands issued by the client application

Every command that the client issues will result in a response to acknowledge the command. The first line will have the command name as the first item followed by the word OK or ERROR indicating success or failure of the command. There will always be a blank line (carriage return/line feed pair) at the end of the response which is simply for formatting if using the protocol manually through telnet/hyperterminal or outputting messages during debugging.

If the command fails then the 2 lines following will again contain 2 items each. The first line will indicate an error number and the second an error description. Lines following will vary depending upon the error number but the error number and description will always be present for client software to use. Most well written clients would normally treat errors as catastrophic problems and end communication with the sensor, that is to say that errors will usually indicate a bug in the client that should be fixed or a problem with the sensor e.g. lack of memory to fulfill the command.

The lines below show formatting for an error where the error number is 200:

```
command<ws>ERROR<crlf>  
<ws>Number<ws>200<crlf>  
<ws>Description<ws>"err description"<crlf>
```

```
<crLf>  
<crCr>
```

If the command succeeds then the lines of the message body will vary from command to command. A basic successful command acknowledge is as follows:

```
command<ws>OK<crLf>  
<crLf>  
<crCr>
```

Asynchronous notification messages from the sensor to the client

As well as a client issuing commands to a sensor and receiving their responses it must also handle asynchronous messages sent from the sensor as things happen. Examples of these notifications are filament state changes, digital input changes, analog input readings and perhaps most importantly partial pressure readings. These notifications are all terminated by two carriage returns like all messages but the format of the message data will vary from notification to notification.

Initial connection

All RGA sensors will listen for connections on tcp/ip port 10014. When a connection is made to the sensor from a client application the sensor will send back it's initial response that allows a client to validate that it is talking to an RGA and that it is compatible with the version of the sensors protocol. The initial message is as follows:

```
MKSRGA<ws>{Type}<crLf>  
<ws>Protocol_Revision<ws>1.1<crLf>  
<ws>Min_Compatibility<ws>1.1<crLf>  
<crLf>  
<crCr>
```

{Type} can be either Single or Multi. Single indicates that the protocol is managing just 1 sensor e.g. it is a MicroVision IP or an eVision. Multi indicates that there is a server application that might be handling multiple sensors e.g. a windows based server application managing MicroVision+ sensors connected to a PC's serial ports. For most OEM applications which will use MicroVision IP's or eVision units this takes a couple of steps out of the connection process as there is no need to select the sensor you wish to talk to. If you are talking to a 'Multi' server then you should first get the list of sensors and then select the appropriate sensor.

Protocol_Revision indicates the version of the protocol that is in use by the sensor. Min_Compatibility indicates the lowest version number of the protocol that this version is compatible with. Clients should check this value against the protocol version that they were written for to ensure that they can communicate with the sensor, if not they should disconnect. It is hoped that a good level of backwards compatibility be maintained with the protocol going forward but at least if clients check these properties of the sensor it will avoid odd behaviour when old clients do connect to updated sensors that unfortunately do break compatibility.

Having made the connection, received and accepted the sensor type and versions you are ready to issue commands that retrieve information about the sensor.

Controlling message formatting

The following commands are used to control the formatting of messages and data in the protocol.

FormatWithTab

Parameters:

UseTab Boolean indicating whether to use tab characters in the output or spaces.

Response:

```
FormatWithTab<ws>OK<crlf>
<crlf>
<crcl>
```

Example:

```
FormatWithTab True
```

```
FormatWithTab OK
```

Description:

By default the output from commands is formatted using spaces to try to line everything up when output using a fixed width font (or terminal program). By sending this command clients can reduce the amount of characters sent in each message slightly as groups of spaces will be replaced by a single tab character.

Remarks:

AcceptProtocol [protocol 1.5]

Parameters:

Protocol Protocol version that the client accepts.

Response:

AcceptProtocol<ws>OK<crLf>
<crLf>
<crCr>

Example:

AcceptProtocol 1.5

AcceptProtocol OK

Description:

Starting with protocol version 1.5 clients can send this message to indicate that they accept messages that may have a new format from the base protocol where the format was first defined. In the case of version 1.5 of the protocol it means that digital input change events and RVC digital input change events now have an additional timestamp returned. If AcceptProtocol 1.5 (or higher) is not sent then the messages will not include the new timestamp information in order to maintain compatibility with existing clients.

Remarks:

Accepting protocol 1.5 provides timestamp data to digital input change event notifications

Accepting protocol 1.6 provides extended information for digital i/o configuration, total pressure, analog inputs, and provides new notification for diagnostic input data.

Sensor Information Commands

The following commands are used to interrogate a sensor about its configuration. Unless otherwise stated these commands can be issued at any time.

Sensors

Parameters:

None

Response:

```
Sensors<ws>OK<crlf>
<ws>State<ws>SerialNumber<ws>Name<crlf>
<ws>{SensorState}<ws>{SensorSerialNo}<ws>{SensorName}<crlf>
...
<crlf>
<crcr>
```

Example:

Sensors

```
Sensors OK
State  SerialNumber  Name
Ready  LM70-00197021  "Chamber A"
```

Description:

Returns a table of sensors that can be controlled through this connection. There may be 0 or more lines after the column headings line depending upon the number of sensors available.

{SensorState} can have the values InUse, Ready or Config indicating that the sensor is in use by another client, ready for use, or requires configuration before it can be used respectively.

Remarks:

If the initial MKSRGA message indicates a type of 'Single' then there is little value in issuing this command. It is only necessary for compatibility with windows servers providing access to older MicroVision+ hardware where there may be many sensors being managed by the server application.

Select

Parameters:

 SerialNumber The serial number of the sensor to select.

Response:

```
Select<ws>OK<crlf>
<ws>SerialNumber<ws>{SerialNumber}<crlf>
<ws>State<ws>{SensorState}<crlf>
<crlf>
<crcl>
```

Example:

```
Select LM70-00197021

Select OK
SerialNumber LM70-00197021
State       Ready
```

Description:

Selects a sensor as the one to get information about. All other commands are then directed at this sensor.

Remarks:

If the initial MKSRGA message indicates a type of 'Single' then the single sensor that the server is managing is already selected making this command unnecessary for most OEM applications. However for compatibility with all hardware the Sensors command and Select command should be used to explicitly select the desired sensor.

SensorState

Parameters:

None

Response:

```
SensorState<ws>OK<crlf>
<ws>State<ws>{State}<crlf>
<ws>UserApplication<ws>{UserApp}<crlf>
<ws>UserVersion<ws>{UserVer}<crlf>
<ws>UserAddress<ws>{UserAddress}<crlf>
<crlf>
<crcl>
```

Example:

SensorState

```
SensorState OK
State           InUse
UserApplication "Process Eye Professional"
UserVersion     V5.2
UserAddress     127.0.0.1
```

Description:

Retrieves the state that the selected sensor is currently in. {State} can be one of the following:

Ready	The unit is ready for use.
InUse	The unit is currently in use by someone.
Config	The unit requires configuring and is unavailable to most applications.
N/A	The unit is unavailable.

{UserApp}, {UserVer} and {UserAddress} will be N/A when {State} is anything other than InUse. When the sensor is in use these values indicate the client application that is using the sensor and it's IP Address.

Remarks:

This information is also duplicated in the response to the Info command.

Info [updated in protocol 1.6]

Parameters:

None

Response:

```
Info<ws>OK<crlf>
<ws>SerialNumber<ws>{SerialNumber}<crlf>
<ws>Name<ws>{FriendlyName}<crlf>
<ws>State<ws>{SensorState}<crlf>
<ws>UserApplication<ws>{UserApp}<crlf>
<ws>UserVersion<ws>{UserVersion}<crlf>
<ws>UserAddress<ws>{UserAddress}<crlf>
<ws>ProductID<ws>{ProductID}<ws>{ProductName}<crlf>
<ws>RFConfiguration<ws>{RFConfigID}<ws>{RFConfigName}<crlf>
<ws>DetectorType<ws>{DetectorTypeID}<ws>{DetectorTypeName}<crlf>
<ws>SEMSupply<ws>{SEMSupplyID}<ws>{SEMSupplyName}<crlf>
<ws>ExternalHardware<ws>{ExternalHardwareID}<ws>{ExternalHWName}<crlf>
<ws>TotalPressureGauge<ws>{TPGaugeID}<ws>{TPGaugeName}<crlf>
<ws>FilamentType<ws>{FilamentTypeID}<ws>{FilamentTypeName}<crlf>
<ws>ControlUnitUse<ws>{ControlUnitUseID}<ws>{CUUName}<crlf>
<ws>SensorType<ws>{SensorTypeID}<ws>{SensorTypeName}<crlf>
<ws>InletType<ws>{InletTypeID}<ws>{InletTypeName}<crlf>
<ws>Version<ws>{SensorSoftwareVersion}<crlf>
<ws>NumEGains<ws>{EGainCount}<crlf>
<ws>NumDigitalPorts<ws>{DigitalPortCount}<crlf>
<ws>NumAnalogInputs<ws>{AnalogInputCount}<crlf>
<ws>NumAnalogOutputs<ws>{AnalogOutputCount}<crlf>
<ws>NumSourceSettings<ws>{SourceSettingsCount}<crlf>
<ws>NumInlets<ws>{InletCount}<crlf>
<ws>MaxMass<ws>{MaxMass}<crlf>
<ws>ActiveFilament<ws>{ActiveFilament}<crlf>
<ws>FullScaleADCamps<ws>{FullScaleADCamps}<crlf>
<ws>FullScaleADCCount<ws>{FullScaleADCCount}<crlf>
<ws>PeakResolution<ws>{PeakResolution}<crlf>
<ws>ConfigurableIonSource<ws>{ConfigurableIonSource}<crlf>
<ws>RolloverCompensation<ws>{SupportsRolloverCorrection}<crlf>
<ws>InterpolatedTuning<ws>{SupportsInterpolatedTuning}<crlf>
<ws>DefaultZeroBeamOff<ws>{Yes/No}<crlf> ** Protocol 1.6
<crlf>
<crcr>
```

Example:

Info

```
Info OK
SerialNumber      LM70-00197021
Name              "Chamber A"
State             Ready
UserApplication   N/A
UserVersion       N/A
UserAddress       N/A
ProductID         70      MicroVision+
RFConfiguration   0       "Smart Head"
DetectorType      0       Faraday
SEMSupply         3000    3.0kV
ExternalHardware   0      None
TotalPressureGauge 0      "Not Fitted"
```

FilamentType	0	Tungsten
ControlUnitUse	4	"Standard RGA"
SensorType	1	"Standard Open Source"
InletType	1	None
Version	V3.70	
NumEGains	3	
NumDigitalPorts	2	
NumAnalogInputs	4	
NumAnalogOutputs	1	
NumSourceSettings	6	
NumInlets	1	
MaxMass	200	
ActiveFilament	1	
FullScaleADCAmps	0.000002	
FullScaleADCCount	8388608	
PeakResolution	32	
ConfigurableIonSource	Yes	
RolloverCompensation	No	

Description:

Returns important configuration information about the sensor. Many applications will be able to safely ignore a lot of the information but for MKS applications that work with a range of different hardware and take advantage of all of the features the information is important.

Remarks:

In protocol version 1.6 the last line of the message will be InterpoletedTuning Yes/No. For Microvision2 and eVision2 units this will be Yes and all others No. If the control unit does support interpolated tuning then the client should also use the SourceAlignmentInfo, SourceResolutionInfo and SourceTuningInfo commands to discover more information and should ignore the LowMassAlignment, HighMassAlignment, LowMassResolution and HighMassResolution information from the SourceInfo command response.

EGains

Parameters:

None

Response:

```
EGains<ws>OK<crlf>
<ws>{ElectronicGain1}<crlf>
...
<ws>{ElectronicGain n}<crlf>
<crlf>
<crcl>
```

Example:

```
EGains
EGains OK
1
100
20000
```

Description:

Returns the list of electronic gain factors available for the sensor. The number of electronic gains may vary for different sensor hardware platforms. The number available can be seen in the response to the Info command.

Remarks:

InletInfo

Parameters:

None

Response:

```
InletInfo<ws>OK<crlf>
<ws>Automatic<ws>{Yes/No}<crlf>  ** Protocol 1.7
<ws>ActiveInlet<ws>{ActiveInletIndex}<crlf>
<ws>Factor<ws>Fixed<ws>CanCalibrate<ws>DefaultFactor<ws>TypeName<crlf>
<ws>{Factor}<ws>{Fixed}<ws>{CanCalib}<ws>{Default}<ws>{InletType}<crlf>
...
<crlf>
<crcl>
```

Example:

InletInfo

```
InletInfo OK
Automatic Yes      ** Protocol 1.7
ActiveInlet 0
  Factor  Fixed  CanCalibrate  DefaultFactor  TypeName
    1      Yes    No             1             "Process Chamber direct"
```

Description:

Returns a table of inlet information. After the first header line there will be 1 or more inlets listed depending upon the configuration of the instrument.

{Factor} Gives the pressure reduction factor of the inlet.

{Fixed} Indicates if it is a fixed or variable inlet.

{CanCalib} Indicates if we can calibrate the inlet or not.

{Default} Is the default inlet factor for this type of inlet.

{InletType} Is the type of this inlet.

Remarks:

Protocol 1.7 introduces the ability to have multiple inlets configured in the control unit but have the ascii protocol client select which is in use via the InletSelect command. This command is only valid when the inlets are not considered to be automatic. The first line in the response indicates if the inlets are automatic or not

RFInfo

Parameters:

None

Response:

```
RFInfo<ws>OK<crLf>
<ws>RFTripEnabled<ws>{Yes/No}<crLf>
<ws>RFTripped<ws>{Yes/No}<crLf>
<crLf>
<crCr>
```

Example:

RFInfo

```
RFInfo OK
    RFTripEnabled      Yes
    RFTripped         No
```

Description:

Retrieves the current configuration and state of the RF Trip. If the RF Trip is enabled then the controlling client will be kept informed of the current trip state by the RFTripState message. {TripEnabled} and {Tripped} will both be either 'yes' or 'no'.

Remarks:

MultiplierInfo [updated in protocol 1.6]

Parameters:

None

Response:

```
<ws>MultiplierInfo<ws>OK<crlf>
<ws>InhibitWhenFilamentOff<ws>{Yes/No}<crlf>
<ws>InhibitWhenRVCHeaterOn<ws>{Yes/No}<crlf>
<ws>MultiplierOn<ws>{Yes/No}<crlf>
<ws>LockedByFilament<ws>{Yes/No}<crlf>
<ws>LockedByRVC<ws>{Yes/No}<crlf>
<ws>LockedBySoftware<ws>{Yes/No}<crlf>
<ws>HardwareTripped<ws><crlf>          ** Added in protocol 1.6
<crlf>
<crcr>
```

Example:

```
MultiplierInfo  OK
  InhibitWhenFilamentOff      Yes
  InhibitWhenRVCHeaterOn      Yes
  MultiplierOn                No
  LockedByFilament             Yes
  LockedByRVC                  No
  LockedBySoftware             No
```

Description:

The sensor can be configured to inhibit the multiplier while filaments are off, or when the RVC heater is on (MicroVision+ or IP with RVC only). This command retrieves this configuration, the current state of the multiplier and the reason why it is locked.

Remarks:

See the MultiplierProtect command for details of inhibiting the multiplier via software.

Protocol version 1.6 adds the HardwareTripped line to the message. In a Microvision2 unit there is a hardware lockout feature for the multiplier and this is reported here, for older control units the line will always read No.

SourceInfo [updated in protocol 1.6]

Parameters:

SourceIndex Zero based index of the source settings table

Response:

```
SourceInfo<ws>OK<crlf>
<ws>SourceIndex<ws>{SourceIndex}<crlf>
<ws>Name<ws>{SourceName}<crlf>
<ws>ElectronEnergy<ws>{ElectronEnergy}<crlf>
<ws>IonEnergy<ws>{IonEnergy}<crlf>
<ws>ExtractVolts<ws>{ExtractVolts}<crlf>
<ws>ElectronEmission<ws>{ElectronEmission}<crlf>
<ws>LowMassAlignment<ws>{LMA}<crlf>
<ws>HighMassAlignment<ws>{HMA}<crlf>
<ws>LowMassResolution<ws>{LMR}<crlf>
<ws>HighMassResolution<ws>{HMR}<crlf>
<ws>MaxRecommendedPressure<ws>{MaxPressure}<crlf>
<ws>NumFaradayEGains<ws>{NumFaradayEGains}<crlf>
<ws>NumMultEGains<ws>{NumMultEGains}<crlf>
<ws>PoleBias<ws>{PoleBiasVoltage}<crlf>      ** Protocol 1.6 and control unit supports PoleBias
<crlf>
<crcl>
```

Example:

SourceInfo 0

```
SourceInfo OK
SourceIndex      0
Name             *Unconfigured1*
ElectronEnergy   70.0
IonEnergy        5.5
ExtractVolts     -112
ElectronEmission 1.0
LowMassAlignment 32767
HighMassAlignment 32767
LowMassResolution 32767
HighMassResolution 32767
MaxRecommendedPressure 1.3333e-002
NumFaradayEGains 2
NumMultEGains    2
```

Description:

Returns information about a specific set of source settings. The SourceIndex parameter is a zero based index of the source settings table to retrieve. To find out how many source settings tables there are an application should use the Info command and look at the NumSourceSettings property.

Remarks:

If the protocol version is 1.6 or greater and the control unit supports PoleBias like Microvision2 then the PoleBias line is added to the end of the message. PoleBias is specified in Volts 0-10.

DetectorInfo

Parameters:

SourceIndex Zero based index of the source settings table

Response 1.1:

```
DetectorInfo<ws>OK
<ws>SourceIndex<ws>{SourceIndex}<crLf>
<ws>Name<ws>DefaultFactor<ws>DefaultVoltage<ws>Factor1<ws>Voltage1<ws>Factor2<ws>Voltage2<crLf>
<ws>{Name}<ws>{DFact}<ws>{DVolt}<ws>{Fact1}<ws>{Volt1}<ws>{Fact2}<ws>{Volt2}<crLf>
...
<crLf>
<crCr>
```

Response 1.2:

```
DetectorInfo<ws>OK
<ws>SourceIndex<ws>{SourceIndex}<crLf>
<ws>Name<ws>DefaultFactor<ws>DefaultVoltage<ws>Factor1<ws>Voltage1<ws>Factor2<ws>Voltage2<ws>CalDate1
<ws>CalDate2<crLf>
<ws>{Name}<ws>{DFact}<ws>{DVolt}<ws>{Fact1}<ws>{Volt1}<ws>{Fact2}<ws>{Volt2}<ws>{Date1}<ws>{Date2}
<crLf>
...
<crLf>
<crCr>
```

Example:

```
DetectorInfo 0

DetectorInfo  OK
SourceIndex 0
Name          DefaultFactor DefaultVoltage Factor1   Voltage1 Factor2   Voltage2 CalDate1
CalDate2
Faraday       1.50e-04      0           1.00e+00  0         1.00e+00  0         2004-11-01_10:01:05
2004-11-01_10:02:00
Mult1         1.50e-04      -650        1.00e+00 -650       1.00e+00 -650       0000-00-00_00:00:00
0000-00-00_00:00:00
Mult2         1.50e-04      -700        1.00e+00 -700       1.00e+00 -700       0000-00-00_00:00:00
0000-00-00_00:00:00
Mult3         1.50e-04      -900        1.00e+00 -900       1.00e+00 -900       0000-00-00_00:00:00
0000-00-00_00:00:00
```

Description:

Returns a table of information about the detector settings for a particular source table (see SourceInfo command). If the sensor is a faraday only device then this table will only contain one row of data corresponding to the faraday detector. For multiplier there will typically be 3 more rows allowing the multiplier to be used with specific gain/calibration characteristics for a given application, however the current list of 3 rows should not be assumed. Different sensor types or detector types may result in a different number of settings in the future and software should be dynamic in dealing with more or less detector settings.

{Name}	Is the name of this detector setting.
{DFact}	Is the default calibration factor for the detector setting in Amps/Pascal.
{DVolt}	Is the multiplier voltage if the detector is to use multiplier, 0 for faraday detector.
{Fact1}	Is the current calibration factor for filament 1.
{Volt1}	Is the current detector voltage to be used for filament 1.
{Fact2}	Is the current calibration factor for filament 2
{Volt2}	Is the current detector voltage to be used for filament 2.
{Date1}	Is the calibration date that filament 1 was calibrated
{Date2}	Is the calibration date that filament 2 was calibrated

Remarks:

CalDate1 and CalDate2 information was added in revision 1.2 of the protocol. All dates in the protocol are formatted as YYYY-MM-DD_hh:mm:ss where

YYYY is the full year as 4 digits

MM is the month number 01 to 12 as 2 digits

DD is the day of the month 01 to 31 as 2 digits

hh is the hour in the day 00 to 23 as 2 digits

mm is the minutes in the hour 00 to 59 as 2 digits

ss is the seconds in the minute 00 to 59 as 2 digits

If all digits are 0 then the time/date is empty and has never been set, otherwise the time/date represents a UTC time so conversion to the local time should be done by clients for display if required.

FilamentInfo

Parameters:

None

Response:

```
FilamentInfo<ws>OK<crlf>
<ws>SummaryState<ws>{State}<crlf>
<ws>ActiveFilament<ws>{1/2}<crlf>
<ws>ExternalTripEnable<ws>{Yes/No}<crlf>
<ws>ExternalTripMode<ws>{X-Trip Mode}<crlf>
<ws>EmissionTripEnable<ws>{Yes/No}<crlf>
<ws>MaxOnTime<ws>{Time (s)}<crlf>
<ws>OnTimeRemaining<ws>{Time (s)}<crlf>
<ws>Trip<ws>{Trip}<crlf>
<ws>Drive<ws>{On/Off}<crlf>
<ws>EmissionTripState<ws>{OK/Fail}<crlf>
<ws>ExternalTripState<ws>{OK/Fail}<crlf>
<ws>RVCTripState<ws>{OK/Fail}<crlf>
<ws>GaugeTripState<ws>{OK/Fail}<crlf>    **Protocol 1.6
<crlf>
<cr>
```

Example:

FilamentInfo

```
FilamentInfo  OK
SummaryState      OFF
ActiveFilament    2
ExternalTripEnable No
ExternalTripMode   Trip
EmissionTripEnable Yes
MaxOnTime          900
OnTimeRemaining    0
Trip               None
Drive              Off
EmissionTripState  OK
ExternalTripState  OK
RVCTripState       OK
```

Description:

Retrieves the current configuration and state of the filaments. For simple applications SummaryState might be all the information required. The information is as follows:

SummaryState	This indicates the overall state of the filaments, possible values are: OFF WARM-UP ON COOL-DOWN BAD-EMISSION
ActiveFilament	The currently selected filament, 1 or 2
ExternalTripEnable	Whether the external trip is enabled or not: Yes/No
ExternalTripMode	The mode of operation for the external trip. Modes are: Control The external trip line controls the filaments Trip The external trip line trips the filaments off
EmissionTripEnable	Whether bad emission will trip the filaments or not: Yes/No
MaxOnTime	The maximum time in seconds that the filaments will stay on for without the controlling

	application sending the FilamentOnTime message. If this value is 0 then the filaments will stay on indefinitely.
OnTimeRemaining	The remaining time for the filaments to stay on in seconds if they are on and MaxOnTime is not 0. Otherwise this value will be reported as 0.
Trip	<p>The current trip state of the filaments. If the filaments trip off then this indicates the reason for the trip. Possible values are:</p> <p>None Filaments are not in a tripped state</p> <p>Emission Filaments tripped due to bad emission</p> <p>External The external trip line caused the filaments to go off</p> <p>RVC The RVC caused filaments to trip (MicroVision+ / IP only with RVC)</p> <p>GaugeFail* Total pressure gauge failed to come on (if fitted)</p> <p>TotalPressure* Filaments tripped due to total pressure level being too high</p> <p>* GaugeFail and TotalPressure are new for Microvision2, protocol 1.6</p>
Drive	This indicates the current state of the hardware, whether power is being applied to the filaments or not: On/Off
EmissionTripState ExternalTripState RVCTripState GaugeTripState*	<p>These remaining fields indicate the current hardware readings for various trip states. The Trip field will indicate what (if anything) caused the filaments to trip. The value reported is OK or Fail and any combination for the 4 fields is possible. Most applications will ignore this information.</p> <p>*GaugeTripState is new for Microvision2, protocol 1.6</p>

Remarks:

An application should use this command to retrieve the current filament configuration and state early on and then be prepared to see the FilamentStatus and FilamentTimeRemaining asynchronous messages to keep in sync.

TotalPressureInfo [protocol 1.2, 1.6]

Parameters:

None

Response:

```
TotalPressureInfo<ws>OK<crlf>
<ws>AverageCount<ws>{AvgCount}<crlf>
<ws>Interval<ws>{Interval}<crlf>
<ws>CalFactor<ws>{Factor}<crlf>
<ws>CalDate<ws>{Date/Time}<crlf>
<ws>Pressure<ws>{Pressure}<crlf>
<crlf>
<cr>
```

Example:

TotalPressureInfo

```
TotalPressureInfo  OK
AverageCount      10
Interval          1000
CalFactor         1.0
CalDate           2004-10-30_11:55:01
Pressure          1.0E-4
```

Description:

If the sensor has a total pressure gauge fitted then this command returns information about the current state and settings being used.

AverageCount and Interval determine the number of readings that are taken and averaged together before the total pressure is reported and the interval between readings in microseconds.

CalFactor is the factor applied to the readings.

Pressure is the current reading for pressure in units of Pascal. Note that depending on the gauge type this value may be 0 if the sensors filaments are off, this is because ion gauges integrate with the sensors filament logic so unless the filaments are on no accurate pressure readings can be measured.

CalDate is the UTC date time that the gauge was calibrated. See DetectorInfo command for details on the format of date/time values.

Remarks:

Protocol 1.6 can now indicate if the gauge reading is invalid (no readings taken yet), under-range or over-range (the analog input reading was outside the limits valid for the gauge). This information is made available when the AcceptProtocol command has been sent with the protocol version 1.6 or greater, in these cases the numeric value for the Pressure reading will be Invalid, UnderRange or OverRange.

AnalogInputInfo [protocol 1.2, 1.6]

Parameters:

None

Response:

```
AnalogInputInfo<ws>OK<crlf>
<ws>Enabled<ws>MinVolts<ws>MaxVolts<ws>Resolution<ws>Interval<ws>AverageCount<ws>Value<
crlf>
<ws>{Enabled}<ws>{Min}<ws>{Max}<ws>{Res}<ws>{Interval}<ws>{Avg}<ws>{Value}<crlf>
...
<crlf>
<crcr>
```

Example:

AnalogInputInfo

```
AnalogInputInfo OK
Enabled  MinVolts  MaxVolts  Resolution  Interval  AverageCount  Value
No       -10.0         10.0      16          100000    10            0
No       -10.0         10.0      16          100000    10            0
No       -10.0         10.0      16          100000    10            0
No       -10.0         10.0      16          100000    10            0
```

Description:

Returns information about all the analog inputs that the sensor has. The number of items following the column headers of the response is available from the Info command. The data for each input is as follows:

Enabled	Yes/No whether the input is being sampled by the sensor or not. When in control of the sensor and the inputs are enabled their readings will be sent back from the sensor in the AnalogInput event response.
MinVolts	The minimum voltage that can be measured by the analog input
MaxVolts	The maximum voltage that can be measured by the analog input
Resolution	The resolution of the ADC
Interval	The number of microseconds between successive readings that the sensor takes from the analog input.
AverageCount	The number of individual readings that are taken and averaged before a result is sent back to the controlling connection.
Value	The last known reading of the analog input or 0 if unknown.

Remarks:

Protocol 1.6 can now indicate if the input reading is invalid (no readings taken yet), under-range or over-range (the analog input reading was outside the limits set in the control unit configuration). This information is made available when the AcceptProtocol command has been sent with the protocol version 1.6 or greater, in these cases the numeric value for the input reading will be Invalid, UnderRange or OverRange.

MicroVision2 supports a user configured conversion function to convert analog input data from voltage to some other logical unit. Currently not all the information about the conversion function is made available via the protocol but it does mean that analog input readings might not always be between the MinVolts and MaxVolts values.

AnalogOutputInfo [protocol 1.2]

Parameters:

None

Response:

```
AnalogOutputInfo<ws>OK<crlf>
<ws>MinVolts<ws>MaxVolts<ws>Resolution<ws>Value<crlf>
<ws>{MinVolts}<ws>{MaxVolts}<ws>{Resolution}<ws>{Value}<crlf>
...
<crlf>
<crcl>
```

Example:

AnalogOutputInfo

```
AnalogOutputInfo  OK
  MinVolts  MaxVolts  Resolution  Value
    0.0      10.0      12          0
```

Description:

Returns information about all analog outputs that a sensor has. The number of analog outputs is available from the Info command. The data returned for each output is as follows

MinVolts	The full scale minimum voltage that can be converted by the DAC
MaxVolts	The full scale maximum voltage that can be converted by the DAC
Resolution	The resolution of the DAC
Value	The last value that the output was set to by a controlling connection or 0

Remarks:

DigitalInfo [protocol 1.2, 1.6]

Parameters:

None

Response:

For pre protocol 1.6 or for Microvision+ and IP units the response is as follows:

```
DigitalInfo<ws>OK<crlf>
<ws>DeglitchEnabled<ws>{Yes/No}<crlf>
<ws>DeglitchTime<ws>{time microseconds}<crlf>
<ws>MaxPB67OnTime<ws>{time in seconds}<crlf>
<ws>Name<ws>ConnectedMask<ws>OutputMask<ws>Value<crlf>
<ws>{Name}<ws>{ConnectedMask}<ws>{OutputMask}<ws>{Value}<crlf>
...
<crlf>
<crcl>
```

For protocol 1.6 communicating with a Microvision2 the response is as follows:

```
DigitalInfo<ws>OK<crlf>
<ws>DeglitchEnabled<ws>{Yes/No}<crlf>
<ws>DeglitchTime<ws>{time microseconds}<crlf>
<ws>MaxPB67OnTime<ws>N/A<crlf>
<ws>Name<ws>ConnectedMask<ws>OutputMask<ws>TimeoutMask<ws>Timeout<ws>Value<crlf>
<ws>{Name}<ws>{ConnectedMask}<ws>{OutputMask}<ws>{TimeoutMask}<ws>{Timeout}<ws>{Value}<
crlf>
...
<crlf>
<crcl>
```

Example:

DigitalInfo

```
DigitalInfo OK
  DeglitchEnabled  No
  DeglitchTime    0
  MaxPB67OnTime   0
  Name  ConnectedMask  OutputMask  Value
  A     63             48           49
  B     0              0           0
```

Or for MicroVision2

```
DigitalInfo OK
  DeglitchEnabled  No
  DeglitchTime    0
  MaxPB67OnTime   N/A
  Name  ConnectedMask  OutputMask  TimeoutMask  Timeout  Value
  A     63             48           8            40      49
  B     0              0           0            0      0
```

Description:

Returns information about the fitted digital input ports:

DeglitchEnabled	This setting refers to all ports and determines whether any deglitching logic is applied when detecting changes in input bits.
DeglitchTime	If DeglitchEnabled is Yes then this is the time in microseconds that a change must occur for before counting as a valid change.
MaxPB67OnTime	MicroVision+ sensors have logic to only allow port B bits 6 and 7 to be set for a certain

	amount of time, this was to control a calibration gas bottle and avoid accidental waste. This value is 0 if not configured or not supported, otherwise the time is in seconds. For MicroVision2 this concept has been extended to allow any output bits on any port to be timed and then this value will be N/A
Name	Each port is assigned a name which is simply an alphabetic index, e.g. A, B, C, etc.
ConnectedMask	Each port is configured with which bits are being used. This value will be a number between 0 and 255 where each set bit indicates that the corresponding bit is configured for use.
OutputMask	Each bit that is configured as an output has it's bit set in this value.
TimeoutMask	Any bits that are outputs can optionally be configured to only stay on for a certain time period. Bits set in this field that are also set in the OutputMask field will only stay on for the number of seconds specified for the port
Timeout	The number of seconds that an output bit that is included in the TimeoutMask will stay on for without re-asserting its state.
Value	The current value of the digital port. Any input changes will cause a DigitalPortChange message to the connection that is controlling the sensor.

Remarks:

TimeoutMask and Timeout are new features of the Microvision2 control unit and protocol version 1.6. In order to see this information the client must have issued the AcceptProtocol command specifying protocol 1.6 or higher.

RolloverInfo [protocol 1.2]

Parameters:

None

Response:

```
RolloverInfo<ws>OK<crLf>
<ws>M1<ws>{value}<crLf>
<ws>M2<ws>{value}<crLf>
<ws>B1<ws>{value}<crLf>
<ws>B2<ws>{value}<crLf>
<ws>BP1<ws>{value}<crLf>
<ws>Mass<ws>Scale<crLf>
<ws>{mass value}<ws>{scale factor}<crLf>
...
<crLf>
<crCr>
```

Example:

```
RolloverInfo  OK
M1  -470
M2  -250
B1  -0.15
B2  -0.91
BP1 0.0012
Mass  Scale
2     0.43
4     0.34
14    0.58
15    0.33
17    0.41
18    0.41
28    0.58
29    0.58
32    0.55
36    1.0
38    1.0
40    1.0
```

Description:

Returns configuration settings for the rollover correction algorithm used in the HPQ2s. The sensor must report that it supports rollover correction in the Info command for this command to succeed.

The algorithm is proprietary to MKS, this and related commands are only expected to be used by MKS software only.

Remarks:

RVCInfo [protocol 1.2]

Parameters:

None

Response:

```
RVCInfo<ws>OK<crlf>
<ws>ValveMode<ws>{Automatic/Manual}<crlf>
<ws>Interlocks<ws>{On/Off}<crlf>
<ws>Status0<ws>{OK/OverPressure}<crlf>
<ws>Status1<ws>{OK/OverPressure}<crlf>
<ws>Valve0<ws>{Open/Closed}<crlf>
<ws>Valve1<ws>{Open/Closed}<crlf>
<ws>Valve2<ws>{Open/Closed}<crlf>
<ws>Heater<ws>{Off/On/CoolingDown}<crlf>
<ws>Pump<ws>{Off/Accelerating/On}<crlf>
<ws>Alarm<ws>{True/False}<crlf>
<ws>DigitalInput0<ws>{True/False}<crlf>
<ws>DigitalInput1<ws>{True/False}<crlf>
<crlf>
<crcr>
```

Example:

RVCInfo

```
RVCInfo OK
ValveMode      Automatic
Interlocks     On
Status0        OverPressure
Status1        OK
Valve0         Closed
Valve1         Closed
Valve2         Closed
Heater         Off
Pump           Off
Alarm          False
DigitalInput0  False
DigitalInput1  False
```

Description:

Returns the current state of the RVC if the sensor has an RVC fitted. See the ExternalHardware information in the Info commands response.

Remarks:

CirrusInfo [protocol 1.2]

Parameters:

None

Response:

```
CirrusInfo<ws>OK<crlf>
<ws>ChamberPressure<ws>{Pressure in Pa or N/A}<crlf>
<ws>HeaterStatus<ws>{Off/Warm/Bake}<crlf>
<ws>CapillaryHeaterStatus<ws>{On/Off}<crlf>
<ws>PumpStatus<ws>{Off/Accelerating/On}<crlf>
<ws>ValveCount<ws>{Count}<crlf>
<ws>ValvePosition<ws>{Position or N/A}<crlf>
<crlf>
<crcr>
```

Example:

CirrusInfo

```
CirrusInfo  OK
ChamberPressure      N/A
HeaterStatus         Off
CapillaryHeaterStatus Off
PumpStatus           Off
ValveCount           4
ValvePosition        0
```

Description:

Returns current Cirrus status and configuration if the sensor is a Cirrus. See the ExternalHardware value in the Info command response.

Remarks:

PECal_Info [protocol 1.2]

Parameters:

SourceIndex The 0 based index of the source parameters
DetectorIndex The 0 based index of the detector (0=Faraday, 1,2,3=Multiplier settings)

Response:

```
PECal_Info<ws>OK<crlf>
<ws>Source<ws>{SourceIndex}<crlf>
<ws>Detector<ws>{DetectorIndex}<crlf>
<ws>Date<ws>{yyyy-mm-dd HH:MM:SS}<crlf>
<ws>Mass<ws>{CalibrationMass}<crlf>
<ws>ProcessPressure<ws>{ProcessPressure}<crlf>
<ws>AnalyserPressure<ws>{AnalyserPressure}<crlf>
<ws>MaxPeakHeight<ws>{MaxPeakHeight}<crlf>
<ws>Contribution<ws>{Contribution}<crlf>
<ws>Method<ws>{Method}<crlf>
<ws>Inlet1<ws>{I1}<crlf>
<ws>Inlet2<ws>{I2}<crlf>
<ws>Inlet3<ws>{I3}<crlf>
<ws>Message<ws>{Message}<crlf>
<crlf>
<crcr>
```

Example:

```
PECal_Info 0 0
```

```
PECal_Info OK
Source          0
Detector        0
Date            0000-00-00_00:00:00
Mass            0
ProcessPressure 0e+000
AnalyserPressure 0e+000
MaxPeakHeight   0e+000
Contribution     0.00
Method          0
Inlet1          1
Inlet2          1
Inlet3          1
Message         ""
```

Description:

Remarks:

This command is specifically part of the ascii protocol to allow Process Eye and Recipe Wizard software to continue to store the same calibration information at the sensor/server that they did prior to the ascii protocol. It is not meant to be used by non MKS software.

Sensor Information Commands In Protocol 1.6

Protocol version 1.6 defines the modifications to the existing protocol to support Microvision2 and eVision2 hardware as well as completely new extensions to the protocol where the operation of the new units differs significantly from its predecessors. The major differences are in tuning the resolution and mass alignment as well as the ability to continually monitor any of the available 19 diagnostic inputs.

SourceAlignmentInfo [protocol 1.6]

Parameters:

SourceIndex Zero based index of the source settings table

Response:

```
SourceAlignmentInfo<ws>OK<crlf>
<ws>SourceIndex<ws>{SourceIndex}<crlf>
<ws>Mass<ws>DAC<crlf>
<ws>{MassValue}<ws>{DACValue}<crlf>
...
<crlf>
<crclr>
```

Example:

```
SourceAlignmentInfo 0

SourceAlignmentInfo  OK
SourceIndex          0
Mass      DAC
1         2e
4         12a
28        7a1
```

Description:

Returns information about the mass alignment settings for a specific source entry for a control unit where the Info command indicates that InterpolatedTuning is supported. The returned table shows the masses that have been calibrated to a specific mass program DAC value. All other masses are interpolated using these key known points on the mass scale.

Remarks:

Note that the DAC values are returned as hexadecimal numbers. In order to avoid the mass scale going backwards at any point each successive Mass/DAC pair must increase along the mass scale and be within limits calculated using information from the SourceTuningInfo command response.

SourceResolutionInfo [protocol 1.6]

Parameters:

SourceIndex Zero based index of the source settings table

Response:

```
SourceResolutionInfo<ws>OK<crlf>
<ws>SourceIndex<ws>{SourceIndex}<crlf>
<ws>Mass<ws>DAC<crlf>
<ws>{MassValue}<ws>{DACValue}<crlf>
...
<crlf>
<crcr>
```

Example:

```
SourceResolutionInfo 0

SourceResolutionInfo  OK
SourceIndex           0
Mass      DAC
1         a43
4         12a
28        7a1
```

Description:

Returns information about the mass resolution settings for a specific source entry for a control unit where the Info command indicates that InterpolatedTuning is supported. The returned table shows the masses that have been calibrated to a specific resolution DAC value. All other masses are interpolated using these key known points on the mass scale.

Remarks:

Note that the DAC values are returned as hexadecimal numbers

SourceTuningInfo [protocol 1.6]

Parameters:

None

Response:

```
SourceTuningInfo<ws>OK<crlf>
<ws>MassDACBits<ws>{MassDACBits}<crlf>
<ws>MassDACUpperLimitM<ws>{UpperLimitM}<crlf>
<ws>MassDACUpperLimitC<ws>{UpperLimitC}<crlf>
<ws>MassDACLowerLimitM<ws>{LowerLimitM}<crlf>
<ws>MassDACLowerLimitC<ws>{LowerLimitC}<crlf>
<ws>ResolutionDACBits<ws>{ResolutionDACBits}<crlf>
<crlf>
<crcr>
```

Example:

SourceTuningInfo

```
SourceTuningInfo  OK
MassDACBits      16
MassDACUpperLimitM  655.349976
MassDACUpperLimitC  13107
MassDACLowerLimitM  655.349976
MassDACLowerLimitC  -13107
ResolutionDACBits  12
```

Description:

Returns information about the mass program DAC and the resolution DAC as well as details for determining the valid limits for the mass program DAC for a given mass.

The MassDACUpperLimitM and C values define a linear function that gives the upper value for any mass via the equation:

$$\text{MaxDAC} = \text{MassDACUpperLimitM} * \text{Mass} + \text{MassDACUpperLimitC}$$

The lower limit for a given mass can be found in the same way using the two lower limit values.

Remarks:

Note that there are no upper and lower limits defined for the resolution settings since any valid DAC value is allowed at any mass.

DiagnosticInputInfo [protocol 1.6]

Parameters:

None

Response:

```
DiagnosticInputInfo<ws>OK<crLf>
<ws>Enabled<ws>Name<ws>Units<ws>Interval<ws>AverageCount<ws>Value<crLf>
<ws>{Enabled}<ws>{Name}<ws>{Units}<ws>{Interval}<ws>{Avg}<ws>{Value}<crLf>
...
<crLf>
<crCr>
```

Example:

DiagnosticInputInfo

DiagnosticInputInfo OK

Enabled	Name	Units	Interval	AverageCount	Value
Yes	3.3V	Volts	15000000	1	3.274487495422
No	"RF Ident"	Volts	15000000	1	0
No	"Extraction Monitor"	Volts	15000000	1	0
Yes	-15V	Volts	15000000	1	-14.9417123794
Yes	"RF Temperature"	"Degrees C"	15000000	1	55.4289855957
Yes	"RF Tune Voltage"	Volts	15000000	1	-1.21917533874
Yes	+24V	Volts	15000000	1	23.9114990234
Yes	-130V	Volts	15000000	1	-139.320251464
No	"Ion Energy"	Volts	15000000	1	0
No	"SEM Monitor"	Volts	15000000	1	0
No	"Source Current Monitor"	mA	15000000	1	0
No	"Mass DAC"	Volts	15000000	1	0
No	"Filament Current"	Amps	15000000	1	0
No	"Electron Energy"	Volts	15000000	1	0
No	Resolution	Volts	15000000	1	0
Yes	"Pole Bias"	Volts	15000000	1	0.010219739746
Yes	"Electrometer Temperature"	"Degrees C"	15000000	1	54.282787322
Yes	+15V	Volts	15000000	1	14.9000310897
Yes	"Internal Temperature"	"Degrees C"	15000000	1	36.5464706420

Description:

Returns information about the available diagnostic input channels. For old units like MicroVision+ and eVision this will be an empty table, the table above is from a Microvision2 unit. Interval is specified in microseconds and enabled channels will send results approximately every Interval*AverageCount microseconds via the DiagnosticInput event message.

Remarks:

Diagnostic inputs are considered low priority and the results may be sent less frequently than the configuration asks for depending on what else the control unit is doing. Interval times are set in microseconds but in the Microvision2 requesting that a reading be returned every 10-100ms would be considered very fast.

Gaining control of a sensor

In order to actually do anything useful with a sensor an application must take control of it. Only one application can ever be in control of a sensor at any one time. When a sensor is being controlled it's State will be seen as InUse to all other connections. Those connections will also be able to see the application name, version and IP address of the controlling application.

Control

Parameters:

AppName String specifying the application name of the controlling application
Version String specifying the version of the controlling application

Response:

```
Control<ws>OK<crlf>
<ws>SerialNumber<ws>{SerialNo}<crlf>
<crlf>
<crcl>
```

Example:

```
Control "Process Eye Pro" "5.1"

Control OK
SerialNumber LM70-00197021
```

Description:

Takes control of a sensor if it is not currently in use already. See Sensors/SensorState and Info commands for details on finding out if a sensor is currently in use or not.

Remarks:

AppName and Version parameters can be any strings that make sense for any particular application but they must both be less than 64 characters in length or they will be truncated.

For the Control command to succeed the sensor must have been selected. For single sensors such as the e-Vision and MicroVision IP they are automatically selected when the tcp/ip connection is made, for compatibility with all sensors see the Select command.

Because another application may have taken control of the sensor before this command is issued you must be prepared to see and handle error responses from this command.

Release

Parameters:

None

Response:

Release<ws>OK<crLf>

<crLf>

<crCr>

Example:

Release

Release OK

Description:

Releases control of the sensor. This command is only valid after a successful Control command. If the sensor is still scanning or has measurements assigned then the scan will be stopped and any resources cleaned up.

Following the command the sensor is still the selected sensor and all sensor information commands can still be issued.

Remarks:

If the connection is lost or closed then the same things happen at the sensor as when the Release command is issued, however cleaning up by sending the Release command is the recommended way to end control.

Sensor Control Commands

The following commands can only be successfully issued when the sensor is being controlled, they are the workhorse commands that allow readings to be taken or settings to be changed.

FilamentControl

Parameters:

State Can be 'On' or 'Off'

Response:

```
FilamentControl<ws>OK<crlf>
<ws>State<ws>{RequestedState}<crlf>
<crlf>
<crcl>
```

Example:

```
FilamentControl On
```

```
FilamentControl OK
State On
```

Description:

Turns the currently selected filament On or Off.

Remarks:

The returned State value is the requested state when the command was sent, so if you send FilamentControl On then the response will report the State as On. However the actual state of the filaments should only be used from the FilamentInfo command response or the asynchronous message FilamentStatus.

Sending this command will likely generate FilamentStatus messages as the filaments come on or go off, a client application should not assume any order for these responses as depending upon the implementation of the sensor, a client may see the FilamentControl response before any FilamentStatus messages or it may come afterwards.

FilamentSelect

Parameters:

Number The filament number to select: 1 or 2

Response:

```
FilamentSelect<ws>OK<crlf>
<ws>Number<ws>{FilamentNumber}<crlf>
<crlf>
<crcl>
```

Example:

```
FilamentSelect 2
```

```
FilamentSelect  OK
Number  2
```

Description:

Selects a particular filament.

Remarks:

The Number parameter returned in the response is simply the requested filament number. Applications should use the response from FilamentInfo command and the asynchronous FilamentStatus message to keep in sync with filament state. This command may cause FilamentStatus messages to be generated before or after the response is received.

FilamentOnTime

Parameters:

Time Number of seconds to keep the filaments on for.

Response:

```
FilamentOnTime<ws>OK<crlf>
<ws>Time<ws>{Time}<crlf>
<crlf>
<crcl>
```

Example:

```
FilamentOnTime 200
```

```
FilamentOnTime  OK
Time 200
```

Description:

Sets the amount of time that filaments will stay on for if the unit is configured to use a time limit before filaments automatically go off. Time is specified in seconds and can be between 60 and 43200 (1 minute to 12 hours).

Remarks:

AddAnalog

Parameters:

Name	The name that the measurement should be called
StartMass	The starting mass that should be scanned
EndMass	The ending mass that should be scanned
PointsPerPeak	Number of points to be measured across each mass
Accuracy	Accuracy code to be used
EGainIndex	Electronic Gain index
SourceIndex	Source parameters index
DetectorIndex	Detector parameters index

Response:

```
AddAnalog<ws>OK<crlf>
<ws>Name<ws>{Name}<crlf>
<ws>StartMass<ws>{Mass}<crlf>
<ws>EndMass<ws>{Mass}<crlf>
<ws>PointsPerPeak<ws>{PointsPerPeak}<crlf>
<ws>Accuracy<ws>{Acc}<crlf>
<ws>EGainIndex<ws>{EGain}<crlf>
<ws>SourceIndex<ws>{Source}<crlf>
<ws>DetectorIndex<ws>{Detector}<crlf>
<crlf>
<crcr>
```

Example:

```
AddAnalog Analog1 1 50 32 5 0 0 0
```

```
AddAnalog OK
```

Name	Analog1
StartMass	1
EndMass	50
PointsPerPeak	32
Accuracy	5
EGainIndex	0
SourceIndex	0
DetectorIndex	0

Description:

Adds a new analog measurement to the sensor. The parameters are as follows:

Name	Unique name for this measurement.
StartMass	Mass to start scanning from, must be between 1 and the instruments maximum mass
EndMass	Mass to end scanning on, must be between StartMass and the instruments maximum mass
PointsPerPeak	Number of points to scan across a peak. The Info command response specifies the maximum value in it's PeakResolution field. Typically this is 32 and values of 16, 8 and 4 are also allowed.
Accuracy	Accuracy code between 0 and 8 where 0 is the fastest but less accurate and 8 is slowest but most accurate. The scan speeds for each accuracy code vary for different sensor models, see appendix for more information.
EGainIndex	0 based index of the electronic gain setting to use for measurement. See the EGains command for details on what electronic gains are available.
SourceIndex	0 based index of the source settings to use for measurement. The e-Vision sensor does not support configurable ion-source parameters and therefore only accepts a value of 0 here. The MicroVision+ and MicroVision IP both support up to 6 sets of ion source parameters which can be configured to suit the hardware configuration and environment where the sensor is used.
DetectorIndex	0 based index of the detector settings to use. 0 is always faraday and if the sensor has a multiplier then 1,2 and 3 are different settings for the detector voltage.

Remarks:

Upon a successful AddAnalog command the analog measurement will be the new selected measurement.

AddBarchart

Parameters:

Name	The name that the measurement should be called
StartMass	The starting mass that should be scanned
EndMass	The ending mass that should be scanned
FilterMode	How masses should be scanned and converted into a single reading
Accuracy	Accuracy code to be used
EGainIndex	Electronic Gain index
SourceIndex	Source parameters index
DetectorIndex	Detector parameters index

Response:

```
AddBarchart<ws>OK<crlf>
<ws>Name<ws>{Name}<crlf>
<ws>StartMass<ws>{Mass}<crlf>
<ws>EndMass<ws>{Mass}<crlf>
<ws>FilterMode<ws>{FilterMode}<crlf>
<ws>Accuracy<ws>{Acc}<crlf>
<ws>EGainIndex<ws>{EGain}<crlf>
<ws>SourceIndex<ws>{Source}<crlf>
<ws>DetectorIndex<ws>{Detector}<crlf>
<crlf>
<crcl>
```

Example:

```
AddBarchart Bar1 1 50 PeakCenter 5 0 0 0
```

```
AddBarchart OK
Name           Bar1
StartMass      1
EndMass        50
FilterMode     PeakCenter
Accuracy       5
EGainIndex     0
SourceIndex    0
DetectorIndex  0
```

Description:

Adds a new barchart measurement to the sensor. The parameters are as follows:

Name	Unique name for this measurement.
StartMass	Mass to start scanning from, must be between 1 and the instruments maximum mass
EndMass	Mass to end scanning on, must be between StartMass and the instruments maximum mass
FilterMode	Specifies how each AMU should be scanned and turned into a single reading: <ul style="list-style-type: none">• PeakCenter Single point at the nominal peak center is measured• PeakMax Central ½ AMU scanned and the max value reported• PeakAverage Central ¼ AMU scanned and the average value reported
Accuracy	Accuracy code between 0 and 8 where 0 is the fastest but less accurate and 8 is slowest but most accurate. The scan speeds for each accuracy code vary for different sensor models, see appendix for more information.
EGainIndex	0 based index of the electronic gain setting to use for measurement. See the EGains command for details on what electronic gains are available.
SourceIndex	0 based index of the source settings to use for measurement. The e-Vision sensor does not support configurable ion-source parameters and therefore only accepts a value of 0 here. The MicroVision+ and MicroVision IP both support up to 6 sets of ion source parameters which can be configured to suit the hardware configuration and environment where the sensor is used.
DetectorIndex	0 based index of the detector settings to use. 0 is always faraday and if the sensor has a

multiplier then 1,2 and 3 are different settings for the detector voltage.

Remarks:

Upon a successful AddBarchart command the barchart measurement will be the new selected measurement.

AddPeakJump

Parameters:

Name	The name that the measurement should be called
FilterMode	How masses should be scanned and converted into a single reading
Accuracy	Accuracy code to be used
EGainIndex	Electronic Gain index
SourceIndex	Source parameters index
DetectorIndex	Detector parameters index

Response:

```
AddPeakJump<ws>OK<crLf>
<ws>Name<ws>{Name}<crLf>
<ws>FilterMode<ws>{FilterMode}<crLf>
<ws>Accuracy<ws>{Acc}<crLf>
<ws>EGainIndex<ws>{EGain}<crLf>
<ws>SourceIndex<ws>{Source}<crLf>
<ws>DetectorIndex<ws>{Detector}<crLf>
<crLf>
<crCr>
```

Example:

```
AddPeakJump PeakJump1 PeakCenter 5 0 0 0
```

```
AddPeakJump OK
Name                PeakJump1
FilterMode           PeakCenter
Accuracy             5
EGainIndex           0
SourceIndex          0
DetectorIndex        0
```

Description:

Adds a new peak jump measurement to the sensor. The parameters are as follows:

Name	Unique name for this measurement.
FilterMode	Specifies how each AMU should be scanned and turned into a single reading: <ul style="list-style-type: none">• PeakCenter Single point at the nominal peak center is measured• PeakMax Central ½ AMU scanned and the max value reported• PeakAverage Central ¼ AMU scanned and the average value reported
Accuracy	Accuracy code between 0 and 8 where 0 is the fastest but less accurate and 8 is slowest but most accurate. The scan speeds for each accuracy code vary for different sensor models, see appendix for more information.
EGainIndex	0 based index of the electronic gain setting to use for measurement. See the EGains command for details on what electronic gains are available.
SourceIndex	0 based index of the source settings to use for measurement. The e-Vision sensor does not support configurable ion-source parameters and therefore only accepts a value of 0 here. The MicroVision+ and MicroVision IP both support up to 6 sets of ion source parameters which can be configured to suit the hardware configuration and environment where the sensor is used.
DetectorIndex	0 based index of the detector settings to use. 0 is always faraday and if the sensor has a multiplier then 1,2 and 3 are different settings for the detector voltage.

Remarks:

Upon a successful AddPeakJump command the peak jump measurement will be the new selected measurement.

AddSinglePeak

Parameters:

Name	The name that the measurement should be called
Mass	The mass that should be measured
Accuracy	Accuracy code to be used
EGainIndex	Electronic Gain index
SourceIndex	Source parameters index
DetectorIndex	Detector parameters index

Response:

```
AddSinglePeak<ws>OK<crlf>
<ws>Name<ws>{Name}<crlf>
<ws>Mass<ws>{Mass}<crlf>
<ws>Accuracy<ws>{Acc}<crlf>
<ws>EGainIndex<ws>{EGain}<crlf>
<ws>SourceIndex<ws>{Source}<crlf>
<ws>DetectorIndex<ws>{Detector}<crlf>
<crlf>
<crcl>
```

Example:

```
AddSinglePeak SinglePeak1 4.2 5 0 0 0
```

```
AddSinglePeak OK
```

Name	PeakJump1
Mass	4.1875
Accuracy	5
EGainIndex	0
SourceIndex	0
DetectorIndex	0

Description:

Adds a new single peak measurement to the sensor. Unlike the other measurements, this one takes a floating point value for the mass and can measure any point across the mass span of the sensor. The parameters are as follows:

Name	Unique name for this measurement.
Mass	Floating point mass value. Notice that in the example mass 4.2 was specified to the command but the response indicates the closest 1/32 of an AMU that can be measured which is 4.1875.
Accuracy	Accuracy code between 0 and 8 where 0 is the fastest but less accurate and 8 is slowest but most accurate. The scan speeds for each accuracy code vary for different sensor models, see appendix for more information.
EGainIndex	0 based index of the electronic gain setting to use for measurement. See the EGains command for details on what electronic gains are available.
SourceIndex	0 based index of the source settings to use for measurement. The e-Vision sensor does not support configurable ion-source parameters and therefore only accepts a value of 0 here. The MicroVision+ and MicroVision IP both support up to 6 sets of ion source parameters which can be configured to suit the hardware configuration and environment where the sensor is used.
DetectorIndex	0 based index of the detector settings to use. 0 is always faraday and if the sensor has a multiplier then 1,2 and 3 are different settings for the detector voltage.

Remarks:

Upon a successful AddSinglePeak command the single peak measurement will be the new selected measurement.

MeasurementAccuracy

Parameters:

Accuracy 0 – 8 Accuracy code

Response:

```
MeasurementAccuracy<ws>OK<crlf>
<ws>Accuracy<ws>{Acc}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementAccuracy 4
```

```
MeasurementAccuracy OK
Accuracy 4
```

Description:

Changes the accuracy code of the currently selected measurement.

Remarks:

See appendix for details of how accuracy codes effect scan speed and quality of data for different sensor types.

MeasurementAddMass

Parameters:

Mass Integer mass value

Response:

```
MeasurementAddMass<ws>OK<crlf>
<ws>Mass<ws>{Mass}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementAddMass 10
MeasurementAddMass OK
Mass 10
```

Description:

Adds a mass to a peak jump measurement. The measurement must be the currently selected measurement

Remarks:

MeasurementChangeMass

Parameters:

MassIndex Index of the mass that should be changed
NewMass New mass value that should be scanned instead

Response:

```
MeasurementChangeMass<ws>OK<crlf>  
<ws>MassIndex<ws>{Index}<crlf>  
<ws>NewMass<ws>{Mass}<crlf>  
<crlf>  
<crcr>
```

Example:

```
MeasurementChangeMass 0 6  
  
MeasurementChangeMass OK  
  MassIndex 0  
  NewMass 6
```

Description:

Changes a mass on a Peak Jump measurement. The measurement must be the selected measurement (see AddPeakJump and MeasurementSelect commands). The MassIndex is the index of the mass in the order that they were added, so if the measurement had mass 5, 10 and 15 added in that order their indexes would be 0, 1 and 2 respectively.

Remarks:

MeasurementDetectorIndex

Parameters:

DetectorIndex 0 based index of the detector to use for the measurement

Response:

```
MeasurementDetectorIndex<ws>OK<crlf>
<ws>DetectorIndex<ws>{Index}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementDetectorIndex 0

MeasurementDetectorIndex OK
DetectorIndex 0
```

Description:

Changes the selected measurements detector index. Faraday is detector 0 and if the sensor has a multiplier then indexes 1,2 and 3 provide alternate settings for the multiplier voltage.

Remarks:

MeasurementEGainIndex

Parameters:

EGainIndex 0 based index of the electronic gain to use for the measurement

Response:

```
MeasurementEGainIndex<ws>OK<crlf>
<ws>EGainIndex<ws>{Index}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementWGainIndex 1

MeasurementEGainIndex OK
EGainIndex 1
```

Description:

Changes a measurements electronic gain index. See EGains command for information on Electronic Gain settings.

Remarks:

MeasurementFilterMode

Parameters:

FilterMode The mode to be used to filter readings down to 1 per AMU

Response:

```
MeasurementFilterMode<ws>OK<crlf>
<ws>FilterMode<ws>{Mode}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementFilterMode PeakCenter
```

```
MeasurementFilterMode OK
FilterMode PeakCenter
```

Description:

Selects the mass filter mode to be used for Barchart and Peak Jump measurements. Filter mode can be PeakCenter, PeakMax or PeakAverage. See AddBarchart and AddPeakJump commands for more details on how the FilterMode affects the acquisition.

Remarks:

MeasurementMass

Parameters:

Mass The mass value to use for the selected single peak measurement. Can be fractional

Response:

```
MeasurementMass<ws>OK<crlf>
<ws>Mass<ws>{NewMass}<crlf>
<crlf>
<cr>
```

Example:

```
MeasurementMass 15.5
```

```
MeasurementMass OK
Mass 15.5
```

Description:

Changes the mass used for the selected single peak measurement. The mass can be fractional.

Remarks:

MeasurementPointsPerPeak

Parameters:

PointsPerPeak The number of points per peak to be measured for Analog measurement

Response:

```
MeasurementPointsPerPeak<ws>OK<crlf>
<ws>PointsPerPeak<ws>{Points}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementPointsPerPeak 16
```

```
MeasurementPointsPerPeak OK
PointsPerPeak 16
```

Description:

Sets the selected analog measurements number of points to measure per peak (or AMU). The Info command gives the maximum PeakResolution of the sensor. Usually this is 32, acceptable values are always powers of 2 so if the maximum peak resolution is 32 then the valid values are 1, 2, 4, 8, 16 and 32.

Remarks:

MeasurementRemoveMass

Parameters:

MassIndex 0 based index of the mass peak to remove from a Peak Jump measurement

Response:

```
MeasurementRemoveMass<ws>OK<crlf>
<ws>MassIndex<ws>{Index}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementRemoveMass 1

MeasurementRemoveMass OK
MassIndex 1
```

Description:

Removes a mass peak from the selected Peak Jump measurement. MassIndex is 0 based so if the measurement had had masses 5, 10 and 15 added in that order then the example above would remove index 1 which is mass 10 from the measurement.

Remarks:

This command cannot be used while a scan is in progress.

MeasurementSourceIndex

Parameters:

SourceIndex 0 based index of the source parameters to use for the measurement

Response:

```
MeasurementSourceIndex<ws>OK<crlf>
<ws>SourceIndex<ws>{Index}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementSourceIndex 0

MeasurementSourceIndex OK
SourceIndex 0
```

Description:

Changes the selected measurements source parameters. The number of source parameter entries for a sensor is given by the Info command and the actual source parameters are listed by the SourceInfo command. The e-Vision and e-Vision+ have just one fixed set of source parameters so this command is of little use for that sensor model.

Remarks:

MeasurementRolloverCorrection

Parameters:

UseCorrection True/False whether to use rollover correction for the selected measurement

Response:

```
MeasurementRolloverCorrection<ws>OK<crlf>
<ws>UseCorrection<ws>{True/False}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementRolloverCorrection True
```

```
MeasurementRolloverCorrection OK
UseCorrection True
```

Description:

Changes whether the selected measurement uses rollover correction. The sensor must be able to support rollover correction (see Info command). For a sensor to support rollover correction it must be provided with regular accurate total pressure readings.

Remarks:

Introduced in protocol revision 1.1

MeasurementZeroBeamOff

Parameters:

BeamOff Boolean indicating if the beam should be off during zero readings.

Response:

```
MeasurementZeroBeamOff<ws>OK<crlf>
<ws>BeamOff<ws>{True/False}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementZeroBeamOff True
```

```
MeasurementZeroBeamOff OK
BeamOff True
```

Description:

Controls whether the ion beam should be on or off during a measurements zero readings. The default is True, so the beam is off during measurements zeroing.

Remarks:

MeasurementZeroBufferDepth

Parameters:

ZeroBufferDepth The depth of the zero reading buffer.

Response:

```
MeasurementZeroBufferDepth<ws>OK<crlf>  
<ws>ZeroBufferDepth<ws>{Depth}<crlf>  
<crlf>  
<crcl>
```

Example:

```
MeasurementZeroBufferDepth 8  
  
MeasurementZeroBufferDepth OK  
ZeroBufferDepth 8
```

Description:

Sets the selected measurements zero buffer depth. The valid values for the buffer depth are 1, 2, 4, 8, 16 and 32. The default depth is 8.

The MeasurementZeroBufferMode command determines how the buffer is used. Factors such as the scan speed and how quickly things may change in the system should be considered when choosing a suitable zero buffer depth.

Remarks:

MeasurementZeroBufferMode

Parameters:

ZeroBufferMode The mode of operation for the zero averaging logic

Response:

```
MeasurementZeroBufferMode<ws>OK<crlf>
<ws>ZeroBufferMode<ws>{Mode}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementZeroBufferMode MultiScanAverage
```

```
MeasurementZeroBufferMode OK
ZeroBufferMode MultiScanAverage
```

Description:

Sets the selected measurements zero buffer mode. The options are as follows:

- SingleScanAverage The entire zero buffer is re-filled every scan
- MultiScanAverage The zero buffer is filled on first scan and then rolling average
- MultiScanAverageQuickStart One zero is used to fill buffer on first scan and then rolling average
- SingleShot Same as SingleScanAverage but only run once. To re-take the zero the MeasurementZeroReTrigger command must be issued.

The default for all measurements except SinglePeak is MultiScanAverage. For SinglePeak measurements the default is SingleShot as this lends itself to the leakcheck style of working.

Remarks:

MeasurementZeroReTrigger

Parameters:

None

Response:

```
MeasurementZeroReTrigger<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
MeasurementZeroReTrigger  
  
MeasurementZeroBufferMode OK
```

Description:

Re-triggers the selected measurements zero buffer if it's mode is SingleShot.

Remarks:

MeasurementZeroMass

Parameters:

ZeroMass The mass value that should be used to take the zero readings for the measurement

Response:

```
MeasurementZeroMass<ws>OK<crlf>
<ws>ZeroMass<ws>{MassPosition}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementZeroMass 5.5
```

```
MeasurementZeroMass OK
ZeroMass 5.5
```

Description:

Sets the mass position where the selected measurement should take it's zero readings from. The default position is mass 5.5.

Remarks:

MultiplierProtect

Parameters:

Protect Boolean indicating if the multiplier should be locked by software.

Response:

```
MultiplierProtect<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
MultiplierProtect True  
  
MultiplierProtect OK
```

Description:

Controls whether the multiplier is allowed to come on or not. This command may cause MultiplierStatus messages to be generated.

Remarks:

This command is only valid when the sensor has a multiplier.

RunDiagnostics

Parameters:

None

Response:

```
RunDiagnostics<ws>OK<crLf>
<ws>Name<ws>Min<ws>Max<ws>Value<ws>Result<crLf>
<ws>{Name}<ws>{Min}<ws>{Max}<ws>{Value}<ws>{Result}<crLf>
...
<crLf>
<crCr>
```

Example:

RunDiagnostics

```
RunDiagnostics  OK
Name           Min      Max      Value    Result
+450V          460     440      0        N/A
-15V           -16.5   -13.5   -15.12    Pass
-130V          -160    -140   -153.32    Pass
+5V            4.75    5.35     5.00     Pass
+15V           13.5    17.5    14.93     Pass
EE             -68     -30    -58.35     Pass
Extractor      -119.2  -102.60  0         N/A
SEM            -690    -610     0         N/A
```

Description:

Runs the sensors diagnostics measurements. The response contains a table of the measurements completed which may be different for different sensor models.

The result field can be N/A if the test was not done, Pass or Fail.

Remarks:

This command can only be done when the instrument is not running a scan.

SetTotalPressure [protocol 1.2]

Parameters:

Pressure Value to be used for total pressure

Response:

```
SetTotalPressure<ws>OK<crlf>
<crlf>
<crcl>
```

Example:

```
TotalPressure 1.0E-4

TotalPressure OK
```

Description:

If no gauge is fitted for measuring total pressure it is sometimes useful to pass in a value for total pressure so that the sensors roll over correction can still function properly. This command is only valid when there is no total pressure gauge fitted.

Remarks:

TotalPressure should be in units of Pascal.

TotalPressureCalFactor [protocol 1.2]

Parameters:

Factor Float value to apply to total pressure reading from an external gauge

Response:

```
TotalPressureCalFactor<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
TotalPressureCalFactor 1.0

TotalPressureCalFactor OK
```

Description:

Sets a value to apply to external gauge total pressure readings to compensate for any differences between the gauge and the true pressure.

Remarks:

This command can only be used when the instrument has a total pressure gauge fitted

TotalPressureCalDate [protocol 1.2]

Parameters:

DateTime Date in form yyyy-mm-dd_HH:MM:SS

Response:

```
TotalPressureCalDate<ws>OK<crlf>
<crlf>
<crcr>
```

Example:

```
TotalPressureCalDate 2005-10-06_16:44:00
```

```
TotalPressureCalDate OK
```

Description:

Sets the date/time associated with a calibration. To reset the time to an empty state pass 0000-00-00_00:00:00. For valid date time values the time should be in UTC.

Remarks:

CalibrationOptions

Parameters:

InletOption	How to apply inlet calibration factor
DetectorOption	How to apply detector calibration factor

Response:

```
CalibrationOptions<ws>OK<crlf>
<crlf>
<crcl>
```

Example:

```
CalibrationOptions Off Off
```

```
CalibrationOptions OK
```

Description:

Sets how to apply calibration factors to acquired measurement data. Both InletOption and DetectorOption parameters can have the same values which are:

- Off The inlet or detector factor will be set to 1.0
- Default The inlet or detector factor will be set to the factory default setting for the sensor/inlet type
- Current The current calibrated factor values will be used

Upon connection to a sensor the calibration options are set to use the current calibrated values for both the detector and inlet.

Remarks:

DetectorFactor

Parameters:

SourceIndex	The 0 based index of the source settings being used
DetectorIndex	The 0 based index of the detector settings being used
Filament	The filament number 1 or 2. Or 0 if both filaments factors to be set
Factor	The new calibration factor

Response:

```
DetectorFactor <ws>OK<crlf>
<crlf>
<crcl>
```

Example:

```
DetectorFactor 0 0 1 1.5e-6

DetectorFactor OK
```

Description:

Sets a calibration factor for a given set of source parameters and detector parameters. For e-Vision and e-Vision+ sensors there is only one set of fixed source settings so SourceIndex will always be 0 but for other sensors there may be more. DetectorIndex 0 refers to the Faraday detector and if the sensor has a multiplier then indexes 1, 2 and 3 are available multiplier settings with different gain settings. Each filament has it's own calibration factor that can be set by specifying Filament 1 or 2, alternatively both filaments calibration factors can be set by specifying filament number of 0.

Remarks:

Calibration factors should be set using Amps/Pascal units so that pressure readings from a sensor come out in the S.I. unit for pressure, Pascal. All MKS software provides conversion to a users preferred pressure units at the display level while stored data is maintained in Pascal units, thus making datafiles portable amongst different users and locales but easily viewable in whatever units the user prefers.

DetectorCalDate [protocol 1.2]

Parameters:

SourceIndex	The 0 based index of the source settings being used
DetectorIndex	The 0 based index of the detector settings being used
Filament	The filament number 1 or 2. Or 0 if both filaments factors to be set
Date	The time and date formatted as yyyy-mm-dd_HH:MM:SS

Response:

```
<ws>OK<CrLf>
<CrLf>
<Crcr>
```

Example:

```
DetectorCalDate 0 0 0 2005-06-01_11:49:00
```

```
DetectorCalDate OK
```

Description:

Sets a calibration date for a given set of source parameters and detector parameters. For e-Vision and e-Vision+ sensors there is only one set of fixed source settings so SourceIndex will always be 0 but for other sensors there may be more. DetectorIndex 0 refers to the Faraday detector and if the sensor has a multiplier then indexes 1, 2 and 3 are available multiplier settings with different gain settings. Each filament has it's own calibration date that can be set by specifying Filament 1 or 2, alternatively both filaments calibration dates can be set by specifying filament number of 0.

Remarks:

The date parameter MUST be formatted as yyyy-mm-dd_HH:MM:SS where yyyy is the year, mm is the month, dd is the day, HH is hours, MM is minutes and SS is seconds. This string must be 19 characters in length to be accepted as a valid time/date value. A special case is where all fields are 0, this effectively resets the date setting to an empty or unset state.

Calibration dates should be set using UTC time. All MKS software provides conversion to a users timezone at the display level while stored data is maintained in UTC, thus making datafiles portable amongst different users and locales but easily viewable in the users current locale.

DetectorVoltage

Parameters:

SourceIndex	The 0 based index of the source settings being used
DetectorIndex	The 0 based index of the detector settings being used
Filament	The filament number 1 or 2. Or 0 if both filaments factors to be set
Voltage	The new multiplier voltage to use

Response:

```
DetectorVoltage<ws>OK<crlf>
<crlf>
<crcl>
```

Example:

```
DetectorVoltage 0 1 1 500
```

```
DetectorVoltage OK
```

Description:

Sets the multiplier voltage for a particular set of detector settings. The sensor must have a multiplier and DetectorIndex must be 1, 2 or 3 since 0 is the Faraday detector. Each filament can use a different multiplier voltage, specifying 1 or 2 sets the individual filament while specifying 0 will set both filaments to the same value.

Remarks:

InletFactor

Parameters:

InletIndex 0 based index of the inlet to set the factor for.
Factor The new inlet factor

Response:

```
InletFactor<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
InletFactor 0 1.5
```

```
InletFactor OK
```

Description:

Sets a particular inlets pressure reduction factor. InletIndex is 0 based index. The inlet must be a calibratable inlet for this command to complete successfully.

Remarks:

ScanAdd

Parameters:

MeasurementName The measurement to add to the scan

Response:

```
ScanAdd<ws>OK<crlf>
<ws>Measurement<ws>{Name}<crlf>
<crlf>
<crcl>
```

Example:

```
ScanAdd Analog1
```

```
ScanAdd OK
Measurement Analog1
```

Description:

Adds a measurement to the scans list of measurements. Any measurement may only be added once to the scan. When all measurements are added to the scan it can be started using the ScanStart command.

Remarks:

Measurements cannot be added to the scan when a scan is already running.

ScanStart

Parameters:

NumScans

Response:

```
ScanStart<ws>OK<crlf>  
<crlf>  
<crcl>
```

Example:

```
ScanStart 1
```

```
ScanStart OK
```

Description:

Starts a scan running and will re-trigger the scan automatically the number of times specified by NumScans. This will cause StartingScan, StartingMeasurement, ZeroReading and MassReading notifications.

Remarks:

ScanStop

Parameters:

None

Response:

```
ScanStop<ws>OK<crLf>  
<crLf>  
<crCr>
```

Example:

```
ScanStop
```

```
ScanStop OK
```

Description:

Stops a scan and removes all measurements from the scan list. If the scan is not running then the measurements are just removed from the list.

Remarks:

ScanResume

Parameters:

NumScans Number of scans to re-trigger the scan for.

Response:

```
ScanResume<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
ScanResume 1
```

```
ScanResume OK
```

Description:

Re-triggers the scan NumScans times. See StartingScan notification for more information on how this command can be used.

Remarks:

ScanRestart [protocol 1.3]

Parameters:

None

Response:

```
ScanRestart<ws>OK<crlf>
<crlf>
<crcl>
```

Example:

```
ScanRestart
```

```
ScanRestart  OK
```

Description:

Re-starts the current scan from the beginning. Sometimes it might be useful to scan continuously and then when some external event occurs synchronise with it. This command allows the current scan to be restarted so that you can be sure all data for the scan is valid after some event.

Remarks:

MeasurementSelect

Parameters:

MeasurementName The measurement that should be selected for other MeasurementXXXX commands

Response:

```
MeasurementSelect<ws>OK<crlf>
<ws>Measurement<ws>{Name}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementSelect Analog1

MeasurementSelect OK
Measurement Analog1
```

Description:

Selects a measurement for other MeasurementXXXX commands to act upon. In the example above the measurement called Analog1 is selected, following this command MeasurementAccuracy might be used to change the accuracy code used by the measurement.

Remarks:

MeasurementStartMass

Parameters:

Mass The new start mass for the Analog or Barchart measurement

Response:

```
MeasurementStartMass<ws>OK<crLf>
<ws>StartMass<ws>{NewStartMass}<crLf>
<ws>EndMass<ws>{NewEndMass}<crLf>
<crLf>
<crCr>
```

Example:

```
MeasurementStartMass 50
```

```
MeasurementStartMass OK
  StartMass 50
  EndMass 50
```

Description:

Sets the selected Analog or Barchart measurements starting mass

Remarks:

The mass must be within the mass range of the sensor. See the Info command for details of the sensors maximum mass range.

If the Mass parameter is higher than the current EndMass of the measurement then the EndMass parameter will be moved up to match the new StartMass, otherwise EndMass will remain as it was before the command.

MeasurementEndMass

Parameters:

Mass The new end mass for the Analog or Barchart measurement

Response:

```
MeasurementEndMass<ws>OK<crlf>
<ws>StartMass<ws>{NewStartMass}<crlf>
<ws>EndMass<ws>{NewEndMass}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementEndMass 45
```

```
MeasurementEndMass OK
  StartMass 45
  EndMass 45
```

Description:

Sets the selected Analog or Barchart measurements ending mass

Remarks:

The mass must be within the mass range of the sensor. See the Info command for details of the sensors maximum mass range.

If the Mass parameter is lower than the current StartMass of the measurement then the StartMass parameter will be moved down to match the new EndMass, otherwise StartMass will remain as it was before the command.

MeasurementMultSkipAutoProtect [Protocol 1.4]

Parameters:

ProtectScans Number of scans of consecutive full scale readings before skipping mass. 0 to disable.

Response:

```
MeasurementMultSkipAutoProtect<ws>OK<crLf>
<ws>ProtectScans<ws>{0-31}<crLf>
<crLf>
<crCr>
```

Example:

```
MeasurementMultSkipAutoProtect 5

MeasurementMultSkipAutoProtect OK
ProtectScans 5
```

Description:

For instruments with a multiplier it is possible for measurements to monitor the number of consecutive scans that masses return full scale readings and if that number exceeds a set amount then the mass will be skipped on subsequent scans. The aim is to prevent premature ageing of the multiplier by over driving it. To remain compatible and consistent with previous versions the default is 0 for this parameter so no automatic skipping of masses is done. By specifying the number of scans that must be consecutively full scale, the auto skipping of masses can be enabled. Setting a value of 0 disables the auto skip feature.

Remarks:

See MultiplierInfo command reference for details on the default values that the auto multiplier protection uses and also the MultAutoSkip event response which is returned when a mass is detected that will be skipped for future scans.

MeasurementMultSkipMassAdd [Protocol 1.4]

Parameters:

Mass The mass to skip when using the multiplier

Response:

```
MeasurementMultSkipMassAdd<ws>OK<crlf>
<ws>Mass<ws>{MassToSkip}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementMultSkipMassAdd 40
```

```
MeasurementMultSkipMassAdd  OK
Mass 40
```

Description:

Adds a mass to the list of masses to be skipped by the selected measurement when the multiplier is being used.

Remarks:

Explicitly adds a mass to the list of masses to be skipped when multiplier is being used. The mass will always be skipped by the measurement when using the multiplier even if the parameters of the measurement are changed with the exception of changes to the mass range of a measurement meaning the mass is no longer scanned. The SinglePeak measurement will begin scanning a mass again when any new mass value is set for the measurement even if it is in the same AMU as was previously set.

MeasurementMultSkipMassRemove [Protocol 1.4]

Parameters:

Mass The mass to re-enable multiplier data for on the measurement

Response:

```
MeasurementMultSkipMassRemove<ws>OK<crlf>
<ws>Mass<ws>{NewStartMass}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementMultSkipMassRemove 40
```

```
MeasurementMultSkipMassRemove OK
Mass 40
```

Description:

Removes a mass from the multiplier skip list so that it will be scanned again with the multiplier detector.

Remarks:

This command can remove a mass from the multiplier skip list whether it was added via the MeasurementMultSkipMassAdd command or via the automatic protection (see MeasurementMultSkipAutoProtect command).

MeasurementMultSkipMassRemoveAll [Protocol 1.4]

Parameters:

None

Response:

```
MeasurementMultSkipMassRemoveAll<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
MeasurementMultSkipMassRemoveAll
```

```
MeasurementMultSkipMassremvoveAll  OK
```

Description:

Removes all masses from the multiplier skip list so that they will again be scanned with the multiplier detector.

Remarks:

MeasurementRemoveAll

Parameters:

None

Response:

```
MeasurementRemoveAll<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
MeasurementRemoveAll
```

```
MeasurementRemoveAll OK
```

Description:

Removes all measurements from the sensor.

Remarks:

Must be in control of the sensor and no scans can be running for this command to succeed.

MeasurementRemove

Parameters:

MeasurementName Name of the measurement to remove

Response:

```
MeasurementRemove<ws>OK<crlf>
<ws>Measurement<ws>{Name}<crlf>
<crlf>
<crcl>
```

Example:

```
MeasurementRemove Barchart1
```

```
MeasurementRemove OK
Measurement Barchart1
```

Description:

Removes the specified measurement from the sensor.

Remarks:

Must be in control of the sensor and no scans can be running for this command to succeed.

SourceIonEnergy

Parameters:

SourceIndex	0 based index of the source parameters entry to modify
IonEnergy	New ion energy value

Response:

```
SourceIonEnergy<ws>OK<crlf>
<crlf>
<crcr>
```

Example:

```
SourceIonEnergy 0 5.5
```

```
SourceIonEnergy OK
```

Description:

Sets a source settings parameters Ion Energy setting. This is only valid if the sensor has a configurable ion source.

Remarks:

IonEnergy is valid in range 0 – 10 eV.

SourceEmission

Parameters:

SourceIndex	0 based index of the source parameters entry to modify
Emission	New emission value.

Response:

```
SourceEmission<ws>OK<crlf>
<crlf>
<crcl>
```

Example:

```
SourceEmission 0 1.0

SourceEmission OK
```

Description:

Sets a source settings parameters Emission setting. This is only valid if the sensor has a configurable ion source.

Remarks:

Emission current is valid in range 0 – 5 mA.

SourceExtract

Parameters:

SourceIndex	0 based index of the source parameters entry to modify
Extract	New extract value.

Response:

```
SourceExtract<ws>OK<crlf>
<crlf>
<crcl>
```

Example:

```
SourceExtract 0 -112

SourceExtract OK
```

Description:

Sets a source settings parameters Extract setting. This is only valid if the sensor has a configurable ion source.

Remarks:

Extract volts is valid in range -130 – 0 V.

SourceElectronEnergy

Parameters:

SourceIndex	0 based index of the source parameters entry to modify
ElectronEnergy	New electron energy value

Response:

```
SourceElectronEnergy<ws>OK<crlf>
<crlf>
<crcl>
```

Example:

```
SourceElectronEnergy 0 70

SourceElectronEnergy OK
```

Description:

Sets a source settings parameters Electron Energy setting. This is only valid if the sensor has a configurable ion source.

Remarks:

Electron Energy is valid in range 0 – 100 eV.

SourceLowMassResolution

Parameters:

SourceIndex 0 based index of the source parameters entry to modify
LowMassResolution New low mass resolution value

Response:

```
SourceLowMassResolution<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
SourceLowMassResolution 0 32767  
  
SourceLowMassResolution OK
```

Description:

Sets a source settings parameters Low Mass Resolution setting.

Remarks:

Valid setting range 0 - 65535

SourceLowMassAlignment

Parameters:

SourceIndex 0 based index of the source parameters entry to modify
LowMassAlignment New low mass alignment value

Response:

```
SourceLowMassAlignment<ws>OK<crLf>  
<crLf>  
<crCr>
```

Example:

```
SourceLowMassAlignment 0 32767  
  
SourceLowMassAlignment OK
```

Description:

Sets a source settings parameters Low Mass Alignment setting.

Remarks:

Valid setting range 0 - 65535

SourceHighMassAlignment

Parameters:

SourceIndex 0 based index of the source parameters entry to modify
HighMassAlignment New high mass alignment value

Response:

```
SourceHighMassAlignment<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
SourceHighMassAlignment 0 32767  
  
SourceHighMassAlignment OK
```

Description:

Sets a source settings parameters High Mass Alignment setting.

Remarks:

Valid setting range 0 - 65535

SourceHighMassResolution

Parameters:

SourceIndex 0 based index of the source parameters entry to modify
HighMassResolution New high mass resolution value

Response:

```
SourceHighMassResolution<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
SourceHighMassResolution 0 32767  
  
SourceHighMassResolution OK
```

Description:

Sets a source settings parameters High Mass Resolution setting.

Remarks:

Valid setting range 0 - 65535

AnalogInputAverageCount [protocol 1.2]

Parameters:

Index The index of the analog input
NumberToAverage The number of readings that should be averaged before returning result

Response:

```
AnalogInputAverageCount<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
AnalogInputAverageCount 0 5

AnalogInputAverageCount OK
```

Description:

Sets the number of readings that should be taken and averaged before results are sent back. The time between readings is the reading interval multiplied by this count. The default is for readings to be taken every 0.1s and 10 readings averaged so that a reading is returned every second.

Remarks:

AnalogInputEnable [protocol 1.2]

Parameters:

Index	The index of the analog input
Enable	True/False to enable or disable the analog input

Response:

```
AnalogInputEnable<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
AnalogInputEnable 0 true

AnalogInputEnable OK
```

Description:

Enables or disables analog input readings from being sent when in control of the sensor

Remarks:

AnalogInputInterval [protocol 1.2]

Parameters:

Index	The index of the analog input
Interval	Time in microseconds between successive analog input readings

Response:

```
AnalogInputInterval<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
AnalogInputInterval 0 100000

AnalogInputInterval OK
```

Description:

Sets the interval between analog input readings in the sensor. Time is specified in microseconds. The default is 100000 or 0.1s. The sensor measures the analog input at this frequency and averages a number of results before sending the value back in the AnalogInput event response. The number of readings that are averaged is specified using the AnalogInputAverageCount command.

Remarks:

AnalogOutput [protocol 1.2]

Parameters:

Index	The index of the analog output
Value	The value to set the analog output to

Response:

```
AnalogOutput<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
AnalogOutput
```

```
AnalogOutput OK
```

Description:

Sets a given analog output channel to the specified voltage.

Remarks:

AudioFrequency [protocol 1.2]

Parameters:

Frequency The frequency in Hz to drive the sensors audio output

Response:

```
AudioFrequency<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
AudioFrequency 1000
```

```
AudioFrequency OK
```

Description:

If the sensor supports audio output then the frequency of the audio output can be driven manually.

Remarks:

The audio mode must be manual for this command to have any effect.

AudioMode [protocol 1.2]

Parameters:

Mode The mode to run the audio in.

Response:

```
AudioMode<ws>OK<crlf>
<crlf>
<crcl>
```

Example:

```
AudioMode Manual
```

```
AudioMode OK
```

Description:

If the sensor supports audio output then this command changes the mode between:

- Off No audio output
- Automatic Audio will change based on what is being monitored (useful for leakcheck)
- Manual The frequency output can be changed manually

Remarks:

CirrusCapillaryHeater [protocol 1.2]

Parameters:

HeatOn True/False to turn heater on/off

Response:

```
CirrusCapillaryHeater<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
CirrusCapillaryHeater  
  
CirrusCapillaryHeater OK
```

Description:

Turns the capillary heater on/off. This will likely result in a CirrusStatus event message to reflect the current status of the Cirrus.

Remarks:

CirrusHeater [protocol 1.2]

Parameters:

Mode State to put heater into: Off, Warm or Bake

Response:

```
CirrusHeater<ws>OK<crlf>  
<crlf>  
<crcl>
```

Example:

```
CirrusHeater Warm
```

```
CirrusHeater OK
```

Description:

Sets the cirrus heater into the mode requested. This will likely result in a CirrusStatus event message to reflect the new state of the cirrus.

Remarks:

CirrusPump [protocol 1.2]

Parameters:

PumpOn True/False to turn pump On/Off

Response:

```
CirrusPump<ws>OK<crLf>  
<crLf>  
<crCr>
```

Example:

```
CirrusPump
```

```
CirrusPump OK
```

Description:

Turns the cirrus pumps on or off. This will result in CirrusStatus event message to reflect the state of the cirrus as the pumps get up to speed or turn off.

Remarks:

CirrusValvePosition [protocol 1.2]

Parameters:

ValvePos 0 based valve position

Response:

```
CirrusValvePosition<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
CirrusValvePosition 1  
  
CirrusValvePosition OK
```

Description:

Moves the cirrus rotary valve to the specified position. CirrusStatus event messages will be sent as the valve moves into position.

Remarks:

The cirrus must be configured to have a rotary valve or this command will fail.

DigitalMaxPB67OnTime [protocol 1.2]

Parameters:

Time Time in seconds for port B bits 6 and 7 to remain set

Response:

```
DigitalMaxPB67OnTime<ws>OK<crLf>  
<crLf>  
<crCr>
```

Example:

```
DigitalMaxPB67OnTime 600
```

```
DigitalMaxPB67OnTime OK
```

Description:

Sets the time that either pin 6 or 7 will remain set for after they are initially set. The time is specified in seconds.

Remarks:

This command is supported to keep compatibility with a customer special feature. Currently it only works with MicroVision+ and IP sensors.

DigitalOutput [protocol 1.2]

Parameters:

Port The port name, A, B, C, etc.
Value The value to set outputs to. 8 bit number 0 – 255

Response:

```
DigitalOutput<ws>OK<crLf>  
<crLf>  
<crCr>
```

Example:

```
DigitalOutput A 192  
  
DigitalOutput OK
```

Description:

Sets digital outputs according to the value specified. The OutputMask and ConnectedMask will be used to ensure that only valid output bits get set regardless of the value specified.

Remarks:

PECal_DateMsg [protocol 1.2]

Parameters:

Date in yyyy-mm-dd_HH:MM:SS format
Message Text message to be displayed when calibration is run

Response:

```
PECal_DateMsg<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
PECal_DateMsg  
  
PECal_DateMsg OK
```

Description:

This is a message specifically for MKS Process Eye software to maintain compatability with some of the softwares calibration features.

Remarks:

PECal_Flush [protocol 1.2]

Parameters:

None

Response:

```
PECal_Flush<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
PECal_Flush
```

```
PECal_Flush OK
```

Description:

This is a message specifically for MKS Process Eye software to maintain compatability with some of the softwares calibration features. It flushes the selected calibration settings to persistent storage of the sensor.

Remarks:

PECal_Inlet [protocol 1.2]

Parameters:

Inlet1
Inlet2
Inlet3

Response:

<ws>OK<crLf>
<crLf>
<crCr>

Example:

PECal_Inlet 1.0 1.0 1.0

PECal_Inlet OK

Description:

This is a message specifically for MKS Process Eye software to maintain compatability with some of the softwares calibration features.

Remarks:

PECal_MassMethodContribution [protocol 1.2]

Parameters:

Mass
Method
Contribution

Response:

```
PECal_MassMethodContribution<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
PECal_MassMethodContribution 28 0 80.5  
  
PECal_MassMethodContribution OK
```

Description:

This is a message specifically for MKS Process Eye software to maintain compatability with some of the softwares calibration features.

Remarks:

PECal_Pressures [protocol 1.2]

Parameters:

Response:

```
PECal_Pressures<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
PECal_Pressures  
  
PECal_Pressures OK
```

Description:

This is a message specifically for MKS Process Eye software to maintain compatability with some of the softwares calibration features.

Remarks:

PECal_Select [protocol 1.2]

Parameters:

SourceIndex
DetectorIndex

Response:

```
PECal_Select<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
PECal_Select 0 0  
  
PECal_Select OK
```

Description:

This is a message specifically for MKS Process Eye software to maintain compatability with some of the softwares calibration features.

Remarks:

RolloverScaleFactor [protocol 1.2]

Parameters:

Mass	The mass to set a specific peak scale factor for
Factor	The peak scale factor for the mass

Response:

```
RolloverScaleFactor<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
RolloverScaleFactor 28 5.2
```

```
RolloverScaleFactor OK
```

Description:

Sets a given masses peak scale factor to compensate for differences in sensitivity to the rollover effect.

Remarks:

This command is only available if the sensor supports rollover correction. See the response from the Info command.

RolloverVariables [protocol 1.2]

Parameters:

M1
M2
B1
B2
BP1

Response:

```
RolloverVariables<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
RolloverVariables -470 -250 -0.15 -0.91 0.0012  
  
RolloverVariables OK
```

Description:

Sets the key rollover algorithm constants. This algorithm is propriety to MKS and it is not expected that third party software would edit these values.

Remarks:

RVCAlarm [protocol 1.2]

Parameters:

State True/False value whether the alarm output should be set on or off.

Response:

```
RVCAlarm<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
RVCAlarm True
```

```
RVCAlarm OK
```

Description:

Sets or clears the digital alarm output on the RVC

Remarks:

The sensor must have an RVC fitted for this command to be available. See the ExternalHardware field in the Info command response.

RVCCloseAllValves [protocol 1.2]

Parameters:

None

Response:

```
RVCCloseAllValves<ws>OK<crLf>  
<crLf>  
<crCr>
```

Example:

```
RVCCloseAllValves  
  
RVCCloseAllValves OK
```

Description:

Closes the RVC valves.

Remarks:

The sensor must have an RVC fitted for this command to be available. See the ExternalHardware field in the Info command response.

RVCh Heater [protocol 1.2]

Parameters:

HeaterOn True/False value whether to switch the heater on/off.

Response:

```
RVCh Heater<ws>OK<cr lf>
<cr lf>
<cr cr>
```

Example:

```
RVCh Heater True
```

```
RVCh Heater OK
```

Description:

Turns the RVC heater on/off. This will may result in RVCh HeaterStatus event messages coming back from the sensor.

Remarks:

The sensor must have an RVC fitted for this command to be available. See the ExternalHardware field in the Info command response.

RVCPump [protocol 1.2]

Parameters:

PumpOn True/False whether to turn the pump on or off.

Response:

```
<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
RVCPump True
```

```
RVCPump OK
```

Description:

Turns the pump on or off. This may result in RVCPumpStatus event messages to indicate the status of the pump.

Remarks:

The sensor must have an RVC fitted for this command to be available. See the ExternalHardware field in the Info command response.

RVCValveControl [protocol 1.2]

Parameters:

Valve	Index of the valve to open/close. 0,1 or 2
Open	True/False to open or close the valve

Response:

```
RVCValveControl<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
RVCValveControl 0 True
```

```
RVCValveControl OK
```

Description:

Opens or closes a specific valve. Note that the interlocks and valve mode might not allow the requested action immediately. Valve changes may result in RVCValveStatus event messages.

Remarks:

The sensor must have an RVC fitted for this command to be available. See the ExternalHardware field in the Info command response.

RVCValveMode [protocol 1.2]

Parameters:

Mode Manual or Automatic

Response:

```
RVCValveMode<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
RVCValveMode Manual
```

```
RVCValveMode OK
```

Description:

Switches valve mode between manual and automatic mode.

Remarks:

The sensor must have an RVC fitted for this command to be available. See the ExternalHardware field in the Info command response.

SaveChanges [protocol 1.2]

Parameters:

None

Response:

```
SaveChanges<ws>OK<crlf>
<crlf>
<crcr>
```

Example:

```
SaveChanges
```

```
SaveChanges OK
```

Description:

Saves any changes that may have been made to the tuning/calibration of the sensor. This action is done when the connection is released or closed anyway but if you want to ensure that settings are saved before carrying on then this will write any persistent settings to disk or flash memory.

Remarks:

StartDegas [protocol 1.2]

Parameters:

StartPower	Percentage power to start at. Typically 10%
EndPower	Percentage power to ramp to. Typically 85%
RampPeriod	Time in seconds to ramp between StartPower and EndPower. Typically 90s
MaxPowerPeriod	Time to hold at EndPower. Typically 240s
ResettlePeriod	Time to return to default settings. Typically 30s

Response:

```
StartDegas<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
StartDegas 10 85 90 240 30

StartDegas OK
```

Description:

Runs a degas operation. Ramps from StartPower percentage power to EndPower over a period of RampPeriod seconds. The power is then held for MaxPowerPeriod seconds and finally the ion source settings are returned to normal and allowed to settle for ResettlePeriod seconds.

EndPower must be greater than StartPower and they must both be between 0 and 100.

RampPeriod can be between 0 and 600s.

MaxPowerPeriod can be between 0 and 900s.

ResettlePeriod can be between 0 and 1800s.

During the degas operation DegasReading event messages will be returned from the sensor indicating the current stage of the degas operation, the percent power, filament current and total time remaining.

Remarks:

This command is only available if the sensor supports configurable ion source settings and can therefore control the degas operation. See the response from the Info command.

Filaments must be on before the degas operation can be run. If the filaments trip off then the degas operation will end.

StopDegas [protocol 1.2]

Parameters:

None

Response:

```
StopDegas<ws>OK<crLf>  
<crLf>  
<crCr>
```

Example:

```
StopDegas
```

```
StopDegas OK
```

Description:

Ends a degas operation that is currently running.

Remarks:

Sensor Control Commands In Protocol 1.6

SourceAlignmentCopyToAll [protocol 1.6]

Parameters:

SourceIndex The source table to copy data from

Response:

```
SourceAlignmentCopyToAll<ws>OK<crlf>
<crlf>
<crcr>
```

Example:

```
SourceAlignmentCopyToAll 0

SourceAlignmentCopyToAll OK
```

Description:

Copies the specified source tables alignment data to all other source tables.

Remarks:

SourceAlignmentUpdate [protocol 1.6]

Parameters:

SourceIndex	The source table to update alignment data for
Mass	The mass to update or add to the table
DAC	The new DAC value to be assigned to the mass

Response:

```
SourceAlignmentUpdate<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
SourceAlignmentUpdate 0 28 0x8a4

SourceAlignmentUpdate OK
```

Description:

Adds or updates a table entry in the mass alignment table belonging to a specific source table.

Remarks:

The DAC parameter must be within valid limits or the command will fail, see SourceTuningInfo command for details of working out the valid limits for any given mass.

In the example above the DAC value is passed as hexadecimal but it could equally well have been specified as a decimal value. When using hexadecimal the prefix 0x must be used.

SourceAlignmentRemove [protocol 1.6]

Parameters:

SourceIndex	The index of the source table which should have some alignment data removed
Mass	The mass value to remove the entry for

Response:

```
SourceAlignmentRemove<ws>OK<crlf>
<crlf>
<crcr>
```

Example:

```
SourceAlignmentRemove 28

SourceAlignmentRemove OK
```

Description:

Removes a specific mass from the mass alignment table for the specified source table.

Remarks:

There must always be 2 or more entries in the table so an attempt to remove a mass when there are only 2 entries will fail

SourceResolutionCopyToAll [protocol 1.6]

Parameters:

SourceIndex The source table to copy data from

Response:

```
SourceResolutionCopyToAll<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
SourceResolutionCopyToAll 0

SourceResolutionCopyToAll OK
```

Description:

Copies the specified source tables resolution data to all other source tables.

Remarks:

SourceResolutionUpdate [protocol 1.6]

Parameters:

SourceIndex	The source table to update resolution data for
Mass	The mass to update or add to the table
DAC	The new DAC value to be assigned to the mass

Response:

```
SourceResolutionUpdate<ws>OK<crlf>
<crlf>
<crcr>
```

Example:

```
SourceResolutionUpdate 0 28 0x8a4

SourceResolutionUpdate OK
```

Description:

Adds or updates a table entry in the resolution table belonging to a specific source table.

Remarks:

In the example above the DAC value is passed as hexadecimal but it could equally well have been specified as a decimal value. When using hexadecimal the prefix 0x must be used.

SourceResolutionRemove [protocol 1.6]

Parameters:

SourceIndex	The index of the source table which should have some resolution data removed
Mass	The mass value to remove the entry for

Response:

```
SourceResolutionRemove<ws>OK<crlf>
<crlf>
<crcr>
```

Example:

```
SourceResolutionRemove 28

SourceResolutionRemove OK
```

Description:

Removes a specific mass from the resolution table for the specified source table.

Remarks:

There must always be at least 1 entry in the table so an attempt to remove a mass when there is only 1 entry will fail

SourcePoleBias [protocol 1.6]

Parameters:

SourceIndex	The index of the source table which should have some resolution data removed
PoleBias	The polebias value in Volts. Valid values -10 to +10

Response:

```
SourcePoleBias<ws>OK<crlf>
<crlf>
<crcl>
```

Example:

```
SourcePoleBias 0 -5

SourcePoleBias OK
```

Description:

Sets a specific source table entries PoleBias setting

Remarks:

Command only valid for instruments that support PoleBias, see SourceInfo for how to tell if PoleBias is supported or not.

DiagnosticInputAverageCount [protocol 1.6]

Parameters:

Index The index of the analog input
NumberToAverage The number of readings that should be averaged before returning result

Response:

```
DiagnosticInputAverageCount<ws>OK<crlf>  
<crlf>  
<crcr>
```

Example:

```
DiagnosticInputAverageCount 0 1  
  
DiagnosticInputAverageCount OK
```

Description:

Sets the number of readings that should be taken and averaged before results are sent back. The time between readings is the reading interval multiplied by this count. The default is for readings to be taken every 15s and 1 readings averaged so that a reading is returned every 15 seconds.

Remarks:

DiagnosticInputEnable [protocol 1.6]

Parameters:

Index	The index of the diagnostic input
Enable	True/False to enable or disable the diagnostic input

Response:

```
DiagnosticInputEnable<ws>OK<crlf>
<crlf>
<crcr>
```

Example:

```
DiagnosticInputEnable 0 true

DiagnosticInputEnable OK
```

Description:

Enables or disables diagnostic input readings from being sent when in control of the sensor

Remarks:

DiagnosticInputInterval [protocol 1.6]

Parameters:

Index	The index of the diagnostic input
Interval	Time in microseconds between successive diagnostic input readings

Response:

```
DiagnosticInputInterval<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
DiagnosticInputInterval 0 100000

DiagnosticInputInterval OK
```

Description:

Sets the interval between diagnostic input readings in the sensor. Time is specified in microseconds. The default is 15000000 or 15s. The sensor measures the diagnostic input at this frequency and averages a number of results before sending the value back in the DiagnosticInput event response. The number of readings that are averaged is specified using the DiagnosticInputAverageCount command.

Remarks:

Sensor Control Commands In Protocol 1.7

InletSelect [protocol 1.7]

Parameters:

InletIndex The inlet to make active

Response:

```
InletSelect<ws>OK<crLf>
<crLf>
<crCr>
```

Example:

```
InletSelect 0

InletSelect OK
```

Description:

When inlets are configured to not be automatic (see InletInfo) this command allows selecting the appropriate inlet.

Remarks:

Asynchronous Sensor Notifications

As data is acquired or other events occur within the sensor (e.g. filament state changes, digital inputs etc.) it will send out a notification. The format of each notification is determined by its type so a client application must look at the notification name in order to determine how to handle it.

MKSRGA

Response:

```
MKSRGA<ws>{Type}<crLf>
<ws>Protocol_Revision<ws>{ProtocolRev}<crLf>
<ws>Min_Compatibility<ws>{Compatibility}<crLf>
<crLf>
<crCr>
```

Example:

```
MKSRGA  Multi
        Protocol_Revision 1.2
        Min_Compatibility 1.4
```

Description:

This is the first thing sent to a client when a connection is made to tcp/ip port 10014 of the sensor. It is used to indicate that we have connected to a valid RGA sensor and also give a little information about the class of sensor and protocol version that it supports. From this clients should be able to decide whether they can communicate successfully with the sensor.

Type can be either Single or Multi. Single indicates that the sensor is a standalone tcp/ip unit, Multi indicates that the sensor is actually a server application handling multiple sensors for example a windows server managing multiple MicroVision+ units connected to the PC's serial ports.

When a sensors Type is Single the sensor will already be selected so there is no need for a client to use the Sensors command or Select command, however both commands are functional if a client wants to remain compatible with all RGA sensor types.

Protocol_Revision indicates the current version of the protocol supported by the sensor.

Min_Compatibility indicates the earliest protocol revision that this version is compatible with. Clients should check this value against the version they were written for and if it is lower or equal then they should have no problem communicating. If it is higher then clients should disconnect and notify the user that an updated version will be required to work with this sensor.

FilamentStatus [updated in protocol 1.6]

Response:

```
FilamentStatus<ws>{FilamentNumber}<ws>{SummaryState}<crlf>
<ws>Trip<ws>{Trip}<crlf>
<ws>Drive<ws>{On/Off}<crlf>
<ws>EmissionTripState<ws>{OK/Fail}<crlf>
<ws>ExternalTripState<ws>{OK/Fail}<crlf>
<ws>RVCTripState<ws>{OK/Fail}<crlf>
<ws>GaugeTripState<ws>{OK/Fail}<crlf>    **Protocol 1.6
<crlf>
<crcl>
```

Example:

```
FilamentStatus 1      OFF
  Trip                None
  Drive               Off
  EmissionTripState   OK
  ExternalTripState   OK
  RVCTripState        OK
```

Description:

This message is sent whenever the state of the filaments changes. See the FilamentInfo command description for details of the parameters and values contained within the message.

FilamentTimeRemaining

Response:

```
FilamentTimeRemaining<ws>{Time}<crlf>  
<crcr>
```

Example:

```
FilamentTimeRemaining 890
```

Description:

If the filaments are configured to have a maximum time that they will stay on for then these messages will be sent while the filaments are on every 2 seconds to update the client with the time remaining before the filaments will go off.

See the FilamentOnTime command for details on how to reset the time.

StartingScan

Response:

```
StartingScan<ws>{ScanNumber}<ws>{Time}<ws>{ScansRemaining}<crLf>  
<crCr>
```

Example:

```
StartingScan 2 16858 0
```

Description:

When a new scan starts, it indicates the scan number, time in ms relative to the first scan and the number of scans left before it must be re-triggered. Time is in milli-seconds.

StartingMeasurement

Response:

```
StartingMeasurement<ws>{MeasurementName}<crlf>  
<cr>
```

Example:

```
StartingMeasurement Barchart1
```

Description:

Indicates the name of the measurement that is starting being scanned. This message gives a client a chance to locate the appropriate measurement and ensure that all ZeroReading and MassReading messages have their data put in the correct place.

ZeroReading

Response:

```
ZeroReading<ws>{MassPosition}<ws>{Value}<crlf>  
<cr>
```

Example:

```
ZeroReading 5.5 1.01e-8
```

Description:

The zero reading value for the currently scanning measurement if it is a partial pressure measurement

MassReading [changed in protocol 1.4]

Response:

```
MassReading<ws>{MassPosition}<ws>{Value}<crlf>  
<cr>
```

Example:

```
MassReading 1 2.9383e-5
```

Description:

The reading for a particular mass for the currently scanning measurement.

Remarks:

As of protocol version 1.4 the {Value} section can be replaced by the word MultSkipped which indicates that the mass is one that is being skipped to protect the multiplier.

MultAutoSkip [Protocol 1.4]

Response:

```
MultAutoSkip<ws>{Mass}<crlf>  
<cr>
```

Example:

```
MultAutoSkip 40
```

Description:

When a measurement has the MeasurementMultSkipAutoProtect setting enabled and is running with the multiplier, if a mass is seen to have enough consecutive scans at full scale then it will be skipped in future scans and this message is sent to notify the client of this change.

Remarks:

This message will always be sent following the last reading that was taken with the multiplier on and measured as full scale.

MultiplierStatus [updated in protocol 1.6]

Response:

```
MutliplierStatus<ws>{Off/On}<crlf>  
<ws>LockedByFilament<ws>{Yes/No}<crlf>  
<ws>LockedByRVC<ws>{Yes/No}<crlf>  
<ws>LockedBySoftware<ws>{Yes/No}<crlf>  
<ws>HardwareTripped<ws><crlf>          ** Added in protocol 1.6  
<cr>
```

Example:

```
MutliplierStatus      Off  
  LockedByFilament    Yes  
  LockedByRVC         No  
  LockedBySoftware    No
```

Description:

Sent when the multiplier status changes. See the MultiplierInfo command for details on the configuration of the multiplier and what the values mean.

RFTripState

Response:

```
RFTripState<ws>{State}<crlf>  
<cr>
```

Example:

```
RFTripState Tripped
```

Description:

Sent whenever the RF trip state changes. See the RFInfo command for details of the current configuration for RF trips. State can be OK or Tripped.

InletChange [protocol 1.2]

Response:

```
InletChange<ws>{Index}<crlf>  
<crcr>
```

Example:

```
InletChange 0
```

Description:

Sent whenever the active inlet changes. For systems with RVC/VSC this message will get sent as valves open and close.

AnalogInput [protocol 1.2, 1.6]

Response:

```
AnalogInput<ws>{Index}<ws>{Value}<crLf>  
<crCr>
```

Example:

```
AnalogInput 0 5.6
```

Description:

Sent whenever an analog input is read based upon its average count and cycle time settings.

Remarks:

Protocol 1.6 can now indicate if the input reading is invalid (no readings taken yet), under-range or over-range (the analog input reading was outside the limits set in the control unit configuration). This information is made available when the AcceptProtocol command has been sent with the protocol version 1.6 or greater, in these cases the numeric value for the input reading will be Invalid, UnderRange or OverRange.

MicroVision2 supports a user configured conversion function to convert analog input data from voltage to some other logical unit. Currently not all the information about the conversion function is made available via the protocol but it does mean that analog input readings might not always be between the MinVolts and MaxVolts values.

TotalPressure [protocol 1.2, 1.6]

Response:

```
TotalPressure<ws>{Value}<crLf>  
<crCr>
```

Example:

```
TotalPressure 1.0e-4
```

Description:

Sent whenever the total pressure is read from an external total pressure gauge.

Remarks:

Protocol 1.6 can now indicate if the reading is invalid (no readings taken yet), under-range or over-range (the gauge reading was outside the limits set in the control unit configuration). This information is made available when the AcceptProtocol command has been sent with the protocol version 1.6 or greater, in these cases the numeric value for the input reading will be Invalid, UnderRange or OverRange.

DigitalPortChange [protocol 1.2]

Response:

```
DigitalPortChange<ws>{Port}<ws>{Value}<crLf>  
<crCr>
```

Example:

```
DigitalPortChange A 175
```

Description:

Sent whenever a digital port value changes

RVCPumpStatus [protocol 1.2]

Response:

```
RVCPumpStatus<ws>{On/Off/Accelerating}<crlf>  
<cr>
```

Example:

```
RVCPumpStatus Accelerating
```

Description:

Sent whenever the pump state changes

RVHeaterStatus [protocol 1.2]

Response:

```
RVHeaterStatus<ws>{On/Off/CoolingDown}<crlf>  
<cr>
```

Example:

```
RVHeaterStatus CoolingDown
```

Description:

Sent whenever the heater status changes

RVCValveStatus [protocol 1.2]

Response:

```
RVCValveStatus<crLf>
<ws>Valve0<ws>{Open/Closed}
<ws>Valve1<ws>{Open/Closed}
<ws>Valve2<ws>{Open/Closed}
<crCr>
```

Example:

```
RVCValveStatus
  Valve0      Open
  Valve1      Closed
  Valve2      Closed
```

Description:

Sent whenever the valve states change

RVCInterlocks [protocol 1.2]

Response:

```
RVCInterlocks<ws>{On/Off}<crlf>  
<crcr>
```

Example:

```
RVCInterlocks On
```

Description:

Sent whenever the interlock state of the RVC changes as a result of the key switch being turned

RVCStatus [protocol 1.2]

Response:

```
RVCStatus<crlf>  
<ws>Status0<ws>{True/False}<crlf>  
<ws>Status1<ws>{True/False}<crlf>  
<cr>
```

Example:

```
RVCStatus  
    Status0    True  
    Status1    False
```

Description:

RVCDigitalInput [protocol 1.2]

Response:

```
RVCDigitalInput<crLf>  
<ws>DigitalInput0<ws>{True/False}<crLf>  
<ws>DigitalInput1<ws>{True/False}<crLf>  
<crCr>
```

Example:

```
RVCDigitalInput  
    DigitalInput0      True  
    DigitalInput1      False
```

Description:

Sent whenever general purpose RVC digital inputs change

LinkDown [protocol 1.2]

Response:

```
LinkDown<ws>{Reason}<crlf>  
<cr>
```

Example:

```
LinkDown Serial
```

Description:

Sent when the link to the sensor has been lost or is no longer able to function reliably. The reasons for LinkDowns are:

Serial	The serial link between a PC and the MicroVision+ sensor has been lost
VSC	The link between a MicroVision IP and it's VSC has been lost. Until the link can be made again it won't be possible to control the sensor.

VSCEvent [protocol 1.2]

Response:

```
VSCEvent<crLf>
<ws>Data<ws>{Value}<crLf>
<ws>Register<ws>{RegisterNumber}<crLf>
<ws>Time<ws>{EventTime}<crLf>
<crCr>
```

Example:

```
VSCEvent
  Data      1234
  Register  405
  Time      6831
```

Description:

Sent when the sensor receives event notifications from the VSC.

DegasReading [protocol 1.2]

Response:

```
DegasReading<ws>{Ramping/Holding/Recover/Complete}<crLf>
<ws>PercentPower<ws>{Value}<crLf>
<ws>FilamentCurrent<ws>{Value}<crLf>
<ws>TimeRemaining<ws>{Seconds}<crLf>
<crCr>
```

Example:

```
DegasReading  Ramping
PercentPower   10.0
FilamentCurrent 0.039063
TimeRemaining  10
```

Description:

Sent when a new reading is available for the Degas mode. See StartDegas for details.

DiagnosticInput [protocol 1.6]

Response:

```
DiagnosticInput<ws>{ChannelIndex}<ws>{ReadingValue}<crLf>
<crCr>
```

Example:

```
DiagnosticInput  0 3.2745
```

Description:

Sent when a new reading is available for the Diagnostic Input. See DiagnosticInputInfo for more details.