Agilent Technologies

1200 Series
Communication Interface

1200 Series
Interface Description

PROTOCOLS
and
INSTRUCTIONS

for

Agilent 1200 SERIES HPLC MODULES

Waldbronn Analytical Division

76337 Waldbronn
Germany

September 11, 2007

Table of Contents

10. Method and Data Handling 151

11. Method Parameter 170

12. Common System Parameter 174

13. Commands & Queries 177

18. Diagnostics 264

22. Detector Instructions (Common) 549

23. Diode-Array Detector (G1315B) MultiWavelength Detector (G1365B) 563

26. Refractive Index Detector (G1362A) 777

1. Introduction

1.1. Modules in this Manual

This document contains information about how to control instruments which are based on the 1200 Series communication architecture.
Following is the list of modules that belong to this family (as of April 2006):

1200 PUMP	Isocratic Pump	G 1310 A
	QuarternaryPump	G 1311 A
	Binary Pump	G 1312 A
	High Performance Gradient Pump -SL	G 1312 B
1200 ALS	Autosampler/Injector	G 1313 A
	Thermostatted Autosampler/Injector	G 1329 A
1200 VWD	Variable Wavelength Detector	G 1314 B
	Variable Wavelength Detector - SL	G 1314 C
1200 MWD	Multiple Wavelength Detector	G 1365 A/B
1200 DAD	Diode Array Detector	G 1315 A/B
1200 TCC	Thermostatted Column Compartment	G 1316 A
	Thermostatted Column Compartment - SL	G 1316 B
1200 FLD	Fluorescence Detector	G 1321 A
1200 RID	Refractive Index Detector	G 1362 A
8453 SPM	Spectrophotometer	G 1103 A

All these modules are based on a common design for communication and a common way of how to perform an analysis.

1.2. Introduction to Communication

Several modules which are inter-connected via the CAN interface form an instrument, e.g. an LC system. Once set up by a controller, this instrument is capable of performing a whole analysis using the CAN interface for inter-module communication and realtime synchronization independent of the controller.
The instrument can be controlled concurrently by one or more instrument controllers like a PC, the hand-held 1200 Series Controlling Module and/or a laboratory data system.
To communicate to the individual modules, only one connection between the controlling device and one of the modules is needed.
For interfacing, an HPIB (IEEE 488.1) and an RS-232 interface are built-in with every module, whereas other interfaces like LAN connections need an extra interface card to be put in one of the module's MIO interface card slot (see chapter 5. Interfaces on page 67).

To communicate through one interface to several modules and to support different types of interfaces, a special protocol is used. The name of that protocol is LICOP (see chapter 4. LICOP Communication Protocol on page 54).

2. Revision History

2.1. Revision 005, November 19, 2003

G1310/11/12: Seal wash pump option (SWP, SWPD, SWPP).
G1321A: New command DEMUL

2.2. Revision 006, May 14, 2004

Update, minor fixes

2.3. Revision 007, June 24, 2004

Update, minor fixes

2.4. Revision 008, November 16, 2004

Update

2.5. Revision 009, October 4, 2005

Update of common detecor commands.
added missing reply codes.
G1313A (Std-ALS): removed description of commands: "TRAY:<cmd>" - internal use only.

2.6. Revision 010, December 12, 2005

Update.

2.7. Revision 011, March 31, 2006

Introduction of Agilent 1200 Series modules. The revision #011 of this document reflects the state of the FW of the Agilent 1200 series modules rev. A.06.02.

2.8. Revision 013, May 26, 2006

Added description of FW-interface of LC-/Chip-MS

2.9. Revision 014, July 27, 2007

PPRG-documentation of FLD is replaced by PRPR, POPR. PRAU, PRMO are not applicable for FLD.
Standard-Autosampler: New commands ACT:DRPO?, ACT:INPO?.
Update.
Common functionality: "Start not ready". ACT:SRDY?, ES 0128/0129.
New Cosy-Error code ES 0309 to supervise programming errors.

2.10. Revision 015, September 11, 2007

changed description of CBLT (G1310A, G1311A)
and CCBL (G1312A, G1312B)

3. Communication Units

The module’s logical interface is represented by communication units (CU). Communication with a module is done by sending a message to a CU or reading the message out of a CU. The CUs are of either In/Out type or Out type. In/Out type means that CUs of this type are able to receive messages as well send own messages, where Out-CUs only send messages. The CUs are not synchronized, that means that during a listing from List CU a reply can arise from Instruction/Reply CU. A detailed description of the CUs is given in the next chapters. The following table summarizes the CUs for all modules:

Module	CU Name	Description	Direction	max buffer size in bytes
pumps	IN	instruction/reply	in/out	1024/2048
	LI	list	out	256
G1310A	EV	event	out	80
G1311A	MO	monitor	out	512
G1312A	DI	diagnosis	out	512
G1312B	RD	rawdata	out	1024
ALS	IN	instruction/reply	in/out	1024/2048
	LI	list	out	256
G1313A	EV	event	out	80
	MO	monitor	out	512
	DI	diagnosis	out	512
VWD	IN	instruction/reply	in/out	1024/2048
	LI	list	out	256
G1314A	EV	event	out	80
G1314B	MO	monitor	out	512
G1314C	DI	diagnosis	out	512
	RD	rawdata	out	2600
	MS	spectra monitor	out	4216
DAD	IN	instruction/reply	in/out	1024/2048
G1315A	LI	list	out	256
G1315B	EV	event	out	80
	MO	monitor	out	512
MWD	DI	diagnosis	out	512
G1365A	RD	rawdata	out	4216
G1365B	MS	spectra monitor	out	4216
SPM				
G1103A				
COL-COMP	IN	instruction/reply	in/out	1024/2048
	LI	list	out	256
	EV	event	out	80
G1316A	MO	monitor	out	512
G1316B	DI	diagnosis	out	512
	RD	rawdata	out	1024

Module	CU Name	Description	Direction	max buffer size in bytes
FLD G1321A	IN LI EV MO DI RD MS	instruction/reply list event monitor diagnosis rawdata spectra monitor	in/out out out out out out out	1024/2048 256 80 512 512 4124 4124
RID G1362A	IN LI EV MO DI RD	instruction/reply list event monitor diagnosis rawdata	in/out out out out out out	1024/2048 256 80 512 512 272
all modules used as CAP	CS_IO	CanSlave IO	in/out	250/250

3.1. Instruction/Reply CU

Messages to the instruction CU are interpreted as instructions to control the module (e.g set parameters, start analysis ...). Each message results in a reply that is available in the reply CU. Instructions are described in the instruction lists for the different modules. It is necessary to read the reply of an instruction before sending another instruction.

All replies from the CU to an external controller are preceeded by 8 ASCII characters which are:

RA nnnn if the Reply is Accepted, the number returned may specify informational message

RE nnnn if there is a Reply Error to be sent, the error number is given by the four digits 'nnnn', typical errors are syntax and parameter range errors.

See the instruction lists of the modules for the accepted instructions and possible reply errors.

A message to the instruction CU may consist of multiple instructions separated by semicolon. In this case the resulting reply will be the reply of the last instruction contained in the message. If a reply error occurs inside multiple instructions, all following instructions are ignored, the reply will be the reply of the instruction that caused the error.
Note: Multiple instructions inside one message must be separated by semicolon.

In case of a multiline instruction the following 'disposition' characters as last character of the instruction allow to modify the reply:

& default, description see above
! return all replies
* discard all replies (e.g return only RA 0000 or RE nnnn respectively)

3.2. List CU

The List CU is used to get listings generated by the module. The listings are offered in one or more messages. For example, a small time table listing might fit in one message, a large time table listing may need several messages. There are as many instructions inside one message until there is no

more space left for the next complete instruction. Thus there will never be a truncated instruction at the end of a message.
Once a message is filled up, a new message is started to be filled. Each message starts with a header. The header defines what is being listed and it contains a <message number> for each message and the time when the listing was generated. With each listing the record number starts with '0001' if the requested data exists, or with ’0000’ when no data is to be listed, e.g if the timetable is empty or a cosy list is empty. The message number is incremented by one for each new message of the listing. The last message is indicated by the letter 'E' or ’X’ as the third character of the header, if the message is not the last the 3rd character is blank. At least one message is generated per valid request. The very last message of a request is marked with the ’X’, in a list request with multiple list items, the last message of each item is marked with the ’E’. If only one item is specified, the ’X’ is used to mark the last message.

The list data format is:

LI<E or X or blank><record number> <item to list>,<time><crlf><list data>
the <list data> are instructions separated by semi colon;

The instruction to request a listing is:

LIST <"item [,item[,...]]">

where item is the ASCII identifier of what to list.(see table below)
any order of items is allowed, messages are generated according
to this order. A reply error of 100 means that a listing is still in progress.

Example:

LIST "HOME, TT"

means: list the home values and the time table of the current method.
at least two messages are generated, one for the home values and at
least one for the time table.

The following table summarizes the list capabilities:

item name	parameters
HOME	actual method’s home values
TT	actual method’s timetable
SYS	system variables
*	same as "HOME, TT, SYS"
COSY:<name>	the cosy list named <name>
COSY:*	list all cosy lists
METH:<name>	the stored parameters named <name>
METH	list all stored method names

Additionally there are several list capabilities that are private to the modules.

3.3. Event CU

This CU provides events that are generated by the module. Events are used to tell the outer world, that something happened within the module, that is considered worth to be mentioned. This can be a state change, any error or a module start. An event consists of a prefix, an event number, a time stamp (optional) and a parameter (optional). The prefix tells what type of event it is, the number identifies the event itself. The event number is a four digit number, a summary of all events common to the *Agilent 1200 Series devices* is found in chapter "Agilent 1200 Series Events" in this document (page 881). The numbers are unique over all modules.

The event data format is:

E<event id character> <event number>, <time> [, <parameter]

where the <event id character> is one of the following:
S: statechange event
C: reset event
F: limit event
I: injector program events (includes column compartment valve)
E: error event (remark: event numbers are unambigous over devices)
V: info event (e.g modthod modified, analysis done)

An example for event data is:
ES 0104 which indicates a statechange to "RUN".

The event CU stores up to the last 20 events for each controller connected to the instrument using first-in first-out. One event is transmitted each time the event CU is read. If the storage space is filled up so that only one event's space is left, the event EV 0067, <time>, <kind of event> is inserted in the queue, further events are skipped.

kind of event	prefix	id if lost
error event	EE	0
state change event	ES	1
reset event	EC	2
info event	EV	3
inj prog event	EI	4
limit event	EF	5

The event numbers ares sorted according to the module that did generate the event. Common events may be generated by any Agilent 1200 series module.

from	to	module
0001	0999	common events
1000	1999	DAD, MWD, Spectrophotometer

from	to	module
2000	2799	PUMP
2800	3000	COL COMP
4000	4999	ALS
7000	7999	VWD / Detector common

3.4. Monitor CU

This CU periodically provides messages containing information about the module status and instrument parameters that might change during an analysis. The messages are numbered. A monitor message has the following format:

MO<end character><message number> [<actual query reply>;]

where
<end character> is either a blank or ’E’ for the last monitor message
<message number> is a four digit message numbering starting with 0000
<actual query reply> is the reply of an "ACT:*" instruction (e.g. ACT:STAT?)

The following instructions control the CU:

MONI:STRT <period>, <actual items>
starts the CU to periodically generate messages every <period> seconds. The allowed period range is 1 to 99 seconds.
The message numbering starts with 0000. After the message number of '9999' the numbering restarts with '0000'.
The <actual items> are a list of instructions. Allowed are only those instructions that start with "ACT:*" (These are the actual queries).
The list is embedded in double quotes, the instructions must be separated by a colon.

MONI:STOP
Stops the CU from delivering periodically messages.
One last message is generated containing the identification character 'E' in the message prefix, indicating that it is the last monitor message. Nevertheless this last message containes the complete monitor data.

MONI:STAT?
returnes as reply MONI:STAT <on/off>, <period>, <actual items>.
If on/off is 0 the monitor is off, if on/off equals 1 the monitor is on. For period and actual items see MONI:START.

3.5. Diagnostic CU

This CU provides data buffers that are stored in the modules memory. A variety of data buffers exist for each module, see chapter ***Diagnostics***. The diagnostic buffers consist of a header and a series of data entries. The header has the following format:

DI<EndCharacter><message number> <buffer name>, <time of listing>, <time of last entry>, <x-axis spacing>, <info>

<**EndCharacter**> is either ’X’ if the message is the very last of the list command, or ’E’ if the

message is the last for a buffer, or blank.

<**message number**> is the number of the message for the currently listed buffer.

<TimeOfList> is the time in seconds since 1970 when the listing was generated.

<TimeOfLastEntry> is the time in seconds since 1970 when the last entry to the buffer was done.

<XSpace> may contain a number if the buffer holds equidistant data. Same is for **XAxisStart** and **XAxisStartValid**.

<Info> may contain some short explanation for the buffer, by default it is **NoInfo**.

A message number of '0000' indicates an empty buffer. If the buffer contains any entries the message numbers start with '0001' and are incremented by one with each new message. The complete header is repeated with each new message. The data entries that follow the above header have different formats depending on their meaning. See chapter ***Diagnostics*** for a complete list of common diagnostic buffers available in all modules and their data types.

The instructions to control the CU are:

DIAG:LIST <bufnamelist>

<bufnamelist> must be embedded in double quotes, names must be separated by a comma. A reply error of 100 means that the previous listing is still pending (not all messages delivered yet).

DIAG:ABRT

skips the listing of further buffers, but completes the one currently in progress.

3.6. Rawdata CU

This CU provides rawdata out of the run buffer. The data are organized as a file, starting with a header record, followed by several data records and ending with a final stop record. This CU is available on the pumps (G1310A, G1311A, G1312A, G1312B), the Variable Wavelength Detector (G1314A/B/C), the Multiple Wavelength Detector (G1365A/B), the Diode Array Detector (G1315A/B), the Thermostated Column Compartment (G1316A/B) and the 8453A Spectrophotometer (G1103A).

Instructions that control the Rawdata Output are:

RAWD:STRT
RAWD:STOP
RAWD:RSET
RAWD:STAT?
RAWS <..
RAWF <..

The contents of the rawdata is dependend on the device. For the detectors see separate document, for column compartment and the pumps the following table gives a overview of the contents.

3.7. Spectra Monitor CU

This CU periodically offers on-line spectral data and is only available for detectors having this capability (e.g. DAD (G1315A/B), MWD (G1365A/B) and 8453 Spectrophotometer). The spectral data format is the same as for the Rawdata CU except the header, which is:

MS BIN,<number of following data points>;
instead of
SL BIN,<number of following data points>;

Instructions to control the spectra monitor CU are:

SPMO:STRT CU starts to provide spectra periodically
SPMO:STOP CU stops providing spectra
SPMO:PERI sets monitor period

See the module specific document for a detailed description of these instructions.

3.8. CanSlave_IO CU

This CU provides access to the programming interface of any CanSlave. All kinds of messages to and from the CanSlave are exchanged using this CU. The type of a message is determined by a prefix consisting of 2 bytes:

0x494E	Instructions / Replies	In/Out
0x4556	Events	Out only
0x4D4F	Monitor Data	not yet implemented
0x4C49	List data	not yet implemented
0x5244	Raw measurement data	not yet implemented

The messages can pop up in random order, this means that there is no rule when a reply or any event arise. Relpies and Events are randomly mixed.
Each message can carry up to 250bytes of data. All messages follow the format shown below:

	MessageType	UserData
length	2 bytes	<= 250 bytes

For example the instruction "IDN?" looks like (binary coded):

MessageType	UserData
0x49 0x4E	0x49 0x4F 0x4E 0x3F

The relpy has the following form ’RA 0000 IDN "AGILENT TECHNOLOGIES, ..."' (binary coded):

MessageType	UserData
0x49 0x4E	0x52 0x41 0x20 0x20 0x20 0x20 0x49 0x4F 0x4E 0x20 ...

Events are as follow:

MessageType	UserData		
0x45 0x56	0x45 0x43 0x20 0x30 0x35 0x30 0x31	EC 0501	Hotstart
0x45 0x56	0x45 0x43 0x20 0x30 0x35 0x30 0x31	EC 0502	Hotstart

0x45 0x56	0x45 0x53 0x20 0x30 0x35 0x32 0x30	ES 0520	State event: "READY"

When a CanSlave_IO CU is opened the HotStart-events will always pop up indicating that the CanSlave is now ready to communicate (if the CanSlave is connected for the first time this will last several seconds!). Every instruction sent before the upcome of the HotStart-events will go lost.
The events following the HotStart-events are CanSlave dependent, e.g. not every CanSlave will bring a state event.
Other message types like Monitor- or List-data are not implemented yet.

4. LICOP Communication Protocol

The communication protocol described in this chapter is used between the controller (ChemStation or handheld control module) and any instrument build up of modules from the Agilent 1200 family. The name of the protocol is LICOP which stands for: Link Independent COmmunication Protocol. Independent in this context means that it does not depend on the hardware interface used. Presently it is supported for LAN (TCP/IP), IEEE 488 (HP-IB), RS-232C and CAN communication and in future may be extended to other serial or parallel interfaces.

First version of LICOP was:	A.01.00 (released Oct. 1995; FW rev. A.01.04)
Next version of LICOP was:	A.02.00 (released June 2000; FW rev. A.04.01)
Next version of LICOP was:	A.06.00 (released June 2005; FW rev. A.06.01) resp. A.03.00 (released June 2005; FW rev. B.01.01)
Current version of LICOP is:	B.01.00 (released July 2007; FW rev. A.06.10 resp. B.06.10)

Terms used in this chapter:

Controller: ChemStation (G2170) on PC or hand held control module (G1323A)

Instrument: Set of modules from the Agilent 1200 family which are connected with CAN bus system. e.g.: G1310A---G1313A---G1315A (Simple HP-LC system)

Module: Component of the Agilent 1200 family (e.g.: G1315A - LC Diode Array Detector)

IEEE 488: HP-IB (High Performance Interface Bus). Widely used bus system for measurement systems. The underlying IEEE 488 protocol is not described in this document.

RS-232C: serial point-to-point connection.
Please refer to see chapter 5.1. RS-232C Interface on page 67 for a description of RS-232C settings.

CAN: Controller Area Network. Bus system for module communication. Physical visible is a 8 wire cable with Jack Plugs (Western Connector). The underlying CAN protocol is not described in this document.

CU: A Communication Unit is a virtual communication channel. CU characteristics are:
- Name e.g.: IN (IN stands for INstruction/reply)
- Module to which it belongs to (e.g.: G1315A)
- Direction of communication (in, out or both)
- Buffer size (e.g.: 2048 bytes out, 1024 bytes in)
- Number of buffers (e.g.: 1)
- Type of information they are going to transmit or accept (e.g.: instructions/replies)

Socket: Independant connections to different modules and/or multiple CUs through a single physical link require an abstract level of virtual channels with uniqe identifiers. These identifiers are so-called socket numbers generated during communication startup.

Trigger: License to send a message to a socket. The trigger permits the instrument or the controller to send a message to a socket. A socket without a trigger is not proposed to receive a message.

4.1. General Overview

The modules of the Agilent 1200 instrument family need a structured communication path to handle all the different types of information flowing between the controller and the instrument. For each module there are several CUs. The CUs have dedicated data types they are transmitting/receiving to/from the controller. Because of only one physical link between controller and instrument as well the numerous hardware configurations of the Agilent 1200 instruments the exchange of messages must be supported by virtual channels. The virtual channels are installed between controller and modules as sockets on demand.

Example
The modules, pump (G1310A), injector (G1313A) and diode array detector (G1315A) are connected via CAN. All modules in this example have CUs for instructions/replies, events and monitor data (IN, EV, MO). The CUs have associated sockets (0x3D0F-0x3D17). Communication to the controller (ChemStation on computer) is done via IEEE 488 (HP-IB). Inside the controller the structure of the instrument is mirrored, i.e. sockets are associated to the modules and the CUs.
To setup the socket connection and keep the communication going, there are 4 socket that are not connected to any module's CU. They are named "Control sockets". The module specific sockets are named "Data sockets".

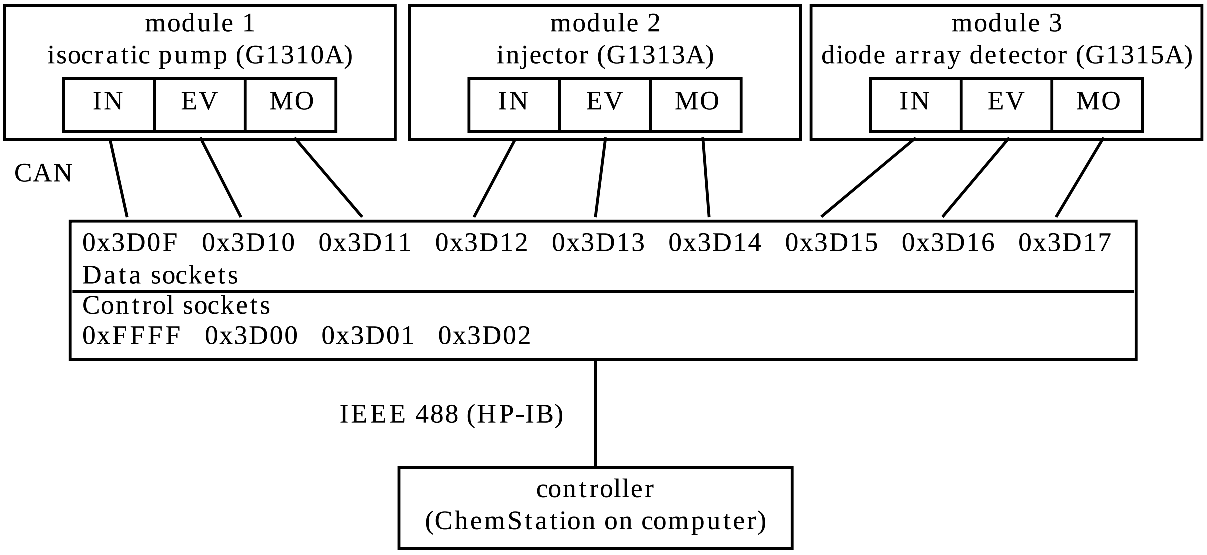


Figure 1: instrument's CU and socket structure

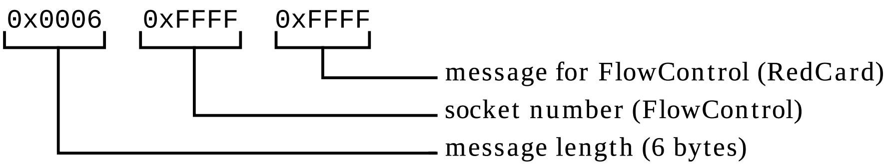
4.2. Message exchange

Every message has 3 parts: message length, socket number and data. Messages have following structure: LLSS<data>

LL is message length, SS is socket number, <data> is message for associated CU

Message length (LL) and socket number (SS) are 16 bit words. The first bit of message length is reserved for future use. The length of the entire message (including message length and socket number) is determined by the message length (LL).

Examples of messages
RedCard



Triggers

0x000C 0xFFFF 0x3D00 0x05 0x3D15 0x01

trigger for socket 0x3D15: 1
socket 0x3D15
trigger for ConfigSocket: 5
ConfigSocket
socket number (FlowControl)
message length (12 bytes)

command for OpenSocket (controller -> instrument): list all open sockets

0x0005 0x3D02 0x0B

message for OpenSocket: OPEN_SOCKETS
socket number (OpenSocket)
message length (5 bytes)

reply from OpenSocket (instrument -> controller): list of all open sockets

0x0011 0x0302 0x0B 0x3D00 0x3D01 0x3D02 0x3D15 0x3D16 0x3D17

open sockets: 0x3D00, 0x3D01, 0x3D02, 0x3D15, 0x3D16, 0x3D1
message from OpenSocket: OPEN_SOCKETS
socket number (OpenSocket)
message length (17 bytes)

command for an Instruction/Reply socket

0x0008 0x3D15 0x69 0x64 0x6E 0x3F

message for IN-CU: "idn?"
socket number (0x3D15)
message length (8 bytes)

NOTE: the messages do not contain any blanks between message length, socket number and ..., spaces are for readability only!

Example:
Assuming the controller wants to know whether the lamp is switched on in the diode array detector. The appropriate command would be: "ACT:LAMP?". A possible answer could be "RA 0000 ACT:LAMP 1" which is in words: "Request Accepted, no error, Lamp is on". The request has to be send to the instruction CU of G1315A. The following information is exchanged on the communication line to the instrument:

from controller to instrument

0x000D 0x3D15 0x41 0x43 0x54 0x3A 0x4C 0x41 0x4D 0x50 0x3F

LL = 13; SS = 0x3D15; message = "ACT:LAMP?"

from controller to instrument

0x0007 0xFFFF 0x3D15 0x01

trigger permits instrument to send the answer from instrument to controller

```
0x0016  0x3D15 0x52 0x41 0x20 0x30 0x30 0x30 0x30 0x20 0x41 0x43 0x54 0x3A
                                               0x4C 0x41 0x4D 0x50 0x20 0x31
```

LL = 22; SS = 0x3D15; message = "RA 0000 ACT:LAMP 1"

from instrument to controller

```
0x0007  0xFFFF  0x3D15 0x01
```

trigger permits controller to send the next instruction

4.3. Sockets

There are 2 types of sockets: control and data sockets.
The control sockets are: FlowControl, ConfigSocket, EventSocket and OpenSocket.

4.3.1. FlowControl

This channel is used to control the status of the connection as well as the flow of messages between controller and instrument. This channel is bi-directional and it is always available. Requests on this channel are neither acknowledged nor responded.

Socket number

FlowControl gets a unique socket number: 0xFFFF

Status

On instrument side there are 2 states defined for the connection controller to instrument:

OUT_OF_SYNC: the controller and the instrument do not have an established communication. In this state the instrument handles every message as a byte stream in which it tries to find a "RedCard"-message. Until a "RedCard"-message was found the instrument stays in this state. After detection of a "RedCard"-message the instrument replies with it's own "RedCard"-message. The controller and the instrument are synchronous.

IN_SYNC: all control sockets are available, controller and instrument can use the synchronized communication line. "Heartbeat"-messages are exchanged.

Messages

FlowControl recognizes three types of messages:

"RedCard"-message: a specific pattern (0xFFFF) which is used to synchronize the communication line. The initial "RedCard"-message is send from the controller to the instrument. The instrument replies with a "RedCard"-message which has 3 socket numbers appended. They define the ConfigSocket, EventSocket and OpenSocket.

"Trigger"-messages: through FlowControl both, controller and instrument, tell each other how many messages will be accepted for a socket. It is a protocol violation if one of the participants sends more messages as the other is willing to receive. "Trigger"- messages hold socket number and corresponding number of triggers. A "Trigger"-message may have triggers for more than one socket (message data: socket# triggers socket# triggers ...).

"Heartbeat"-messages: If no message is send within a time out period, FlowControl has to send a "Heartbeat"-message to inform the counterpart that it is still alive. A "Heartbeat"-message is a trigger message for the ConfigSocket which contains a trigger value of "0".

```
0x0007  0xFFFF  0x0300 0x00
```

Heartbeat: trigger "0" for ConfigSocket

The Heartbeat must be provided from both sides. Each participant must know whether its partner is active or not. If no HeartBeat is transmitted within the time out period the connection returns to OUT_OF_SYNC-state.

Transition changes between synchronous and asynchronous state of the controller and the instrument are given in the figures below:

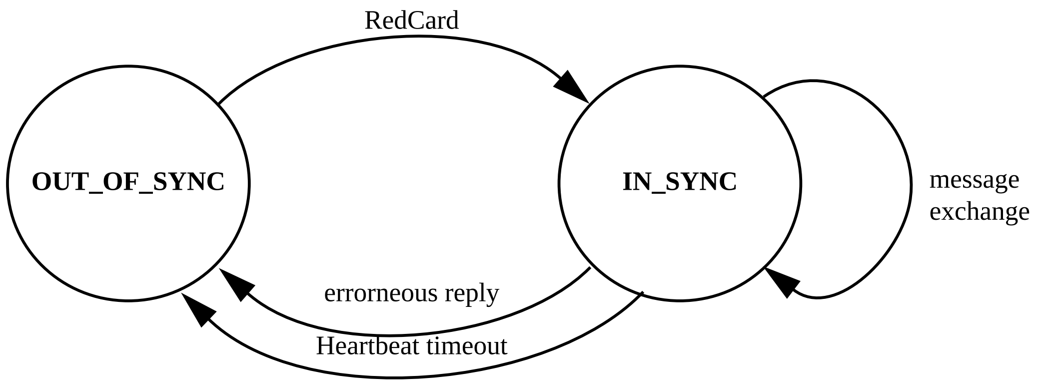
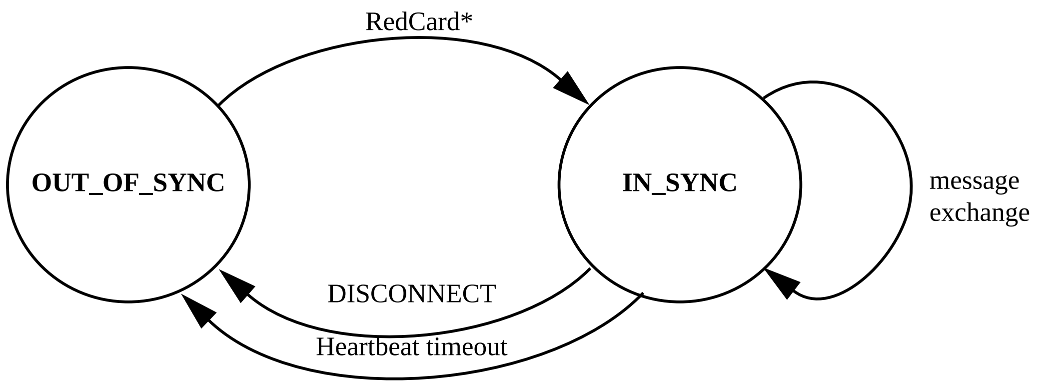


Figure 2: status diagram of the controller



* send back RedCard, activate Config-, Event- and OpenSocket, start Heartbeat

Figure 3: status diagram of the instrument

The behavior of FlowControl at the controller and the instrument is nearly identical. Both send and receive trigger messages to/from their counterpart. However, only the controller is able to initate a new connection.

4.3.2. ConfigSocket, EventSocket, OpenSocket

The control sockets are implicitely opened with "RedCard"-reply. ConfigSocket and OpenSocket get 1 implicit trigger, i.e. they will listen to the next message and do not need an additional trigger. For further messages new triggers are required. ConfigSocket and OpenSocket work in duplex mode, they accept requests from the controller and send replies. The transmission mode of EventSocket is simplex, event messages only flow from the instrument to the controller.
All control sockets have a buffer size of 128 bytes, longer messages are truncated. Starting with LICOP version B.01.00 the OUT-buffer size of all control sockets is increased to 2048 bytes.
The messages to/from control sockets have the following format:
msg-data = code[P1|P1PV]
PV = [P1|PV]
P1 = parameter (string params will be followed by '\0')
code = depends on the specific control socket

Valid commands (`cmd_key`) for ConfigSocket and OpenSocket are listed in the following tables:

commands	code
FIRST_MODULE_DESC	0x01

commands	code
NEXT_MODULE_DESC	0x02
MODULE_IDENTIFICATION	0x03
FIRST_CU_DESC	0x04
NEXT_CU_DESC	0x05
SPECIFIC_CU_DESC	0x06
HEARTBEAT	0x10
CONTROLLER	0x08
VERSION	0x11
FIRST_SLAVE_DESC	0x12
NEXT_SLAVE_DESC	0x13

Valid commands for ConfigSocket

commands	code
OPEN	0x09
CLOSE	0x0A
DISCONNECT	0x07
OPEN_SOCKETS	0x0B
OPEN_SOCKETS_EX	0x17

Valid commands for OpenSocket

An erroneous message to either ConfigSocket or OpenSocket will be answered by an error-return message (ERROR_RTN) from the specified socket. Any other unacceptable message (wrong length, wrong socket#) will give an event-return message (EVENT_RTN) from EventSocket. In both cases ERROR_RTN respectively EVENT_RTN follows the socket number. The original message which caused the error is appended to the returned message. ERROR_RTN and EVENT_RTN are coded as follows:

ERROR_RTN	0x0E
EVENT_RTN	0x0F

The errors which will be returned by either ConfigSocket or OpenSocket are:

ERROR	0x0001	unspecified error: should never occur!
UNKNOWN_COMMAND	0x0002	unknown command
WRONG_FORMAT	0x0003	wrong message format
BUFFER_TOO_SHORT	0x0004	buffer is too short to carry the reply
UNKNOWN_CU	0x0005	unknown CU
LAST_CU	0x0006	last CU, no next CU
NO_CU_REGISTERED	0x0007	no CUs registered at this module
LAST_MODULE	0x0008	last module, no next module
UNKNOWN_MODULE	0x0009	unknown module
NO_NEW_SOCKET	0x000A	could not open new socket
COULD_NOT_CONNECT_CU	0x000B	could not connect CU to socket
WRONG_SOCKET	0x000C	wrong socket #
SOCKET_FAILED	0x000D	message couldn't delivered
COULD_NOT_CONNECT_SLAVE	0x000E	connection to a slave failed
WRONG_MODULE_TYPE	0x000F	wrong module type: could not execute cmd

Errors returned by ConfigSocket and OpenSocket

parameter	range	type
<messagelength>	0x0001..0x7FFF	signed int (first bit reserved for future use)
<command>	0x01...	unsigned char
<socket#>	0x0000..0xFFFC	unsigned int
<ModType>		string (maxlen == 16, including '\0')
<SerialNo>		string (maxlen == 16, including '\0')
<CU>		string (maxlen == 16, including '\0')
<trigger>	0x00..0xFE	unsigned char
<buffers>	0x00..0xFE	unsigned char
<buffersize>	0x0000..0x7FFF	unsigned int
<timeout>	0x0000..0xFFFF	unsigned int

parameter	range	type
<errorcode>	0x0001..0xFFFF	unsigned int
<eventcode>	0x0001..0xFFFF	unsigned int

Type of used parameters and ranges

All parameters of type signed/unsigned int are transferred with HIGH-byte first: 2 bytes: HI-LO, 4 bytes: HI1-LO1-HI2-LO2 (MOTOROLA format).

ConfigSocket

ConfigSocket is used to inform the controller about the configuration of an instrument.

FIRST_MODULE_DESC

desc: returns the description of the first module
code: 0x01
syn: FIRST_MODULE_DESC
rep: FIRST_MODULE_DESC<ModType><SerNo>
err: ERROR_RTN<errorcode>FIRST_MODULE_DESC
errors: WRONG_FORMAT the message has a wrong format

NEXT_MODULE_DESC

desc: returns the description of the next module
code: 0x02
syn: NEXT_MODULE_DESC<ModType1><SerNo1>
rep: NEXT_MODULE_DESC<ModType2><SerNo2>
err: ERROR_RTN<errorcode>NEXT_MODULE_DESC<ModType1><SerNo1>
errors: WRONG_FORMAT the message has a wrong format
UNKNOWN_MODULE the specified module is unknown
LAST_MODULE last module is specified, there exists no NEXT module

The FIRST_MODULE_DESC and NEXT_MODULE_DESC commands will be used to get an overview which modules are accessible through the present connection. From this set of modules an individual instrument can be set up.

MODULE_IDENTIFICATION

desc: the specified module will identify itself (blinking LED)
code: 0x03
syn: MODULE_IDENTIFICATION<ModType><SerNo>
rep: MODULE_IDENTIFICATION<ModType><SerNo>
err: ERROR_RTN<errorcode>MODULE_IDENTIFICATION<ModType><SerNo>
errors: WRONG_FORMAT the message has a wrong format
UNKNOWN_MODULE the specified module is unknown

FIRST_CU_DESC

desc: returns description of the first CU of the specified module
code: 0x04
syn: FIRST_CU_DESC<ModType><SerNo>
rep: FIRST_CU_DESC<ModType><SerNo><CU>
<OutBuffers><OutBufferSize><InBuffers><InBufferSize>
err: ERROR_RTN<errorcode>FIRST_CU_DESC<ModType><SerNo>
errors: WRONG_FORMAT the message has a wrong format
UNKNOWN_MODULE the specified module is unknown
NO_CU_REGISTERED at this module no CU is registered

NEXT_CU_DESC

desc: returns the description of the next CU of the specified module
code: 0x05
syn: NEXT_CU_DESC<ModType><SerNo><CU1>
rep: NEXT_CU_DESC<ModType><SerNo><CU2>
<OutBuffers><OutBufferSize><InBuffers><InBufferSize>
err: ERROR_RTN<errorcode>NEXT_CU_DESC<ModType><SerNo><CU1>
errors: WRONG_FORMAT the message has a wrong format

UNKNOWN_MODULE the specified module is unknown
UNKNOWN_CU the specified CU is unknown
LAST_CU last CU is specified, there exist no NEXT_CU

SPECIFIC_CU_DESC

desc: returns the description of the specific CU of the specified module
code: 0x06
syn: SPECIFIC_CU_DESC<ModType><SerNo><CU>
rep: SPECIFIC_CU_DESC<ModType><SerNo><CU>
<OutBuffers><OutBufferSize><InBuffers><InBufferSize>
err: ERROR_RTN<errorcode>SPECIFIC_CU_DESC<ModType><SerNo><CU>
errors: WRONG_FORMAT the message has a wrong format
UNKNOWN_MODULE the specified module is unknown
UNKNOWN_CU the specified CU is unknown

The reply for FIRST_CU_DESC, NEXT_CU_DESC and SPECIFIC_CU_DESC contains number and size of out/in-buffers as a proposal for the CU. Real values for number and size of the buffers are determined by OPEN command. The controller can request lower values but may not increase values.

HEARTBEAT

desc: set new timeout value for heartbeat and/or returns current value
code: 0x10
syn: HEARTBEAT[<timeout>]
rep: HEARTBEAT<timeout>
err: ERROR_RTN<errorcode>HEARTBEAT
errors: WRONG_FORMAT the message has a wrong format

<timeout> is measured in seconds, default is 600 secs. If a controller does not wish to use HEARTBEAT, it must set the time out value to '0', otherwise the connection will be terminated after the default time out! With a timeout of 0 the instrument will still send HEARTBEATs every 2 sec if there hasn't been any other communication. However, the controller doesn't need to answer HEARTBEATs to keep the protocol in synchronous state.

CONTROLLER

desc: returns info about connected controllers, CPU load, free memory
code: 0x08
syn: CONTROLLER
rep: CONTROLLER<# controllers><CPU load><free mem (in KB)>
err: ERROR_RTN<errorcode>CONTROLLER

VERSION

desc: returns version info of LICOP
code: 0x11
syn: VERSION
rep: VERSION<version string>
err: ERROR_RTN<errorcode>VERSION
<version string> is zero-terminated, current version is: "LICOP A.01.00"

FIRST_SLAVE_DESC

desc: returns the description of the first CAN-slave
code: 0x12
syn: FIRST_SLAVE_DESC
rep: FIRST_SLAVE_DESC<ModType><SerNo>
err: ERROR_RTN<errorcode>FIRST_SLAVE_DESC
errors: WRONG_FORMAT the message has a wrong format

NEXT_SLAVE_DESC

desc: returns the description of the next CAN-slave
code: 0x13
syn: NEXT_SLAVE_DESC<ModType1><SerNo1>

rep: NEXT_SLAVE_DESC<ModType2><SerNo2>
err: ERROR_RTN<errorcode>NEXT_SLAVE_DESC<ModType1><SerNo1>
errors: WRONG_FORMAT the message has a wrong format
UNKNOWN_MODULE the specified CAN-slave is unknown
LAST_MODULE last CAN-slave is specified, there exists no NEXT CAN-slave

The FIRST_SLAVE_DESC and NEXT_SLAVE_DESC commands will be used to get an overview which CAN-slaves are accessible through the present connection.

OpenSocket

OpenSocket is used to open and close data sockets.

OPEN

desc: opens a socket to the specified CU
code: 0x09
syn: OPEN<ModType><SerNo><CU><OutBuffers><OutBufferSize>
<InBuffers><InBufferSize>
rep: OPEN<ModType><SerNo><CU><OutBuffers><OutBufferSize>
<InBuffers><InBufferSize><socket#>
err: ERROR_RTN<errorcode>OPEN<ModType><SerNo><CU>
errors: WRONG_FORMAT the message has a wrong format
UNKNOWN_MODULE the specified module is unknown
UNKNOWN_CU the specified CU does not exist at this module
NO_CU_REGISTERED at this module NO CU is registered
COULD_NOT_CONNECT_CU
could not connect the specified CU to a socket, lack of resources
NO_NEW_SOCKET could not open new socket

Number and size of out/in-buffers in OPEN requests are proposed values. The replied values for number and size of out/in-buffers are less or equal to the requested values and determines the number of buffers and buffersize in use. If (InBuffers > 0), there is automatically one trigger set for this new socket, no further triggers for this socket are sent until this trigger is used. For further messages new triggers are required.

CLOSE

desc: closes a socket
code: 0x0A
syn: CLOSE<socket1><socket2>..<socketN>
rep: CLOSE<socket1><socket2>..<socketN>
err: ERROR_RTN<errorcode>CLOSE<socket1><socket2>..<socketN>
errors: WRONG_FORMAT the message has a wrong format
WRONG_SOCKET wrong socket#

OPEN, CLOSE are used to install and remove data sockets. The features of each socket depends on the module and connected CU.

DISCONNECT

desc: disconnects the controller (includes the closing of all open sockets!)
code: 0x07
syn: DISCONNECT
rep:
err:

DISCONNECT removes the connection between the controller and the instrument.

OPEN_SOCKETS

desc: returns a list of all open sockets of the controller
code: 0x0B
syn: OPEN_SOCKETS
rep: OPEN_SOCKETS<socket1><socket2>...<socketN>
err: ERROR_RTN<errorcode>OPEN_SOCKETS

errors: `WRONG_FORMAT` the message has a wrong format

The list of open sockets includes the control sockets and all data sockets. Maximum number for <socketN> is 63, thus means that the length of the reply does not exceed 128 bytes.
ATTENTION: no error is generated in case this listing is truncated!

OPEN_SOCKETS_EX

desc: returns a list of all open sockets of the controller
code: `0x17`
syn: `OPEN_SOCKETS_EX`
rep: `OPEN_SOCKETS_EX<socket1><socket2>...<socketM>`
err: `ERROR_RTN<errorcode>OPEN_SOCKETS_EX`
errors: `WRONG_FORMAT` the message has a wrong format

Starting with LICOP version B.01.00 the OUT-buffer size of all control sockets is increased to 2048 bytes. This change was neccessary for situations where the controller has more than 63 open sockets. Now the maximum number of open sockets <socketM> is limited to 1023 (3 control sockets and up to 1020 data sockets).

EventSocket
EventSocket is used to inform the controller of events concerning the communication line and the instrument configuration.

EVENT_RTN

desc: an event happened in the instrument
code: `0x0F`
syn:
rep: `EVENT_RTN<eventcode>[<input>]`
err:

EVENT_RTNs are generated in case of changes of the configuration of the instrument (e.g. modules turned off or on) as well as a message from the controller arrives which could not be delivered (wrong socket#, no free buffer).

EVENT	0x0001	unspecified event, should never occure
EVENT_OVERFLOW	0x0002	EventSocket has no more free buffers
WRONG_SOCKET **	0x0003	wrong socket#
NO_BUFFERS **	0x0004	socket has no free buffers
CONFIG_CHANGE	0x0005	configuration change: a module was turned off or on
SOCKET_FAILED **	0x0006	message couldn't delivered

** these events append the original message which caused the event
Events returned by EventSocket

4.3.3. Data sockets

These sockets are used for data exchange between controller and instrument. A socket is connected to a CU of a module through an OPEN command. The reply from the OPEN command holds the new assigned socket number. The behavior of a data socket depends on the module and CU associated. See modules' description for details of available CUs.

4.4. Examples

The examples in this chapter will show the steps which are necessary until a controller can send an instruction to the instrument and will receive a reply. Following examples show specifics on the different hardware interfaces.

4.4.1. Get a Module's Identification

Legend:
< Denotes messages to instrument
> Denotes messages from instrument
{} Comment

< 0x0006 0xFFFF 0xFF 0xFF {RedCard send}
> 0x000C 0xFFFF 0xFF 0xFF 0x3D00 0x3D01 0x3D02
{RedCard answered, socket numbers for ConfigSocket:
0x3D00, EventSocket: 0x3D01, OpenSocket: 0x3D02}
< 0x0007 0xFFFF 0x3D01 0x01 {Trigger EventSocket, enable error messages}
< 0x0005 0x3D00 0x01 {First Module Description Request}
< 0x0007 0xFFFF 0x3D00 0x01 {Trigger ConfigSocket, enable socket to answer}
> 0x0007 0xFFFF 0x3D00 0x00 {HeartBeat}
< 0x0007 0xFFFF 0x3D00 0x00 {HeartBeat answered by controller}
> 0x0007 0xFFFF 0x3D00 0x01 {Trigger ConfigSocket}
> 0x0017 0x3D00 0x47 0x31 0x33 0x31 0x35 0x41 0x00 0x44 0x45 0x30 0x30 0x30 0x30 0x30 0x30 0x30 0x30 0x00
{First Module Description received:
G1315A DE00000000}
< 0x0020 0x3D02 0x09 0x47 0x31 0x33 0x31 0x35 0x41 0x00 0x44 0x45 0x30 0x30 0x30 0x30 0x30 0x30 0x30 0x30 0x00 0x49 0x4E 0x00 0x01 0x0800 0x01 0x0400
{Open request to G1315A DE00000000 for
INstruction CU with 1 output buffer of size 2048 and
1 input buffer of size 1024.}
< 0x0007 0xFFFF 0x3D02 0x01 {Trigger OpenSocket for reply}
> 0x0007 0xFFFF 0x3D02 0x01 {Trigger OpenSocket, enable socket to answer}
> 0x0022 0x3D02 0x09 0x47 0x31 0x31 0x30 0x33 0x41 0x00 0x44 0x45 0x30 0x30 0x30 0x30 0x30 0x30 0x30 0x30 0x00 0x49 0x4E 0x00 0x01 0x0800 0x01 0x0400 0x3D18
{Answer from OpenSocket:
G1315A DE00000000 IN with
1 output buffer of size 2048 and 1 input buffer of size 1024
is now open with socket number 0x3D18.}
< 0x0008 0x3D17 0x49 0x44 0x4E 0x3F {To socket 0x3D17 (IN-CU): IDN?}
< 0x0007 0xFFFF 0x3D17 0x01 {Trigger INstruction CU, enable socket to answer}
> 0x0007 0xFFFF 0x3D17 0x01 {Trigger INstruction CU, enable to send next request}
> 0x003B 0x3D17 0x52 0x41 0x20 0x30 0x30 0x30 0x30 0x20 0x49 0x44 0x4E 0x20 0x22 0x48 0x45 0x57 0x4C 0x45 0x54 0x54 0x2D 0x50 0x41 0x43 0x4B 0x41 0x52 0x44 0x2C 0x47 0x31 0x31 0x30 0x33 0x41 0x2C 0x44 0x45 0x30 0x30 0x30 0x30 0x30 0x30 0x30 0x30 0x2C 0x41 0x2E 0x30 0x31 0x2E 0x30 0x33 0x22
{Reply from IDN? instruction:
RA 0000 IDN "HEWLETT-PACKARD,G1103A,DE00000000,A.01.03"}

Remark: The example is reduced to necessary messages to send the instruction "IDN?" to the Agilent 1200 DAD (G1315A). Repeated "Next Module Description" and "First/Next CU Description" will give the full information about all available modules and their CUs.

4.4.2. HP-IB communication

This is an example on HP-IB communication between controller and instrument:
The Controller starts with a DCL (Device CLear) on the HP-IB (step 0). Then the instrument at HP-IB address 26 is read. The instrument is present but in asynchronous mode. The instrument sends

a '?' as reply to indicate 'Nothing to offer'.
The controller sends a 'RedCard' to the instrument beginning with step 7 of the example. In this example the controller waits for 2 seconds before it reads from the instrument again (Starting with step 16 of the example). The instrument replies with a 'RedCard' and appends a HeartBeat as there was no communication within the last 2 seconds.
The controller replies the HeartBeat beginning with step 38. Finally the controller disconnects from the instrument starting with step 48 of our example.
To get a message from the instrument the controller must poll the HP-IB because the instrument don't use SRQ (Service ReQuest). In those cases that the instrument has no message for the controller, the instrument generates a 'dummy'-message ('?') to serve the controller's read request. In our example this 'dummy'-message is shown from step 1 to step 6.

```
00000   *   0x14! DCL (Device CLear)
00001   *   0x3F! UNL (UNListen)
00002   *   0x35! MLA (My Listen Address)
00003   *   0x5A! TALK 26
00004   >   0x00
00005   >   0x03
00006   >   0x3F! (Instrument has nothing to offer) + EOI
00007   *   0x3F! UNL (UNListen)
00008   *   0x55! MTA (My Talk Address)
00009   *   0x3A! LISTEN 26
00010   <   0x00
00011   <   0x06
00012   <   0xFF
00013   <   0xFF
00014   <   0xFF
00015   <   0xFF! (RedCard sent to instrument) + EOI
            ! Time delay 2 seconds
00016   *   0x3F! UNL (UNListen)
00017   *   0x35! MLA (My Listen Address)
00018   *   0x5A! TALK 26
00019   >   0x00
00020   >   0x0C
00021   >   0xFF
00022   >   0xFF
00023   >   0xFF
00024   >   0xFF
00025   >   0x3D
00026   >   0x00
00027   >   0x3D
00028   >   0x01
00029   >   0x3D
00030   >   0x02! (RedCard from instrument)
00031   >   0x00
00032   >   0x07
00033   >   0xFF
00034   >   0xFF
00035   >   0x3D
00036   >   0x00
00037   >   0x00! (HeartBeat from instrument) + EOI
00038   *   0x3F! UNL (UNListen)
00039   *   0x55! MTA (My Talk Address)
00040   *   0x3A! LISTEN 26
00041   <   0x00
00042   <   0x07
```

00043	<	0xFF
00044	<	0xFF
00045	<	0x3D
00046	<	0x00
00047	<	0x00! (HeartBeat to instrument) + EOI
00048	*	0x3F! UNL (UNListen)
00049	*	0x55! MTA (My Talk Address)
00050	*	0x3A! LISTEN 26
00051	<	0x00
00052	<	0x05
00053	<	0x3D
00054	<	0x02
00055	<	0x07! (Disconnect protocol) + EOI

Legend:	1st col:	line number
	2nd col:	*HP-IB command (control byte)
		<From controller to instrument (data byte)
		>From instrument to controller (data byte)
	3rd col:	byte on HP-IB
	4th col:	comment

4.4.3. RS-232C communication

Because LICOP is a binary transmission protocol it is not possible to use the XON/XOFF mechanism. If the controller uses XON/XOFF extra bytes will be inserted into the byte stream transferred from controller to the module. On module side these bytes would be wrongly interpreted which will lead to a reset of the communication line and loss of data.
To control the data exchange on RS-232C LICOP uses the hardware handshake lines DSR, DTR, CTS and RTS. The module sets DTR, it checks DSR for an active controller at the other end of the line. As long as the module has set RTS it is able to receive data, the module will stop sending data if it detects a low CTS.

5. Interfaces

5.1. RS-232C Interface

All modules of the Agilent 1200 series have a RS-232C connector which is a 9 pin D-sub-min connector (male). The modules are desigend as DCE (Data Communication Equipment).
It is recommended that the default settings for the RS-232C interface is used (Baud Rate: 19200, Data Bits: 8, Parity: None). Baudrate, Data Bits and Parity are selectable using the control switches at rear side of the module.

Baud rate:
Selectable baud rates (38400, 19200, 9600, 4800, 2400 and 1200). Default is 19200 baud.

Functions (Data bits, Parity):
7 Bits + Even Parity + 1 Stop Bits
7 Bits + Odd Parity + 1 Stop Bits
8 Bits + None Parity + 1 Stop Bits (default)

Break:
for HP internal use only!

Flowcontrol:
XON/XOFF: the modules do not react to XON-XOFF.
Handshake lines:RTS, CTS, DTR, DSR and DCD.

RS-232C connector:
9 pin D-sub-min connector (male):
protective ground:shield
DCD (in):pin 1
RxD (in):pin 2
TxD (out):pin 3
DTR (out):pin 4
signal ground:pin 5
DSR (in):pin 6
RTS (out):pin 7
CTS (in):pin 8
RI (in):pin 9 (not used)

To connect a module of the Agilent 1200 series to a computer use the cable:
null modem cable, 9-pin F to 9-pin F (Agilent Technologies Part Number: F1047-80002)
Agilent Technologies Order Number: Agilent Technologies 34398A RS-232 Cable Kit

5.2. MIO Interface

MIO (Modular I/O) is an HP internal extension interface. Connector and software protocol are standardized to enable the easy use of optional communication hardware, such as RS232, HPIB or LAN. (For information about the installation of a MIO card, please refer to the chapter "Interface Board" in your Reference Manual.)

Since RS232 and HPIB are available onboard, the most interresting use of the MIO interface is to setup a LAN connection. The appropriate MIO card available for this purpose is the HP JetDirect Print Server card, widely used wherever printers shall have connection to a LAN.

The HP JetDirect Print Server card is available in different versions:

J2550A	HP JetDirect Print Server Card Ports: RJ-45 (10BaseT)
J2552B	HP JetDirect Print Server Card Ports: RJ-45 (10BaseT), BNC (10Base2), LocalTalk System7

For more information visit: http://www.hp.com/net_printing/jetdirect/miocards.html

Make sure that the card's on-board firmware is not older than Rev.A.05.05. This can be done by dumping the cards I/O description with the instructions "MIO:FIRS?" and "MIO:NEXT?". The description will appear in the following manner:

```
HP JETDIRECT          J2552B
FIRMWARE REVISION:    A.05.05
LAN HW ADDRESS:   080009E5EC70
PORT SELECT:          BNC
MFG ID:      3611000703010315
=============================
I/O CARD READY
...
```

To setup a TCP/IP connection to the card only a few card parameters have to be configured. The most important parameter is the IP address. Depending on your networking environment, there are several ways to set it. You may use

the BOOTP/TFTP protocol
the Telnet command
the JetDirect card's configuration menu

(See also the corresponding chapters in the card's manual.)

Besides the IP address you need to configure two additional parameters, important for the communication protocol LICOP. These parameters infect the card's behaviour on buffer-packing and push-bit handling.
Unfortunately, these two parameters are not configurable from the card's configuration menu. And also, they are not stored in the card's Non-Volatile-RAM, i.e. they are lost when the module is power-cycled. Performing a cold start on your module (e.g. after firmware download or by the corresponding DIP switch setting) leads to a loss of all parameters stored in the JetDirect card's NVRAM. The card is reset to factory defaults. In this case the card is awaiting configuration by BOOTP/TFTP.

5.2.1. MIO Card Configuration

Using the BOOTP/TFTP protocol

BOOTP servers are typically found in UNIX environments, but are also available for PC networks also. When using this facility, configuration parameters are loaded on the card every time your Agilent 1200 series module is turned on.
A typical entry in the bootptab file may be:

```
wadpc070:\
   :ht=ether:\
   :ha=080009E5EC70:\
   :hn:\
   :ip=15.136.250.0:\
   :sm=255.0.0.0: \
   :T146=01: \
   :T147=02: \
   :vm=rfc1048:
```

where the entries `:T146=01:` and `:T147=02:` are the two additinal parameters mentioned above. (For details contact your sytem administrator.)

Using the Telnet command

The JetDirect card may also be configured using the Telnet command. By typing

telnet IP address

the configuration session is started. To set the IP address (when Telnet was called with the default address) type

ip: 15.136.250.0

To set the two special parameters mentioned above, type

buffer-packing: off
write-mode: eoi_push

Typing a

/

lists the current settings like

```
===JetDirect Telnet Configuration===
    Present Config  : FRONT PANEL/TELNET
    MAC Address     : 08:00:09:e5:ec:70

    IP Address      : 15.136.250.0
    Subnet Mask     : 255.0.0.0
    Default Gateway : 15.136.250.0
    Syslog Server   : 0.0.0.0
    Idle Timeout    : 90 Seconds
    Set Cmnty Name  : Not Specified
    Host Name       : Not Specified

    DHCP Config     : Disabled
    Passwd          : Disabled
    Other Protocols : Enabled
    Banner page     : Enabled
```

Using the configuration menue

Similar to a control panel initialization on a printer, you can use the JetDirect card's configuration menue to set the networking parameters from the module side. Use the instructions "MIO:CONF" and "MIO:MENU" to perform this kind of configuration process.

(MIO card I/O description and configuration menu will also be available on the Agilent 1200 series hand-held control module, G1323A, Rev. A.02.xx., or newer)

5.2.2. Connecting to your module

To run LICOP on a TCP/IP connection to your Agilent 1200 series module, create a stream-based socket and connect it to Port 9100 of the JetDirect card.

Mind that the JetDirect card monitors a so-called End-of-Job timeout (apart from the TCP Connectionn timeout). This parameter can be set through the configuration menue only. When the connection becomes idle the card starts counting and switches to an other protocol when the timeout elapses. To prevent the card from doing so, the connection should be used periodically. Usually this is done anyway, since LICOP owns a so-called Heart-Beat mechanism.

5.2.3. MIO Card Instructions

Name

`MIO:CONF`

change configuration mode

Synopsis

Instruction:	`MIO:CONF <mode>`
Reply:	`<rc> MIO:CONF <mode>`

Description

Enter/leave MIO card's configuration mode. To be called, if configuration will be changed with configuration menue.

Parameters

<mode>	= 1: close active connection and begin configuration
	= 0: end configuration and store current settings into NVRAM

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0805` MIO card failed executing command

Examples

Instruction:	`MIO:CONF 1`
Reply:	`RA 0000 MIO:CONF 1`

Name

MIO:MENU
browse configuration menu

Synopsis

Instruction: `MIO:MENU <cmd>`
Reply: `<rc> MIO:MENU <cmd>,<"parm">,<"val">,<flag>`

Description

Browse the entries of the MIO card's configuration menue, change parameter values and set them active.

Parameters

<cmd>	= 0: return first parameter with current set value
	= 1: return next parameter
	= 2: return previous parameter
	= 3: return next value for the selcted parameter
	= 4: return previous value
	= 5: make selected value the new set value of the selected parameter
<"parm">	parameter string
<"val">	value string
<flag>	= 1: returned value is the current set value
	= 0: returned value is not the current set value

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0300 end of list
RE 0805 MIO card failed executing command

Examples

Instruction:	MIO:MENU 0
Reply:	RA 0300 MIO:MENU 0,"CFG NETWORK","NO",1
Instruction:	MIO:MENU 3
Reply:	RA 0300 MIO:MENU 3,"CFG NETWORK","YES",0
Instruction:	MIO:MENU 5
Reply:	RA 0000 MIO:MENU 5,"CFG NETWORK","YES",1
Instruction:	MIO:MENU 1
Reply:	RA 0000 MIO:MENU 1,"NETWORK","AUTO",1

Name

MIO:FIRS?
get first I/O description string
MIO:NEXT?
get next I/O description string

Synopsis

Instruction:	`MIO:FIRS?`
Reply:	`<rc> MIO:FIRS <idx>,<"desc">`
Instruction:	`MIO:NEXT? <idx>`
Reply:	`<rc> MIO:NEXT <idx>,<"desc">`

Description

List MIO card's I/O description. To get the next string, the index of the previous string has to be enterred.

Parameters

<idx>	index of returned description string
<"desc">	description string

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0300 end of list
RE 0805 MIO card failed executing command

Examples

Instruction:	`MIO:FIRS?`
Reply:	`RA 0000 MIO:FIRS 0,"HP JETDIRECT          J2552B"`
Instruction:	`MIO:NEXT? 0`
Reply:	`RA 0000 MIO:NEXT 1,"FIRMWARE REVISION    A.05.05"`
Instruction:	`MIO:NEXT? 1`
Reply:	`RA 0000 MIO:NEXT 2,"LAN HW ADDRESS: 080009E5EC70"`

Name

MIO:STAT?
inquire MIO card status

Synopsis

Instruction:	`MIO:STAT?`
Reply:	`<rc> MIO:STAT <data>,<err>,<link>,<mode>,<->, <stat>,<conn>`

Description

Inquire MIO card's I/O status.

Parameters

<data>	data status
<err>	error status
<link>	link status
<mode>	card mode
<->	-
<stat>	card status
<conn>	connection status

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0805 MIO card failed executing command

Examples

Instruction:	`MIO:STAT?`
Reply:	`RA 0000 MIO:STAT 0,0,0,0,0,0,2`

6. DIP Switches

The modules are equipped with an 8-bit DIP switch which is used to set communication characteristics, test & boot mode selection and modules specific initialization settings.
A specific Mode Select setting is taken over at startup time and is then stored in non-volatile RAM. At the next powercycle, an other mode may be selected, while the previous one is retained. After a coldstart all values are reset to their defaults. Test & boot settings will not be retained over power-cycles.

Mode Select	1	2	3	4	5	6	7	8
HPIB	0	0	0	HPIB Address				
RS-232	0	1	Baudrate			DataBits	Parity	
RSVD	1	0	Module Specific					
TEST/ BOOT	1	1		System Selection		Enable Monitor	Inhibit Watchdog	Force Coldstart

System Selection (SS): 00: Loaded, 01: Resident, 10: Boot, 11: Reserved.
Enable Monitor (EM): Enable Serial Debug Monitor (19200, 8, N, 1)
Inhibit Watchdog (IW): Processor Watchdog disabled.
Force Coldstart (FC): Start Cold.

Mode Select			3	4	5	6	7	8
HPIB	0	0	0	HPIB Address				
RS-232	0	1	Baudrate			Bits	Parity	
RES	1	0	Module Specific					
Test	1	1		SS		EM	IW	FC

Baudrate							
0	0	0	9600	1	0	0	9600
0	0	1	1200	1	0	1	14400
0	1	0	2400	1	1	0	19200
0	1	1	4800	1	1	1	38400

Bits	
0	7
1	8

Parity		
0	0	N
0	1	O
1	0	E

Test			
SS	System Select	IW	Inhibit Watchdog
EM	Enable Monitor	FC	Force Coldstart

Setting the DIP switch to all zeroes causes all values to be left untouched.
Setting the DIP switch to all ones forces defaults.

7. COSY (COntrol and SYnchronization)

7.1. Preface

An Agilent 1200 Series HPLC system is usually a set of several modules which are connected through the CAN-Network. This connection provides high flexibility for the system design including free programmable exchange of control and synchronization data among modules.

7.2. Introduction / Overview

COSY is a general mechanism implemented to realize module internal and module overlapping system control and synchronization. The main components of COSY are lists and an interpreter to process the lists. A COSY-List is a sequence of COSY-Elements each of which is a functional unit. COSY-Elements can be classified into three categories:

- Synchronization elements
- Command elements
- Instruction elements

7.2.1. Synchronization elements

Synchronization elements are used to synchronize COSY-Lists running in parallel within the same module and to synchronize with COSY-Lists running in other modules. They are also used to synchronize COSY-Lists with module specific functions.
A Sync is a synchronization event that can be sent, received or cleared.
The keywords used for this type of elements are: SEND, SYNC, INC, DEC, CLR and EVNT.

7.2.2. Command elements

Command elements are control directives for the COSY-Interpreter. They are used to control the flow of execution and to deal with COSY-Lists. The keywords used for command elements are: COND, NCND, WAIT, REM, DCPO, TRC, STOP, REPT, ERPT, RUN, DEL and NEW.

7.2.3. Instruction elements

Instruction elements are used to call instructions as they are defined in the communication interface of the module.
When COSY receives an instruction element it calls the parser to get the instruction parsed. The result is an internal representation of the instruction (totab) which is stored in the COSY-List. There are two ways how an instruction gets executed at run time. One way is to send mail with the mail code received from instruction parsing. The other way is to call the corresponding function directly. Which way actually is chosen at run time depends on how the instruction is defined in the parser table (Leaf and FLeaf).
An Instruction element given as keyword ISEQ causes the COSY-Interpreter to wait for completion of the instruction before proceeding with the next element in the COSY-List. Keyword IPAR allows the COSY-Interpreter to work in parallel which means, it does not wait for the instruction to be completed but immediately proceeds with the next COSY-Element.

7.3. Theory of Operation

COSY is designed as a core firmware package for Agilent 1200 Series modules. It is a general mechanism to process lists of sequential actions. The lists can be edited on a controller and down-loaded to a module via instructions, or they are build in as default COSY-Lists. This functionality makes it possible to setup or customize an HPLC system of several modules by just creating appropriate COSY-Lists.
A module can hold several COSY-Lists. Once a COSY-List is down-loaded it can be started by instruction COSY:STRT <ListNum>. Up to 32 COSY-Lists can run in parallel within one module. A COSY-List is finished by either reaching the list end or by instruction COSY:STOP <ListNum>.

7.3.1. Syncs

Syncs are used to synchronize a COSY-List with module specific actions and to synchronize between different COSY-Lists. Sending a Sync means to broadcast it in the system. That means, every COSY-List in all modules gets the Sync. Even if a COSY-List was not running when the Sync was sent, it inherits the Sync as soon as it gets started. A Sync can be consumed e.g. by COSY:SYNC, so that it's no longer present for this cosylist. A sync can be cleared by COSY:CLR so that the Sync is no longer present for all COSY-Lists in the system.
The Sync's number decides whether the sync is propagated to all other modules in the system or not. More in detail

- Sync ∈ [1,69999]: The Sync is propagated to all modules in the system.
- Sync ∈ [70000,79999]: This range is protected for system-generated syncs, e.g. appended to a COSY:ISEQ-cmd. Use is not allowed to manipulate these Syncs.
- Sync ∈[80000,89999]: The Sync is available for all COSY-lists within thet module where it was sent. The other modules within the system do not participate in that Sync.
- Sync >=90000: Protected, do not use.

7.3.2. Counters

Cosy provides different counter concepts:

- Manual counter operation
- Startup-Counter
- Preset-Counter

Also the counter id's reflective two different behaviors:

- Counter Id ∈ [1,39999]: Analysis counters are deleted at end of the analysis and at Abort.
- Counter Id ∈ [40001,49999]: These counters are still present after the analysis. They are also not affected by an abort. Such a counter is used within a COSY-list that runs independently from the analysis state. Resetting these counters can only be done by COSY:RSET or RSET.

7.3.2.1. Manual counter operation

In the traditional concept counters can be incremented and decremented and they produce a Sync if the counter's value is decremented to zero.
The sync is automatically cleared by COSY:INC-command if the counter doesn't exist.

Example:

Module #1	Module #2	Decscription
COSY:NEW 1	COSY:NEW 1	create cosylist
COSY:INC 1,100,42	COSY:INC 1,100,42	

Module #1	Module #2	Decscription
COSY:WAIT 1,1.0	COSY:WAIT 1,0.01	symbolizes that module #1 has something to do and module #2 does not.
COSY:DEC 1,100,42	COSY:DEC 1,100,42	Module is finished with action
COSY:SYNC 1,42	COSY:SYNC 1,42	Wait until all actions are finished.
COSY:EVNT 1,42	COSY:EVNT 1,42	Send Event to EV-CU: EV 0300, <SecondsSince1970>, 42

If an unknown number of COSY-lists (e.g. one list within each module) shall execute one or more action phases in common then the counter is incremented in all COSY-lists prior to the action and decremented when the action is finished. After decrementing the COSY-list is waiting for the sync number 42.
If the last COSY-list finishes the action phase it decrements the counter to zero. All modules recognize counter is zero and send the sync. Hence all modules continue with the COSY-list in common.

Be aware that the sync is not broadcasted over CAN. So it's not possible to simply wait for the sync with COSY:SYNC 1,42 without counter manipulation.

7.3.2.2. Preset-Counters

Synchronization by using the Manual counter operation may fail if an action phase within a COSY-list has duration zero. In that case the behavior depends on the execution time. If this COSY-list executes INC and DEC prior to one of the other lists executing INC then the counter is set to one and back to zero and the sync is sent too early because the other COSY-lists didn't even start with the action phase.
This lack is really a problem if one wants to execute an action phase synchronously more than once. Therefore the preset counters are introduced: With COSY:INCP resp. COSY:DECP one can manipulate the preset value of a counter. An action phase is finished with COSY:DECS. The counter is incremented. If the counter is zero a module private sync is sent and it's value automatically reset to the preset value.

Do not use preset-counters with COSY:DEC because the sync coming with COSY:DEC will never be sent but the module private generated sync.

Example:

Module #1	Module #2	Decscription
COSY:NEW 1	COSY:NEW 1	create cosylist
COSY:INP 1,100	COSY:INCP 1,100	
COSY:SYNP 1	COSY:SYNP 1	Wait until all COSY-lists are started. (see Startup-Counter on page 78)
COSY:REPT 1,100	COSY:REPT 1,100	Repeat the action phase 100-times
COSY:WAIT 1,1.0		symbolizes that module #1 has something to do and module #2 hasn't.

Module #1	Module #2	Decscription
COSY:DECS 1,100	COSY:DECS 1,100	Module is finished with action.
COSY:EVNT 1,42	COSY:EVNT 1,42	Send Event to EV-CU: EV 0300, <SecondsSince1970>, 42
COSY:ERPT 1	COSY:ERPT 1	Begin next loop.

Note that the COSY-list also indicates "I want to participate in an action phase". But it has to do this only once. So one can have loops with always the same counter.

7.3.2.3. Startup-Counter

If an analysis starts all main COSY-lists start logically at the same time. If one wants to use counters for synchronizing between the COSY-lists it has to be ensured that all COSY-lists are started and have placed the increment -counter command for the first phase.

Therefore there is the special startup command COSY:SYNP. It's implementing a counter with a given preset value even at begin of an analysis. This counter is always present and set to the number of all analysis participating modules (or to one if a start-single-analysis with STRS is performed). This number comprises all LC-modules except for resident modules, CANslaves and local control module. Executing the command comprises decrementing the counter and waiting for counter value being zero. It can also be used within loops.
The counter's ID is is private to the system.

7.3.3. Instructions

To process instruction elements of type ISEQ the COSY-Interpreter adds a Sync to the mail which is used to send the internal representation of the instruction (totab) to the task that has to execute the instruction. If no Sync was given for the instruction (see COSY:ISEQ on page 113) COSY itself adds a Sync to it. This sync has to be sent by the application when the instruction is finished. If the application doesn't take over the Sync to send it after the instruction is finished, the Sync comes back to COSY with the instruction-mail. In this case COSY by itself sends the Sync assuming the instruction is finished and proceeds with next element in the COSY-List.
The sync for IPAR if given is sent by the application when the instruction is finished. This Sync can be used to control the flow of COSY-Lists. The COSY-Interpreter doesn't obey this Sync but immediately proceeds with next element in the COSY-List.

7.3.4. Create and edit a COSY-List

COSY:NEW 5,1creates COSY-List named "5" residing in normal RAM
COSY:ISEQ 5,"tray:move 2"enter an instruction of type ISEQ into COSY-List "5"
Note: there is no Sync given for the instruction, so at execution time COSY adds an internal Sync to the instruction and waits for the end of the instruction

7.4. Edit COSY - Lists

7.4.1. Synchronization Elements

Name

COSY:SYNC
Wait for a Sync.

Synopsis

Instruction:	`COSY:SYNC <list>,<synclist>`
Reply:	`<rc> COSY:SYNC <list>,<synclist>`
Instruction:	`COSY:SYNC <list>,<synclist>,<clr>`
Reply:	`<rc> COSY:SYNC <list>,<synclist>,<clr>`

Description

This instruction adds an element to the given COSY-List that waits for one of the specified synchronization events. The syncs contained in <synclist> are linked with an "or" operator.
<clr> not given, or given as 1 means: clear the sync after waiting for it;
<clr> given as 0 means: do not clear the sync after waiting

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<synclist>	a string representing a list of syncs (e.g. "12" or "555,46,101"); for sync range see chapter 7.3.1. Syncs on page 76.
<clr>	an optional flag that indicates if the sync is to clear after occurrence (default is to clear the sync) Note: if this flag is 1 or not given, the sync gets cleared only for the COSY-List that was waiting for the sync - all other COSY-Lists still have the sync!

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0901 no such COSY-List

Examples

(a)	Comment:	enter a sync element into COSY-List 5
	Instruction:	`COSY:SYNC 5,"666"`
	Reply:	`RA 0000 COSY:SYNC 5,"666"`
(b)	Comment:	try to enter a sync element into a not existing COSY-List
	Instruction:	`COSY:SYNC 23,"666"`
	Reply:	`RE 0901 COSY:SYNC 23,"666"`

Name

COSY:SEND
Send a Sync.

Synopsis

Instruction: COSY:SEND <list>,<sync>
Reply: <rc> COSY:SEND <list>,<sync>

Description

This instruction adds an element to the specified COSY-List that broadcasts the sync given by parameter <sync>.

Parameter

<list>	the COSY-List to edit; <list> $\in$[-32768, 32767]
<sync>	the sync to broadcast; for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0901 no such COSY-List

Examples

(a) Comment: enter a "send sync 33" element into COSY-List 5
Instruction: COSY:SEND 5,33
Reply: RA 0000 COSY:SEND 5,33

Name

`COSY:EVNT`

Send Sync to controller.

Synopsis

Instruction: `COSY:EVNT <list>,<event>`

Reply: `<rc> COSY:EVNT <list>,<event>`

Description

This instruction adds an element to the specified COSY-List that sends synchronization event <event> to a controller (WS, daVinci).

Parameter

<list>	the COSY-List to edit; <list> $\in$[-32768, 32767]
<event>	synchronization event to controller; <event> $\in$[0, 2147483647]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

COSY:CLR
Clear a Sync.

Synopsis

Instruction:	`COSY:CLR <list>`	Note: Available with revisions A.03.00
Reply:	`<rc> COSY:CLR <list>`	
Instruction:	`COSY:CLR <list>,<sync>`	
Reply:	`<rc> COSY:CLR <list>,<sync>`	

Description

This instruction adds an element to the specified COSY-List that broadcasts the clearing of the sync given by parameter <sync>. Broadcasting means, the sync is cleared in all modules of the CAN cluster. If parameter <sync> is not given all syncs in the module are cleared (no broadcasting).

Parameter

<list>	the COSY-List to edit; <list> $\in$[-32768, 32767]
<sync>	the sync to clear; for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	let list 5 clear all syncs in the module
	Instruction:	`COSY:CLR 5`
	Reply:	`RA 0000 COSY:CLR 5`
(b)	Comment:	let list 5 clear sync 99 globally in a CAN cluster
	Instruction:	`COSY:CLR 5,99`
	Reply:	`COSY:CLR 5,99`

Name

COSY:INC
Increment counter.

Synopsis

Instruction: `COSY:INC <list>,<CounterId>,<Sync>`
Reply: `<rc> COSY:INC <list>,<CounterId>,<Sync>`

Description

This command element broadcasts to increment the given counter according to the concept of Manual counter operation. If the counter is new, i.e. it has actual counter value zero, the sync is cleared. The sync associated is sent when after decrementing the counter value is 0. For decrementing use COSY:DEC.
Associated with this counter-ID do not use COSY:INCP, COSY:DECP, COSY:DECS.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<CounterId>	CounterId ∈[1, 49999], see also Counters on page 76
<Sync>	for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: RA 0000

Name

`COSY:DEC`

Decrement counter.

Synopsis

Instruction: `COSY:DEC <list>,<CounterId>,<Sync>`

Reply: `<rc> COSY:DEC <list>,<CounterId>,<Sync>`

Description

This command element broadcasts to decrement the given counter according to the concept of Manual counter operation. The sync associated is sent when after decrementing the counter value is 0. See COSY:INC on page 83. The sync is NOT propagated to the other modules regardless the number of the sync!
Associated with this counter-ID do not use COSY:INCP, COSY:DECP, COSY:DECS.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<CounterId>	CounterId ∈[1, 49999], see also Counters on page 76
<Sync>	For sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

COSY:INCP

Increment preset value.

Synopsis

Instruction: `COSY:INCP <list>,<CounterId>`

Reply: `<rc> COSY:INCP <list>,<CounterId>`

Description

This command element broadcasts to increment the preset value of a counter. See COSY:DECP. For counter usage see Preset-Counters on page 77.

Associated with this counter-ID do not use COSY:INC, COSY:DEC.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<CounterId>	CounterId ∈[1, 49999], see also Counters on page 76

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

`COSY:DECP`

Decrement preset value.

Synopsis

Instruction: `COSY:DECP <list>,<CounterId>`

Reply: `<rc> COSY:DECP <list>,<CounterId>`

Description

This command element broadcasts to decrement the preset value of a counter. See COSY:INCP. For counter usage see Preset-Counters on page 77.
Associated with this counter-ID do not use COSY:INC, COSY:DEC.

Parameter

<list>	the COSY-List to edit; <list> ∈ [-32768, 32767]
<CounterId>	CounterId ∈ [1, 49999], see also Counters on page 76

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: `RA 0000`

Name

COSY:DECS

Decrements a counter and waits for "counter = 0".

Synopsis

Instruction: `COSY:DECS <list>,<CounterId>`

Reply: `<rc> COSY:DECS <list>,<CounterId>`

Description

This command element broadcasts to decrement the value of a counter. If the new value is zero, the command is finished. In that case furthermore the actual value is reset to the preset value if there is one (see COSY:INCP and COSY:DECP). If counter value is non-zero the execution waits until counter is zero.

Associated with this counter-ID do not use COSY:INC, COSY:DEC.

Parameter

<list>	the COSY-List to edit; <list> ∈ [-32768, 32767]
<CounterId>	CounterId ∈ [1, 49999], see also Counters on page 76

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

COSY:SYNP

Synchronize all analysis participating modules.

Synopsis

Instruction: `COSY:SYNP <list>`

Reply: `<rc> COSY:SYNP <list>`

Description

This command element holds all modules that are participating in an analysis until they have also reached COSY:SYNP but not later than a timeout of two seconds. The command is one of the first commands of all default main COSY-lists as the startup of the different mainlists can slightly differ between the different modules. Hence the normal solution with incrementing and decrementing counters doesn't work at begin of the COSY-lists.
The number of modules participating in an analysis comprises modules registered to LICOP that are able to participate in the analysis, i.e. all LC- modules except for resident modules, CANslaves and local control module.

Parameter

<list> the COSY-List to edit; <list> $\in$ [-32768, 32767]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

7.4.2. Command Elements

Name

COSY:COND
Execution on condition.

Synopsis

Instruction:	`COSY:COND <list>,<synclist>`
Reply:	`<rc> COSY:COND <list>,<synclist>`

Description

Adds an element to the COSY-List that controls execution as follows:
- execute next element in the COSY-List only if at least one of the specified syncs is set
- otherwise skip next element. Note: This element does not clear any sync!.

Parameter

<list>	the COSY-List to edit; <list> $\in$ [-32768, 32767]
<synclist>	a string representing a list of syncs (e.g. "12" or "555,46,101"); for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: RA 0000

Name

COSY:NCND

Execution if not condition.

Synopsis

Instruction: `COSY:NCND <list>,<synclist>`

Reply: `<rc> COSY:NCND <list>,<synclist>`

Description

Adds an element to the COSY-List that controls execution as follows:

- execute next element in the COSY-List only if none of the specified syncs is set
- otherwise skip next element. Note: This element does not clear any sync!.

Parameter

<list>	the COSY-List to edit; <list> ∈ [-32768, 32767]
<synclist>	a string representing a list of syncs (e.g. "12" or "555,46,101"); for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: RA 0000

Name

COSY:WAIT
Wait time.

Synopsis

Instruction: `COSY:WAIT <list>,<time>`
Reply: `<rc> COSY:WAIT <list>,<time>`

Description

This command element lets the COSY-List wait until the given time is elapsed.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<time>	wait time in minutes; <time> ∈[0.0, 10000.0] NOTE!: For G3010A wait time in seconds.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: `RA 0000`

Name

COSY:TIME

Preset Time Counter of List.

Synopsis

Instruction: `COSY:TIME <list>,<time>`

Reply: `<rc> COSY:TIME <list>,<time>`

Description

This command sets the time counter of the running list to a new value.

Parameter

<list>	the COSY-List to edit; <list> $\in$[-32768, 32767]
<time>	wait time in minutes; <time> $\in$[0.0, 10000.0] for ALP-Systems in seconds

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: Reset the time of a list 18 to 0

Instruction: `COSY:TIME 18,0`

Reply: `RA 0000 COSY:TIME 18,0`

Name

COSY:REM
Comment in a COSY-List.

Synopsis

Instruction:	`COSY:REM <list>,<comment>`
Reply:	`<rc> COSY:REM <list>,<comment>`

Description

This element is just a comment in the COSY-List. It has no effect at execution time.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<comment>	comment string up to 80 characters

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: `RA 0000`

Name

`COSY:DCPO`

Restart a COSY-List.

Synopsis

Instruction:	`COSY:DCPO <list>`
Reply:	`<rc> COSY:DCPO <list>`
Instruction:	`COSY:DCPO <list>,<sync>`
Reply:	`<rc> COSY:DCPO <list>,<sync>`

Description

This command element restarts execution of the COSY-List until the specified sync occurs. This is an endless loop if no sync is given as parameter or if the given sync never occurs. To end an endless COSY-List see COSY:STOP on page 96.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<sync>	optional sync, for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

COSY:TRC

Traces in COSY-Lists.

Synopsis

Instruction: `COSY:TRC <list>,<string>`

Reply: `<rc> COSY:TRC <list>,<string>`

Description

This element creates trace information output on a terminal.

Parameter

<list>	the COSY-List to edit; <list> $\in$[-32768, 32767]
<string>	trace info string up to 80 characters

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

COSY:STOP

Stop a COSY-List with another COSY-List.

Synopsis

Instruction: `COSY:STOP <list>,<ListToStop>`

Reply: `<rc> COSY:STOP <list>,<ListToStop>`

Description

This command element allows a COSY-List to terminate another COSY-List.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<ListToStop>	the COSY-List to stop; <ListToStop> ∈[-32768, 32767]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

Note: Available with revisions A.03.00

`COSY:REPT`

Begin of a repeat loop.

Synopsis

Instruction: `COSY:REPT <list>,<repetition>`

Reply: `<rc> COSY:REPT <list>,<repetition>`

Description

This element defines the begin of a loop. For the end of a loop see COSY:ERPT on page 98.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<repetition>	number of repeated loop cycles [0, 2147483647]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

Note: Available with revisions A.03.00

COSY:ERPT
End of a repeat loop.

Synopsis

Instruction: COSY:ERPT <list>
Reply: <rc> COSY:ERPT <list>

Description

This element defines the end of a loop. See also COSY:REPT on page 97.

Parameter

<list> the COSY-List to edit; <list> ∈[-32768, 32767]

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: RA 0000

Name

Note: Available with revisions A.03.00

COSY:RUN

COSY-List runs other COSY-Lists.

Synopsis

Instruction:	`COSY:RUN <list>,<ListToRun>`
Reply:	`<rc> COSY:RUN <list>,<ListToRun>`
Instruction:	`COSY:RUN <list>,<ListToRun>,<Sync>`
Reply:	`<rc> COSY:RUN <list>,<ListToRun>,<Sync>`

Description

This element allows a COSY-List to run (fork) other COSY-Lists. The sync if given is sent after the forked COSY-List finished.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<ListToRun>	the COSY-List to run; <ListToRun> ∈[-32768, 32767]
<Sync>	for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

Note: Available with revisions A.03.00

COSY:DEL
Delete COSY-Lists with a COSY-List.

Synopsis

Instruction: `COSY:DEL <list>,<ListToDelete>`
Reply: `<rc> COSY:DEL <list>,<ListToDelete>`

Description

This element allows a COSY-List to delete another COSY-List.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<ListToDelete>	the COSY-List to delete; <ListToDelete> ∈[-32768, 32767]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: `RA 0000`

Name

Note: Available with revisions A.03.00

`COSY:NEW`

Create COSY-Lists with a COSY-List.

Synopsis

Instruction:	`COSY:NEW <list>,<ListToCreate>,<MemType>`
Reply:	`<rc> COSY:NEW <list>,<ListToCreate>,<MemType>`

Description

This element enables a COSY-List to create a new COSY-List.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<ListToCreate>	the new COSY-List to create; <ListToCreate> ∈[-32768, 32767]
<MemType>	type of memory the new COSY-List resides: 1 = MEM_RAM, 2 = MEM_RAMCONT

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

Note: Available with revisions A.03.00

`COSY:CSTA`

Start a clock in a COSY-List.

Synopsis

Instruction:	`COSY:CSTA <List>,<ClockID>`
Reply:	`<rc> COSY:CSTA <List>,<ClockID>`
Instruction:	`COSY:CSTA <List>,<ClockID>,<StartValue>`
Reply:	`<rc> COSY:CSTA <List>,<ClockID>,<StartValue>`

Description

This command starts a clock in a COSY-List. The clock is referenced by parameter ClockID and starts with the time StartValue. The clock runs at a rate of 10 Hz i.e., beginning with StartValue the clock is incremented every 100 ms. The default start value is 0.

Parameter

<List>	COSY-List to edit; <List> ∈[-32768, 32767]
<ClockID>	unique clock identifier ∈[0, 32767]
<StartValue>	the clock is preset by this value ∈[0.0, 100000.0] seconds

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: edit COSY-List 2 to start clock 55 with start value 0

Instruction: `COSY:CSTA 2,55,0`

Reply: `RA 0000 COSY:CSTA 2,55,0`

Name

Note: Available with revisions A.03.00

`COSY:CSTO`

Stop a clock in a COSY-List.

Synopsis

Instruction: `COSY:CSTO <List>,<ClockID>`

Reply: `<rc> COSY:CSTO <List>,<ClockID>`

Description

This command stops a clock in a COSY-List. It has no effect if the clock referenced by parameter ClockID was never started. See COSY:CSTA on page 102. All actions pending related to this clock are discarded.

Parameter

<List>	COSY-List to edit; <List> ∈[-32768, 32767]
<ClockID>	unique clock identifier ∈[0, 32767]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: edit COSY-List 2 to stop clock 55

Instruction: `COSY:CSTO 2,55`

Reply: `RA 0000 COSY:CSTO 2,55`

Name

Note: Available with revisions A.03.00

`COSY:WCLK`

Wait until clock reaches specified time.

Synopsis

Instruction:	`COSY:WCLK <List>,<ClockID>,<Time>,<Abs>`
Reply:	`<rc> COSY:WCLK <List>,<ClockID>,<Time>,<Abs>`
Instruction:	`COSY:WCLK <List>,<ClockID>,<Time>,<Abs>,<Sync>`
Reply:	`<rc> COSY:WCLK <List>,<ClockID>,<Time>,<Abs>,<Sync>`

Description

This command delays a COSY-List until the clock reaches the time given. If the clock is not running (see COSY:CSTA on page 102) the command returns immediately. The command never returns if the clock gets stopped while the command is pending. If an additional sync is given, this sync is broadcasted at command return.

Parameter

<List>	COSY-List to edit; <List> ∈ [-32768, 32767]
<ClockID>	unique clock identifier ∈ [0, 32767]
<Time>	time in seconds (with 1 decimal place)
<Abs>	specifies the meaning of parameter <Time> Abs=1 means <Time> is an absolute time Abs=0 means <Time> is an offset to the current time of the clock
<Sync>	sync to broadcast (optional), for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	let COSY-List 2 wait until clock 55 reaches the time of 24.5 seconds
	Instruction:	`COSY:WCLK 2,55,24.5,1`
	Reply:	`RA 0000 COSY:WCLK 2,55,24.5,1`
(b)	Comment:	let COSY-List 2 wait until clock 55 reaches the time "now" + 12 sec
	Instruction:	`COSY:WCLK 2,55,12.0,0`
	Reply:	`RA 0000 COSY:WCLK 2,55,12.0,0`

Name

Note: Available with revisions A.03.00

`COSY:AT`

Add a sync to a clock.

Synopsis

Instruction: `COSY:AT <List>,<ClockID>,<Time>,<Abs>,<Sync>`

Reply: `<rc> COSY:AT <List>,<ClockID>,<Time>,<Abs>,<Sync>`

Description

This command adds a sync to the clock specified by ClockID. The sync is broadcasted in the system when the clock reaches the time given by parameter Time. If the clock gets stopped all syncs are discarded (see COSY:CSTO on page 103).

Parameter

<List>	COSY-List to edit; <list> $\in$[-32768, 32767]
<ClockID>	unique clock identifier $\in$[0, 32767]
<Time>	time in seconds (with 1 decimal place)
<Abs>	specifies the meaning of parameter <Time> Abs=1 means <Time> is an absolute time Abs=0 means <Time> is an offset to the current time of the clock
<Sync>	sync to broadcast, for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: COSY-List 2 lets clock 55 broadcast sync 900 at the time 60.0 seconds
Instruction: `COSY:AT 2,55,60.0,1,900`
Reply: `RA 0000 COSY:AT 2,55,60.0,1,900`

(b) Comment: COSY-List 2 lets clock 55 broadcast sync 900 at the time "now" + 12 sec
Instruction: `COSY:WCLK 2,55,12.0,0,900`
Reply: `RA 0000 COSY:WCLK 2,55,12.0,0,900`

Name

Note: Available with revisions A.03.00

COSY:PER

Add a periodic sync to a clock.

Synopsis

Instruction: `COSY:PER <List>,<ClockID>,<Period>,<Sync>`

Reply: `<rc> COSY:PER <List>,<ClockID>,<Period>,<Sync>`

Description

This command adds a periodic sync to the clock specified by ClockID. The sync is broadcasted in the system in constant intervals until it gets stopped. A periodic sync can be stopped by stopping the corresponding clock or by calling this instruction again with parameter Period set to -Period.

Parameter

<List>	COSY-List to edit; <list> $\in$[-32768, 32767]
<ClockID>	unique clock identifier $\in$[0, 32767]
<Period>	time in seconds (with 1 decimal place)
<Sync>	sync to broadcast, for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: COSY-List 2 initiates clock 55 to broadcast sync 900 every 2.5 seconds

Instruction: `COSY:PER 2,55,2.5,900`

Reply: `RA 0000 COSY:PER 2,55,2.5,900`

(b) Comment: COSY-List 2 stops the periodic broadcasting of sync 900

Instruction: `COSY:PER 2,55,-2.5,900`

Reply: `RA 0000 COSY:PER 2,55,-2.5,900`

Name

Note: Available with revisions A.03.00

COSY:IF

Conditional branches in COSY-Lists.

Synopsis

Instruction:	`COSY:IF <List>,<Sync>`
Reply:	`<rc> COSY:IF <List>,<Sync>`

Description

This command allows the conditional execution of a block of instructions in a COSY-List. The condition is true if the specified sync is present. The block includes all COSY-List elements until ELSE or FI.

Parameter

<List>	COSY-List to edit; <List> ∈[-32768, 32767]
<Sync>	sync to be evaluated for the condition, for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

COSY:IFN
Conditional branches in COSY-Lists.

Synopsis

Instruction: `COSY:IFN <List>,<Sync>`
Reply: `<rc> COSY:IFN <List>,<Sync>`

Description

This command allows the conditional execution of a block of instructions in a COSY-List. The condition is true if the specified sync is **NOT** present. The block includes all COSY-List elements until ELSE or FI.

Parameter

<List>	COSY-List to edit; <List> $\in$[-32768, 32767]
<Sync>	sync to be evaluated for the condition, for sync range see chapter 7.3.1. Syncs on page 76.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: `RA 0000`

Name

`COSY:ELSE`

Conditional branches in COSY-Lists.

Synopsis

Instruction:	`COSY:ELSE <List>`
Reply:	`<rc> COSY:ELSE <List>`

Description

This COSY-List element is the beginning of the alternative block if the condition part of an IF statement is false.

Parameter

<List>	COSY-List to edit; <List> ∈ [-32768, 32767]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

`COSY:FI`

End of a conditional branche.

Synopsis

Instruction: `COSY:FI <List>`

Reply: `<rc> COSY:FI <List>`

Description

This COSY-List element is the end of an IF block or, if there was an ELSE statement, the end of the alternative block.

Parameter

<List> COSY-List to edit; <List> $\in$[-32768, 32767]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

COSY:WRUN
Wait until begin of run

Synopsis

Instruction: `COSY:WRUN <List>`
Reply: `<rc> COSY:WRUN <List>`

Description

This COSY-List element holds the list execution until begin of run.

Parameter

<List> COSY-List to edit; <List> ∈ [-32768, 32767]

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: RA 0000

Name

COSY:WERU

Wait until end of run.

Synopsis

Instruction: `COSY:WERU <List>`

Reply: `<rc> COSY:WERU <List>`

Description

This COSY-List element holds the list execution until end of run.

Parameter

<List> COSY-List to edit; <List> $\in$[-32768, 32767]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

7.4.3. Instruction Elements

Name

`COSY:ISEQ`
Execute instruction and wait for completion.

Synopsis

Instruction:	`COSY:ISEQ <list>,<Inst>`
Reply:	`<rc> COSY:ISEQ <list>,<Inst>`
Instruction:	`COSY:ISEQ <list>,<Inst>,<Sync>`
Reply:	`<rc> COSY:ISEQ <list>,<Inst>,<Sync>`
Instruction:	`COSY:ISEQ <list>,<Time>,<Inst>`
Reply:	`<rc> COSY:ISEQ <list>,<Time>,<Inst>`
Instruction:	`COSY:ISEQ <list>,<Time>,<Inst>,<Sync>`
Reply:	`<rc> COSY:ISEQ <list>,<Time>,<Inst>,<Sync>`

Description

This instruction element handles the given instruction as follows:
1. send an appropriate TotabMail including all parameter necessary
2. wait for completion of the instruction
Waiting for completion of the instruction is independent of the optional parameter <Sync>.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<Inst>	any module specific instruction
<Sync>	a sync indicating completion of the instruction; for sync range see chapter 7.3.1. Syncs on page 76.
<Time>	do not execute the instruction until the given time is elapsed counting from the start of the COSY-List; <Time> in minutes for Agilent 1200 series modules or in seconds for G3010A, range is [0.0, 10000.0]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: `RA 0000`

Name

COSY:IPAR

Execute instruction witout waiting for completion.

Synopsis

Instruction:	COSY:IPAR <list>,<Inst>
Reply:	<rc> COSY:IPAR <list>,<Inst>
Instruction:	COSY:IPAR <list>,<Inst>,<Sync>
Reply:	<rc> COSY:IPAR <list>,<Inst>,<Sync>
Instruction:	COSY:IPAR <list>,<Time>,<Inst>
Reply:	<rc> COSY:IPAR <list>,<Time>,<Inst>
Instruction:	COSY:IPAR <list>,<Time>,<Inst>,<Sync>
Reply:	<rc> COSY:IPAR <list>,<Time>,<Inst>,<Sync>

Description

This element handles the given instruction by sending an appropriate TotabMail including all parameter necessary but does not wait for completion of the instruction.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<Inst>	any module specific instruction
<Sync>	a sync indicating completion of the instruction; for sync range see chapter 7.3.1. Syncs on page 76.
<Time>	do not execute the instruction until the given time is elapsed counting from the start of the COSY-List; <Time> in minutes for Agilent 1200 series modules, in seconds for G3010A, range is [0.0, 10000.0]

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: RA 0000

7.5. COSY - Instructions

7.5.1. Control Instructions

Name

COSY:DEL
Delete COSY-Lists.

Synopsis

Instruction:	`COSY:DEL`
Reply:	`<rc> COSY:DEL`
Instruction:	`COSY:DEL <list>`
Reply:	`<rc> COSY:DEL <list>`

Description

This instruction deletes the named (user defined) COSY-List. If no list is specified all user defined COSY-Lists are deleted.

Parameter

<list> the COSY-List to delete; <list> $\in$[-32768, 32767]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0901` no such COSY-List

Examples

(a)	Comment:	delete COSY-List 5
	Instruction:	`COSY:DEL 5`
	Reply:	`RA 0000 COSY:DEL 5`
(b)	Comment:	try to delete a not existing COSY-List
	Instruction:	`COSY:DEL 23`
	Reply:	`RE 0901 COSY:DEL 23`

Name

COSY:NEW

Create a new COSY-List.

Synopsis

Instruction:	`COSY:NEW <list>`
Reply:	`<rc> COSY:NEW <list>`
Instruction:	`COSY:NEW <list>,<MemType>`
Reply:	`<rc> COSY:NEW <list>,<MemType>`

Description

This instruction creates a new COSY-List.

Parameter

<list>	the COSY-List to edit; <list> ∈[-32768, 32767]
<MemType>	type of memory the new COSY-List resides: 1 = MEM_RAM, 2 = MEM_RAMCONT

Default Values

MEM_RAM

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

COSY:STOP
Stop a running COSY-List.

Synopsis

Instruction:	`COSY:STOP <list>`
Reply:	`<rc> COSY:STOP <list>`

Description

This instruction terminates a running COSY-List.

Parameter

<list>	the COSY-List to edit; <list> $\in$[-32768, 32767]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: RA 0000

Name

`COSY:CLR0`
Clear a Sync.

Synopsis

Instruction:	`COSY:CLR0 <Sync>`
Reply:	`<rc>`

Description

This instruction broadcasts the clearing for the given sync.

Parameter

<Sync>	the sync to clear; for sync range see chapter 7.3.1. Syncs on page 76.

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:
Instruction:
Reply: `RA 0000`

Name

`COSY:FIRS?`

List the first element of a COSY-List.

Synopsis

Instruction: `COSY:FIRS? <list>`

Reply: `<rc> COSY:FIRS <string>,<ID>`

Description

This instruction lists the first element of a COSY-List.

Parameter

<list>	the COSY-List; <list> ∈[-32768, 32767]
<string>	the instruction as it was used to edit the COSY-List
<ID>	a unique id of this COSY-List element

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: list first element of an injector program

Instruction:
```
COSY:FIRS? 33
```

Reply:
```
RA 0000 COSY:FIRS
"COSY:ISEQ 33,"DRAW 5.0,9,0,-2.6"",15747684
```

Name

COSY:LAST?

List the last element of a COSY-List.

Synopsis

Instruction:	`COSY:LAST? <list>`
Reply:	`<rc> COSY:LAST <string>,<ID>`

Description

This instruction lists the last element of a COSY-List.

Parameter

<list>	the COSY-List; <list> $\in$[-32768, 32767]
<string>	the instruction as it was used to edit the COSY-List
<ID>	a unique id of this COSY-List element

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)

Comment:	list first element of an injector program
Instruction:	`COSY:LAST? 33`
Reply:	`RA 0000 COSY:LAST "COSY:ISEQ 33,"INJT 302"",15748124`

Name

COSY:NEXT?
List the next element of a COSY-List.

Synopsis

Instruction: `COSY:NEXT? <list>,<IDin>`
Reply: `<rc> COSY:NEXT <string>,<IDout>`

Description

This instruction lists the next element of a COSY-List relatted to the given ID.

Parameter

<list>	the COSY-List; <list> ∈[-32768, 32767]
<IDin>	a unique id of a COSY-List element
<string>	the instruction as it was used to edit the COSY-List
<IDout>	a unique id of this COSY-List element

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0300 end of list

Examples

(a) Comment: list successor of first element of an injector program
Instruction: `COSY:NEXT? 33,15747684`
Reply: `RA 0000 COSY:FIRS "COSY:ISEQ 33,"EJCT 4.0,9,0,-2.6"",15747832`

(b) Comment: list successor of last element of an injector program
Instruction: `COSY:NEXT? 33,15748124`
Reply: `RE 0300 COSY:NEXT? 33,15748124`

Name

COSY:PREV?

List the prevous element of a COSY-List.

Synopsis

Instruction:	`COSY:PREV? <list>,<IDin>`
Reply:	`<rc> COSY:PREV <string>,<IDout>`

Description

This instruction lists the privious element of a COSY-List relatted to the given ID.

Parameter

<list>	the COSY-List; <list> $\in$[-32768, 32767]
<IDin>	a unique id of a COSY-List element
<string>	the instruction as it was used to edit the COSY-List
<IDout>	a unique id of this COSY-List element

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 0301 begin of list

Examples

(a)	Comment:	list predecessor of last element of an injector program
	Instruction:	`COSY:PREV? 33,15748124`
	Reply:	`RA 0000 COSY:PREV "COSY:ISEQ 33,"SMIX 0.2,9,1,3"",15747980`
(b)	Comment:	list predecessor of first element of an injector program
	Instruction:	`COSY:PREV? 33,15747684`
	Reply:	`RE 0301 COSY:PREV? 33,15747684`

Name

COSY:CHLD?
List "child" COSY-Lists of a COSY-List.

Synopsis

Instruction:	`COSY:CHLD? <list>`
Reply:	`<rc> COSY:CHLD <child>,<IDout>`
Instruction:	`COSY:CHLD? <list>,<IDin>`
Reply:	`<rc> COSY:CHLD <child>,<IDout>`

Description

The instruction lists a "child" of the specified COSY-List. A child is another COSY-List that is started from this COSY-List (see COSY:RUN on page 99). The id returned is a unique id of this child element.

Parameter

<list>	the COSY-List; <list> $\in$[-32768, 32767]
<child>	the child COSY-List; <child> $\in$[-32768, 32767]
<IDout>	a unique id of this COSY-List element
<IDin>	a unique id of a "child COSY-List" element

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0300 end of list

Examples

(a)	Comment:	list first child of COSY-List 2
	Instruction:	`COSY:CHLD? 2`
	Reply:	`RA 0000 COSY:CHLD 33,3034084`
(b)	Comment:	list next child of COSY-List 2
	Instruction:	`COSY:CHLD? 2,3034084`
	Reply:	`RA 0000 COSY:CHLD 9,3034172`
(c)	Comment:	list first child of COSY-List 33
	Instruction:	`COSY:CHLD? 33`
	Reply:	`RE 0300 COSY:CHLD? 33`

Name

`COSY:SND`

Broadcast a Sync via instruction.

Synopsis

Instruction: `COSY:SND <Sync>`

Reply: `<rc> COSY:SND <Sync>`

Description

This instruction allows a controller to broadcast a sync among modules.

Parameter

<Sync> the sync to send; for sync range see chapter 7.3.1. Syncs on page 76.

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

`COSY:STAT?`

Get COSY status.

Synopsis

Instruction: `COSY:STAT?`

Reply: `<rc> COSY:STAT <NumRun>`

Description

This instruction returns the number of COSY-Lists currently running.

Parameter

<NumRun> number of running COSY-Lists

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

COSY:STRT

Start a COSY-List.

Synopsis

Instruction: COSY:STRT <list>

Reply: <rc> COSY:STRT <list>

Description

This instruction starts the given COSY-List.

Parameter

<list> the COSY-List to start; <list> ∈[-32768, 32767]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 0901 no such COSY-List

Examples

(a) Comment:

Instruction:

Reply: RA 0000

Name

`COSY:RSET`

Reset COSY status.

Synopsis

Instruction: `COSY:RSET`

Reply: `<rc> COSY:RSET`

Description

Resets the COSY status. Includes clearing of all syncs in the module, resetting all clocks and stopping all running COSY-lists.

Parameter

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

Name

`COSY:TOUT`

Set timeout for COSY.

Synopsis

Instruction: `COSY:TOUT <Time>`

Reply: `<rc> COSY:TOUT <Time>`

Description

Sets timeout for COSY. If after starting a list no command is executed for the given time then an error (with event "EE 0308 COSY timed out.") is generated.

Parameter

<Time>	<Time> in minutes for Agilent 1200 series modules, in seconds for G3010A, range is [0.0, 10000.0]

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: `RA 0000`

7.5.2. COSY Actuals

Name

ACT:INST?
Get actual instruction.

Synopsis

Instruction: `ACT:INST? <list>`
Reply: `<rc> ACT:INST <list>,<Inst>`

Description

This instruction returns the actual instruction element COSY is executing.

Parameter

<list> the COSY-List considered; <list> $\in$[-32768, 32767]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0901 no such COSY-List

Examples

(a) Comment: get actual instruction during injector program
Instruction: `ACT:INST? -2`
Reply: `RA 0000 ACT:INST -2,"SRNG 3,0,9"`

7.5.3. COSY Method Parameter

Name

MAIN

Set Main-COSY-List.

Synopsis

Instruction:	`MAIN <list>`
Reply:	`<rc> MAIN <list>`
Instruction:	`MAIN?`
Reply:	`<rc> MAIN <list>`

Description

This instruction defines the Main-COSY-List. This is the list that gets started when a controller starts an analysis. The query returns the current Main-COSY-List.

Parameter

<list> the Main-COSY-List; <list> $\in$ [-32768, 32767]

Default Values

-2

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: get Main-COSY-List
Instruction: `MAIN?`
Reply: `RA 0000 MAIN -2`

7.5.4. Debug Tools

Name

`COSY:MEM?`

Get memory usage for COSY.

Synopsis

Instruction: `COSY:MEM?`

Reply: `<rc> COSY:MEM <MemRam>,<MemRamCont>,<nLists>,<nElem>`

Description

This instruction returns information about memory usage by COSY as well as the number of COSY-Lists and the total number of elements in all COSY-Lists.

Parameter

<MemRam>	memory used in normal RAM
<MemRamCont>	memory used in none volatile RAM
<nLists>	number of COSY-Lists in the system
<nElem>	number of elements in all COSY-Lists

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get memory usage

Instruction: `COSY:MEM?`

Reply: `RA 0000 COSY:MEM 5336,684,13,99`

Name

COSY:DUMP

Dump a COSY-List.

Synopsis

Instruction: COSY:DUMP <list>

Reply: <rc> COSY:DUMP <list>

Description

This instruction prints the given COSY-List to an RS232 terminal.

Parameter

<list> COSY-List to print; <list> $\in$ [-32768, 32767]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply: RA 0000

8. About Performing an Analysis

8.1. General Description

An Agilent 1200 Series HPLC system is designed to perform an analysis with each Agilent 1200 Series HPLC module contributing to by offering its individual capabilities. The main vehicle for 'playing' together is the connection via CAN from module to module. This connection provides (apart from normal data exchange) the real time synchronisation neccessary for reproducibility, customization and automation.

Especially, with respect to customization and multi controller functionality, and last not least to the integrity and tracebility (GLP) of the analysis, the CAN connection replaces and enhances the APG remote connection as used in 1050/1090 HPLC series and a variety of Agilent's analytical products. For backward compatibility the APG remote connector is still included but needed only if products of previous generations are used. The remote behaviour is identical to the 1050/1090 series and will not be discussed in this chapter.

8.2. Principles

All CAN wired modules start the analysis simultanously and end the analysis simultanously. The **first step** towards analysing an aliquot of a sample is to wait until all controllers connected to any of the CAN devices have given their ready indication (See instruction CRDY). This means that if the 'Handheld Controller G1323B' is about to perform a sequence, each analysis of the sequence is carried out when the PC based chemstation(s) has(have) finished data evaluation and report {and has(have) sent CRDY!}.

The **second step** towards working is to synchronously start each CAN devices' COSY list. Accompanied with this start is a number to later identify this analysis. This 'Analysis Counter' is the one of the previous analysis incremented by 1. The device that increments and initiates the start of the COSY lists, is the one that has received the initial kick off (typically the STRT instruction, but also possible due to a remote start request puls or a time programmed start).

The **third and last step** of the analysis is to wait til all CAN wired modules are done with their COSY lists, then simultanously change the state to 'no analysis' again.

Now the analysis is **done**!!! but: there was neither gradient nor rawdata generation, let alone the explanation what a second or third autosampler (CAN-connected) would do. The answer lies in the above mentioned COSY lists. These lists contain each modules' contribution to the analysis. If a list is **empty** the module will take part in the analysis with the three steps explained above, but will not do nothing like injection, balance, wait a while, gradient .....

The COSY lists can be very complicated and tricky, like for handling online coupling to a second Agilent HPLC system, or for synchronizing a complex dissolution cycle, but **in general the lists are extremely simple**. Each module provides a default COSY list that makes it contribute to a system to behave as did 1050 and 1090. (This is clearely not the best but caused no discussion!). The default lists are explained and listed later in this document.

8.3. Instruction to Deal with the Analysis

STRT

Start the analysis. It is enough to send the instruction to only one of the CAN devices. It doesn't matter which of the devices is used.

The instruction is rejected if either leak, shutdown is active or if an analysis or run is already in progress.

ABRT

Abort the analysis. If only a long runtime should be stopped, the instruction STPR can be used, the analysis will proceed.

COSY:NEW <list number>

Generates a new,empty cosy list. If a list with the same number existed it is now empty. The numbers must be non zero positiv. (negativ numbers are used for internally defined lists.

MAIN <list number>

Defines the COSY list to be used with the next analysis. This parameter is a method parameter. Listing of the actual parameter is done with MAIN?
MAIN -2 selects the default COSY list.

DIAG:LIST "ACHG, AHIS, RUN, ALMT, ANFO, ASTT, AERR"

List analysis related diagnostic buffers:

ACHG: Buffer of parameters that have been changed during the last analysis Cleared with each start of a new analysis
AHIS: Buffer of the last analyses with time, type of start, type of end
RUN: Buffer of the last starts of the runtime
ALMT: Limit events generated during the last analysis
ANFO: Info events generated during the last analysis
ASTT: State change events during the last analysis
AERR: Error Event generated during the last analysis. Typically only one entry is found since any statechange to error aborts the analysis.

ACT:CNT? "CNAN"

Lists the counter of the analyses, associated limit event is EF 0403

ACT:CNT? "CRUN"

Lists the counter of runs, associated limit event is EF 0402

ACT:ATIM?

Queries for the actual time since the analysis was started. If no analysis is active, 0 is returned

CRDY <ready/not ready>

Allowes all controllers to set 'not ready' for the next analysis. The flags are multi controller functional, that means that the flag is automatically cleared to ready if a controller is not active anymore (like closed LAN session, unplugged G1323, ...).

8.4. Elements to Build an Analysis

The elements used in 1050/1090 series were: Balance, Injection cycle, Gradient + Rawdata, Postrun. These elements are still available (and used by the default list) but are now only a few of what is available. The principle of what is called an *element* is that it must have a certain **duration**, must have a begin and an end. This is essential because the **COSY list provides the means to synchronize the begin of an element to the end of another element anywhere in the CAN wired system!** In addition to these elements any instruction can be part of a COSY list. This means that parameters could be changed at any time during the analysis (totally reproducible since the COSY list is part of the method and neither controller nor communication is involved during the analysis execution!). Setting a parameter is considered to have no time extension, allthough in reality it takes a few milliseconds.
The following table summarizes the elements of the Agilent 1200 Series devices that build a standard HPLC-analysis.It's neither complete by set of module nor complete by time consuming elements.

Pumps	Autosampler	Column Compartment	Detectors
flow gradient (run)	all single movements that are used for the injection	temperature change (run)	balance
composition gradient (run)			generate rawdata (run)
postrun	postrun	postrun	postrun

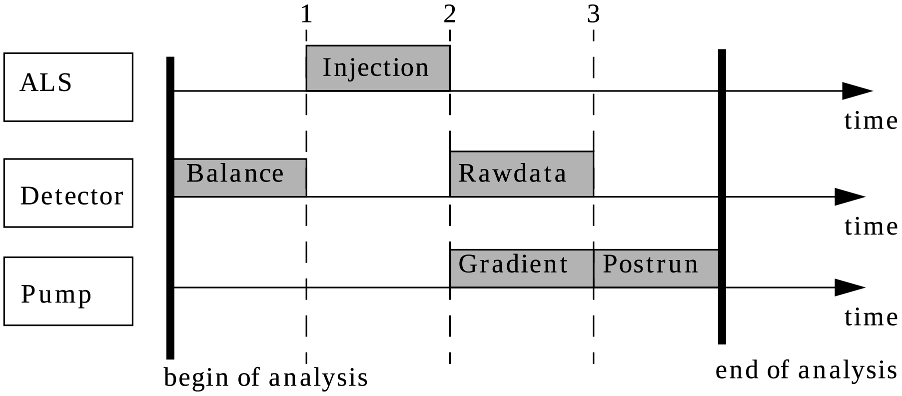
Pumps	Autosampler	Column Compartment	Detectors
external contacts	external contacts		external contacts
wait all lists started	wait all lists started	wait all lists started	wait all lists started
wait for volume	wait injector program finished	switch CSV	
wait for piston reversal		wait for left/ right tempera-ture	
wait till pressure is reached			
flow ramp			
(composition ramp)			

The shaded cells are the new ones compared to 1050/1090 Series.
The COSY list mechanism allows to realtime synchronize the elements of the above table in any order. This is especially of interrest if a customer wants to adapt the order of execution/and or repetition to his special problem.

8.5. Main COSY list

8.5.1. Example for an application depending set of COSY lists

An Agilent 1200 system consisting of an autosampler/injector, a detector and a pump does a 'standard' analysis by linking the shaded actions of the following drawing:



The timing is indicated by the vertical dashed lines numbered 1,2,and 3. For the COSY lists we call these **syncs**. The number of a sync is propagated over the CAN to be available in parallel for all modules connected to the CAN. To compile a COSY list, the picture is evaluated for each device from left to right. For the ALS this results in:

Wait for Sync 1
Draw Sample then switch valve and send Sync 2

Translated to the instructions for a COSY list this looks like:(the text after ! is comment)

```
COSY:NEW 5                    ! generate a new list called 5
COSY:SYNC 5,1                 ! wait for sync numbered 1
COSY:ISEQ 5,"DRAW ..."        !Draw sample
                              !ISEQ is the command to SEQentially execute the
                              !Instruction inside the double quotes
COSY:ISEQ 5, "INJT",2         !Switch valve and send sync 2
```

The next time axis to translate into a list is the one of the detector:

Balance then send sync 1
wait for sync 2
generate rawdata
wait for sync 3
stop rawdata generation

As COSY instructions:

```
COSY:NEW 5
COSY:ISEQ 5,"PRPC",1          !Balance than send sync 1
COSY:SYNC 5,2                 !Wait for sync 2
COSY:ISEQ 5, "RUNL"           !Start runtime
COSY:SYNC 5,3                 !Wait for sync 3
COSY:ISEQ 5,"STPR"            !Stop runtime
```

The Pump's time axis evaluation results in :

Wait for sync 2
start gradient, when done send sync 3
start postrun

As COSY instructions:

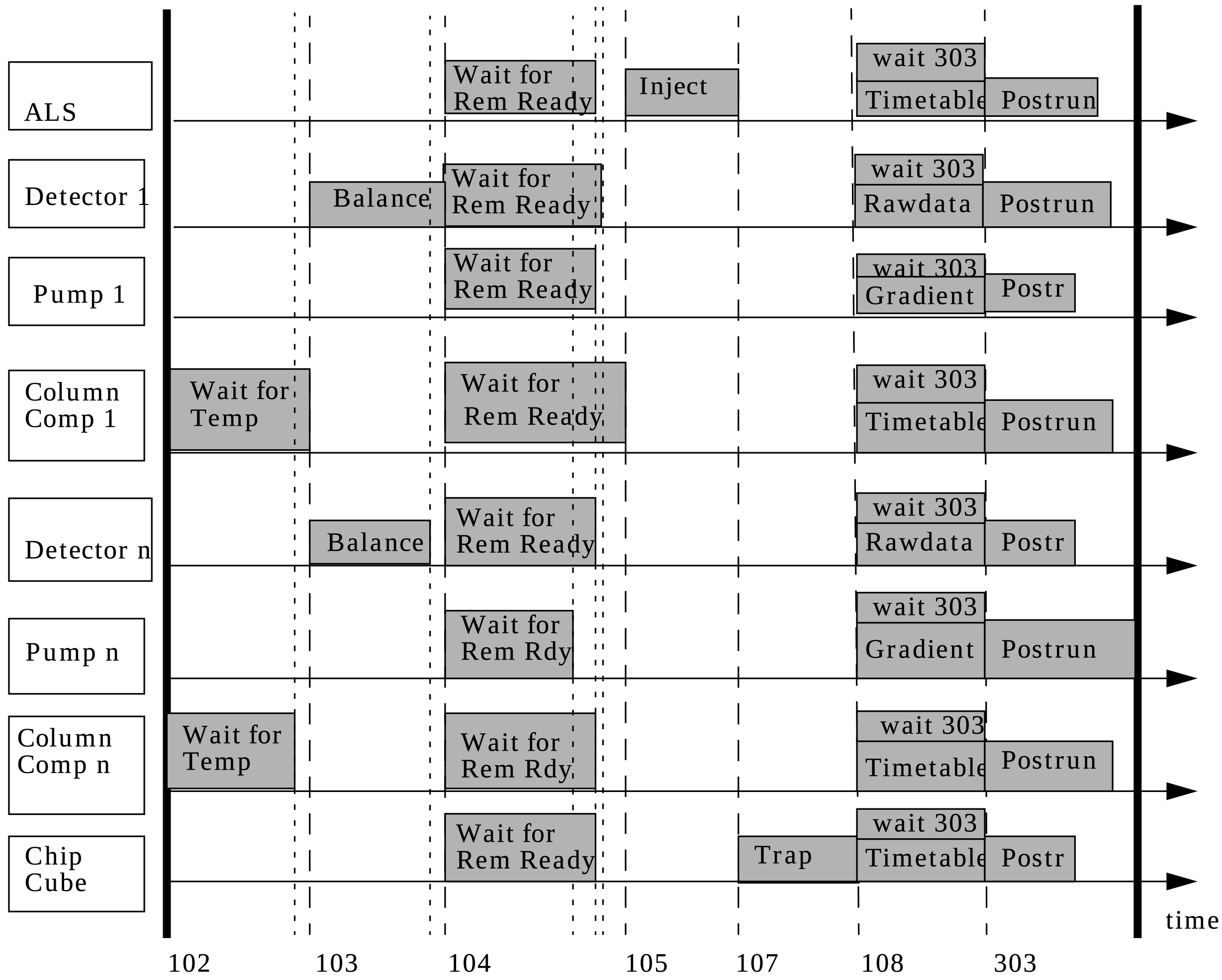
```
COSY:NEW 5
COSY:SYNC 5,2                 !Wait for sync2
COSY:ISEQ 5,"RUNL",3          !Start runtime, sync 3 if stoptime is reached
COSY:ISEQ 5,"RPST"            !Start posttime
```

8.5.2. Default main COSY list

As state-of-the-art controllers do not provide the above shown timing picture as graphical user interface, there are more complex main COSY lists implemented to allow at least some 'plug and play' functionality. This is (for each module) the default main COSY list "-2" along with all its children.

This is namely due to allow multiple detectors, pumps and column compartments and the obsolescence of an Agilent LC autosampler, presence of a ChipCube as part of one CAN wired system.

The following picture shows almost the whole provided functions by the default lists "-2" and their children:



The default list consists of a number of phases. The first three phases are for initializing issues (mainlist startup, send syncs, clear syncs). They are not included in the picture. The others represent functional blocks (wait for temperature, ...). Every phase is entered and left synchronously by all modules. If a module has nothing to do within a phase it simply waits for all other modules having this phase finished.

From left to right: All column compartments **wait for the temperature** to reach the setpoint. The dotted lines indicate that a 'multiple wait' is in progress. The following sync will be generated once all 'multiple' actions are done, in our case when both column compartements have reached their temperature setpoint. In the COSY list this is implemented as a counter named 103 that is incremented in startup phase of cosylist and decremented when the temperature reaches the setpoint. If the counter reaches 0 the associated sync is propagated. This mechanism applies for all initialization phases and for the phases "wait for temperature", "prepare/balance", "wait for remote ready" and "wait for inject". Counters are also identified by numbers. In the default COSY lists the counter identifier for the upper mentioned phases are 101 to 105, the associated syncs have the same number as the counter. Incrementing and decrementing counters is also propagated via CAN, so each device keeps track of the counter value.
Once the last column compartment reached its temperature, the **prepare phase** is started, i.e. for the detectors perhaps a **balance** and for other modules any preparation issue. The injectors send a prepare pulse on the remote line to initiate prepare of all external devices. Again it is a multiple

wait.
The next action is also common to all devices: **Wait for ready on the remote line**. This is for backward compatibility with other HP/Agilent analytical products and for including external modules that have a prepare-phase, e.g. detectors.
When the last device has decoded a ready on the remote not ready line, the phase before the injection is started. Right now this phase is empty, i.e. all modules just decrement the counter 106 and wait for "counter = 0". Afterwards the **injection** phase is started with sync 106 and leads to decrement counter 107 if done. The injection is not made up of one single command but may consist of any number of commands (synchronized also with a COSY list) to form an injector program. The injection cycle/Injector program is specific to the ALS and will not be explained here.
Afterwards the **trapping** phase is entered if a Chip Cube is present.
The next block is the synchronously started run of each device. The detectors typically deliver **raw-data**, the pumps typically start the composition **gradient** and the column compartments start their **time tables**. This is all included in the default list but mostly not needed. The 1050/1090 Series defined the run to be done with the first runtime counted up to the STPT parameter. The COSY mechanism for this behaviour is to 'fork' the run to be executed in parallel to waiting for the end. The first device that reaches the end sends sync 303. All others are waiting for **sync 303 in parallel** to their run and now use the instruction STPR to stop the run.
The **postrun** is the last action of each device. Since the **analysis is completed** as soon as the last action in all lists is done, the longest postrun will terminate the analysis.
Again: these default COSY lists are complex due to the fact that 'plug and play' behaviour is implemented. A controller that knows about the connected devices, could use much easier lists (see "Example for an application depending set of COSY lists"). Example: if no remote devices are connected, no 'wait for remote ready' is neccessary. Isocratic pumps do not use timetables except if external contacts or flow gradients are used. A postrun is used for reconditioning the system for a while. A postrun (the longest) in one of the devices would be enough.

The mainlist startup phase is for incrementing all used counters. A special command terminates this phase "COSY:SYNP". If a list comes to this command, the execution is delayed until all modules participating in an analysis have reached "COSY:SYNP"-command. After a timeout of two seconds the lists proceed regardless all modules reaching the command. This feature is for being able to work with modules having an empty mainlist. The implementation with counters in the default mainlist (-2) wouldn't work as the mainlist of different modules may start slightly delayed.

So if a controller implements its own main COSY list for each module, it should take care for the startup phase. Additionally initializing the COSY environment may be helpful - clear all relevant syncs prior to wait for them within the list.

Remark: Predefined COSY lists have negative numbers and cannot be modified.

To get an impression about how the list looks like please find here an exemplar COSY list "42" that is very close to the default list "-2" of the Agilent 1200 Series Column Thermostat:

```
/*----------------------------------------------------------------------*/
/*----------------------------------------------------------------------*/
"cosy:new 42,1",                    /* GLOBAL */

/*----------------------------------------------------------------------*/
/*---------------  increment all the used counters  --------------------*/
"cosy:inc 42,101,101",              /* "all the syncs sent"             */
"cosy:inc 42,102,102",              /* "all the syncs cleared"          */
"cosy:inc 42,103,103",              /* "temperature is ready"           */
"cosy:inc 42,104,104",              /* "prep done"                      */
"cosy:inc 42,105,105",              /* "remote is ready                 */
"cosy:inc 42,106,106",              /* "injection can begin"            */
"cosy:inc 42,107,107",              /* "injection done/trap can begin"  */
"cosy:inc 42,108,108",              /* "trap done/run can begin"        */
```

```
"cosy:synp 42",                     /* initial synchronization of 1st phase  */

/*-------------------------------------------------------------------*/
/*---------------    send all the syncs necessary   --------------------*/
"cosy:dec 42,101,101",
"cosy:sync 42,101",                 /* sync: "all the syncs sent" */

/*-------------------------------------------------------------------*/
/*---------------    clear all the syncs necessary   -------------------*/
"cosy:clr 42,303",                  /* sync: Stop */
"cosy:clr 42,80000",                /* sync: private stop */
"cosy:dec 42,107,107",
"cosy:dec 42,108,108",

"cosy:dec 42,102,102",
"cosy:sync 42,102",                 /* wait for "all the syncs cleared" */

/*-------------------------------------------------------------------*/
/*---------------    wait temperature ready   -------------------------*/
"cosy:iseq 42,'wltr -1'",           /* wait for left temp rdy, use TOUT */
"cosy:iseq 42,'wrtr -1'",           /* wait for right temp rdy, use TOUT */

"cosy:dec 42,103,103",
"cosy:sync 42,103",                 /* sync: "all col comps' temps rdy" */

/*-------------------------------------------------------------------*/
/*---------------    prep and wait prep done  -------------------------*/
"cosy:iseq 42,'ATOG'",              /* toggle valve if appropriate */

"cosy:dec 42,104,104",
"cosy:sync 42,104",                 /* sync: "all prepare phases done" */

/*-------------------------------------------------------------------*/
/*-------------------    wait remote ready ----------------------------*/
"cosy:iseq 42,'wrmr -1'",
"cosy:dec 42,105,105",
"cosy:sync 42,105",                 /* sync: "all remwaits are done" */

/*-------------------------------------------------------------------*/
/*----------    wait between "remote ready" and "injection"  -----------*/
"cosy:dec 42,106,106",
"cosy:sync 42,106",                 /* sync: "begin of injection"       */

/*-------------------------------------------------------------------*/
"cosy:sync 42,107",                 /* sync: end injection */
"cosy:sync 42,108",                 /* sync: end trap */
/*------------------------    start -----------------------------------*/
"cosy:ipar 42,'rung',80000",        /* start runtimer with sync Stop */
"cosy:sync 42,'303,80000'",         /* wait for sync Stop or runtime done */
"cosy:iseq 42,'stpr'",              /* stop runtimer if still running */
"cosy:iseq 42,'rpst'",              /* start postrun */
/*-----------------------------------------------------------------*/
```

8.6. Pseudo Code for a COSY list Compiler

Assumed is that the graphical representation used in the previous chapters is given as input to the compiler.

```
for each sync define a unique, non zero, positiv number
for each device do
        add 'COSY:NEW'
        for each device's action do
                if the action is 'sequencial'
                        if the action's position is not 'analysis begin'
                                add 'CLR' for the action's starting sync
        for each device's action do
                if the action's termination is 'multi'
                        if the action's position is not 'analysis end'
                                add 'INC' for the action's  counter
        from left to right, with each  action do
                if the action's position is not 'analysis begin'
                        add 'wait sync' for the action's starting sync
                if the action is parallel
                        add instruction using COSY:IPAR
                else
                        add instruction using COSY:ISEQ
                if the action's end is 'multi wait'
                        add 'DEC' for the action's counter
                else
                        if the action's position is not 'analysis end'
                                add the action's ending sync
```

From the above mini spec the following data structures are derived

```
        COSY LIST:
                LIST OF DEVICES
        DEVICE:
                LIST OF ACTIONS
        ACTION:
                struct:         {Starting sync::long
                                Counter number::long
                                Instruction::string
                                Position::analysis begin, middle, analysis end
                                Concurrency::[parallel, sequential]
                                Termination::[multi, sync]
                                Ending sync::long}
```

9. Parameter Handling Instructions

Name

PARA:RSET
reset all parameter to default values

Synopsis

Instruction: PARA:RSET
Reply: <rc> PARA:RSET

Description

All parameters of the actual method are set to their default values.
The timetable is deleted.
System parameters are set to their default values.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: reset method and sytem parameter to their default values, clear timetable
Instruction: PARA:RSET
Reply: RA 0000 PARA:RSET

Name

PARA:DFLT
reset method parameter to default values

Synopsis

Instruction: PARA:DFLT
Reply: <rc> PARA:DFLT

Description

All parameters of the actual method are set to default values.
Timetable is deleted.
System parameters are kept as they are.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: set actual method parameter to their default values and clear timetable
Instruction: PARA:DFLT
Reply: RA 0000 PARA:DFLT

Name

PARA:HOME
set method parameters to pre time table values

Synopsis

Instruction: PARA:HOME
Reply: <rc> PARA:HOME

Description

All parameters of the actual method are set to their values as they were before an intrinsic change like timetable, ramp, calibration... occured . When the stop time has elapsed this is done automatically.
System parameters are not affected.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: set actual method parameters to their pre timetable values
Instruction: PARA:HOME
Reply: RA 0000 PARA:HOME

Name

`PARA:AHOM`

reset method parameters to their pre analysis values

Synopsis

Instruction: `PARA:AHOM`
Reply: `<rc> PARA:AHOM`

Description

All parameters of the actual method are set to the values they had before the analysis cosy list was started. This is done automatically if a analysis terminates.
System parameter are not affected.

Parameters

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set actual method parameter to their pre analysis values
Instruction: `PARA:AHOM`
Reply: `RA 0000 PARA:AHOM`

Name

`PARA:MOD <mod>`

set parameter modification info

`PARA:MOD?`

query parameter modification info

Synopsis

Instruction:	`PARA:MOD <mod>`
Reply:	`<rc> PARA:MOD <mod>`
Instruction:	`PARA:MOD?`
Reply:	`<rc> PARA:MOD <modified>`

Description

Every HP1050 module controls a 'method modification flag'. This flag is set automatically if any parameter of the actual method is changed. It is listed using the query PARA:MOD?. The flag can be changed using the PARA:MOD instruction.

Parameters

`<modified>`	0 or 1 (1 means modified)

Default Values

flag is set to 0 at hotstart or at power on.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	query 'method modification flag'
	Instruction:	`PARA:MOD?`
	Reply:	`RA 0000 PARA:MOD 1`
(b)	Comment:	clear 'method modificaion flag'
	Instruction:	`PARA:MOD 0`
	Reply:	`RA 0000 PARA:MOD 0`

Name

FIRS?
LAST?
<keyword>:NEXT?
<keyword>:PREV?
list first/last/previous/next parameter

Synopsis

Instruction:	FIRS?
Reply:	<rc> <keyword> [<parameter>]
Instruction:	LAST?
Reply:	<rc> <keyword> [<parameter>]
Instruction:	<keyword>:NEXT?
Reply:	<rc> <keyword> [<parameter>]
Instruction:	<keyword>:PREV?
Reply:	<rc> <keyword> [<parameter>]

Description

The parameter of the actual method are ordered sequentially. This allows to list a first (last) parameter and to list a previous (next) parameter relative to a given parameter.
Another way to get the actual method parameter setpoints is to list the whole method (see instruction LIST).

Parameters

<keyword>	keyword of a method parameter
<parameter>	one or more parameter values (depending on individual method parameter)

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0300 end of list
(when trying to list the next of the last method parameter)
RE 0301 begin of list
(when trying to list the previous to the first method parameter)

Examples

(a)	Comment:	list first method parameter of the MWD
	Instruction:	FIRS?
	Reply:	RA 0000 PKWD 3
(b)	Comment:	list the next parameter (after the PKWD) of the MWD method
	Instruction:	PKWD:NEXT?
	Reply:	RA 0000 SIG1 254,4,550,80

Name

`AT:<keyword> <time>, <value>`
`AT:<keyword>? <time>`
timetable entry

Synopsis

Instruction:	`AT:<keyword> <time>,<value>`
Reply:	`<rc> AT:<keyword> <time>,<value>`
Instruction:	`AT:<keyword>? <time>`
Reply:	`<rc> AT:<keyword> <time>,<value>`

Description

Enter a new event in the timetable. Overwrites any existing timetable entry, if it is the same method parameter (keyword) and the same time.

Parameters

<keyword>	keyword of any timetable programmable method parameter
<value>	one or more parameter values

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0304` no timetable
  when trying to list a timetable entry, that does not exist

Examples

(a)	Comment:	timeprogram the external contact at time 3 min
	Instruction:	`AT:EC1 3,1`
	Reply:	`RA 0000 AT:EC1 3.00,1`
(b)	Comment:	query timetable entry
	Instruction:	`AT:EC1 3?`
	Reply:	`RA 0000 AT:EC1 3.00,1`

Name

AT:DEL
AT:<keyword>:DEL
AT:<keyword>:DEL <time>
delete timetable

Synopsis

Instruction: AT:DEL
Reply: <rc> AT:DEL

Description

deletes timetable entry:

AT:DEL	deletes complete timetable of the actual method.
AT:<keyword>:DEL <time>	deletes a single timetable entry
AT:<keyword>:DEL	deletes all timetable entries of <keyword>

Parameters

<keyword>	any timetable programmable method parameter
<time>	0 .. 99999.00 [min]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0304 no timetable
when trying to delete a single timetable entry, which does not exists

Examples

(a) Comment: delete whole timetable
Instruction: AT:DEL
Reply: RA 0000 AT:DEL

(b) Comment: delete singel timetable entry
Instruction: AT:EC1:DEL 3
Reply: RA 0000 AT:EC1:DEL 3.00

Name

AT:FIRS?
AT:LAST?
AT:<keyword>:NEXT? <time>
AT:<keyword>:PREV? <time>
list first/last/next/previous timetable entry

Synopsis

Instruction:	AT:FIRS?
Reply:	<rc> AT:<keyword> <time>,<value>
Instruction:	AT:LAST?
Reply:	<rc> AT:<keyword> <time>,<value>
Instruction:	AT:<keyword>:NEXT? <time>
Reply:	<rc> AT:<keyword> <time>,<value>
Instruction:	AT:<keyword>:PREV? <time>
Reply:	<rc> AT:<keyword> <time>,<value>

Description

List first (last) timetable entry or list next (previous) timetable entry relative to a specified timetable entry.
Another way to get all timetable entries is to list the whole timetable (see instruction LIST).

Parameters

<keyword>	any timetable programmable method parameter
<time>	0 .. 99999.00 [min]
<value>	one or more parameter values

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0301 begin of list
  when trying to read the previous entry of the first entry
RE 0300 end of list
  when trying to read the next entry of the last entry
RE 0304 no timetable
  when trying to list first or last entry out of an empty timetable

Examples

(a)	Comment:	list first timetable entry
	Instruction:	AT:FIRS?
	Reply:	RA 0000 AT:EC1 3.00,1
(b)	Comment:	list next entry from this timetable entry
	Instruction:	AT:EC1:NEXT? 3.00
	Reply:	RA 0000 AT:EC1 10.00,0

Name

SYS:FIRS?
SYS:LAST?
SYS:<keyword>:NEXT?
SYS:<keyword>:PREV?
list first/last/next/previous system parameter

Synopsis

Instruction:	SYS:FIRS?
Reply:	<rc> <keyword> <value>
Instruction:	SYS:LAST?
Reply:	<rc> <keyword> <value>
Instruction:	SYS:<keyword>:NEXT?
Reply:	<rc> <keyword> <value>
Instruction:	SYS:<keyword>:PREV?
Reply:	<rc> <keyword> <value>

Description

List first (last) system parameter or list next (previous) system parameter relative to a specified system parameter.
Another way to get all system parameter is to use the LIST instruction.
.

Parameters

<keyword>	any system parameter
<value>	one or more parameter values

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0301 begin of list
    when trying to list the previous parameter of the first one
RE 0300 end of list
    when trying to list the next parameter of the last one

Examples

(a)	Comment:	list first system parameter
	Instruction:	SYS:FIRS?
	Reply:	RA 0000 RAWF 1,120
(b)	Comment:	list next entry of this system parameter
	Instruction:	SYS:RAWF:NEXT?
	Reply:	RA 0000 TOUT 0

10. Method and Data Handling

If the instrument controller has no mass storage capabilities, the storage of methods can be done locally in the modules. A set of instructions supports all method handling. All current method parameters (Home Values, Time Tables, COSY Lists and System Variables) or only parts of it may be be stored under an arbitrary name (max. 8 characters long, case is ignored).
For method storage, three different memory areas (classes) are available, depending on the importance or lifetime of a method. Methods stored in the RAM will be lost with the next power-cycle, methods stored in the NVRAM will be lost with the next firmware-update (coldstart). To create an ever-lasting method, store it in the FLASH memory. Methods are allways referenced by their name.

The data handler is a simple facility for the storage of data strings. The user can create his own hirarchical storage structure by defining sections and keywords. Every section may contain a set of keywords, one keyword for each data string stored. In this way every data string is referenced by a section name and a keyword. All entries are stored in NVRAM.

10.1. Method management

Name

`FILE:RSET`
delete all methods

Synopsis

Instruction: `FILE:RSET [<class>]`
Reply: `<rc> FILE:RSET <class>`

Description

Deletes all methods of the specified memory class.

Parameters

`<class>`	memory class 1 RAM 2 NVRAM 3 FLASH

Default Values

Default memory class for method storage is 2, NVRAM.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0202` class protected

Examples

(a) Comment: delete all methods in default memory class
Instruction: `FILE:RSET`
Reply: `RA 0000 FILE:RSET 2`

(b) Comment: delete all methods in transient FLASH memory
Instruction: `FILE:RSET 3`
Reply: `RA 0000 FILE:RSET 3`

Name

`FILE:DEL`
delete a single method

Synopsis

Instruction: `FILE:DEL <"name">`
Reply: `<rc> FILE:DEL <"name">`

Description

Deletes the specified method.

Parameters

`<"name">`	name of method to be deleted

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0200` not existing
`RE 0202` class protected

Examples

(a) Comment: delete the method named "meth42"
Instruction: `FILE:DEL "meth42"`
Reply: `RA 0000 FILE:DEL "METH42"`

Name

FILE:STAT?
query method handler status

Synopsis

Instruction: `FILE:STAT?`
Reply: `<rc> FILE:STAT <"current">,<modified>,<total>, <used>,<#methods>`

Description

Returns the status of the method handler, consisting of the name of the actual method, a modify flag, the total size of the default memory class, the used amount of memory of this class and the over all number of methods stored in the module.

Parameters

<"current">	name of actual method (empty string means no method defined)
<modified>	modified flag 1 modified since time of last storage 0 not modified
<total>	total amount of memory available in the default memory class (memory size in bytes)
<used>	used amount of memory available in the default memory class (memory size in bytes)
<#methods>	number of methods (all memory classes together)

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query method handler status
Instruction: `FILE:STAT?`
Reply: `RA 0000 FILE:STAT "METH42",0,10240,3857,34`

Name

FILE:STAT?
query status of a method

Synopsis

Instruction: `FILE:STAT? <"name">`
Reply: `<rc> FILE:STAT <"name">,<"components">,<class>,<size>,<time>,<#ttlines>,<#cosylists>`

Description

Returns the complete status information of a method stored in the module. The information contains the method components, the memory class, the method size, the storage time, the number of timetable lines and the number of cosy lists stored with the specified method.

Parameters

<"name">	name of the method
<"components">	components of the method: "SYS" system variables "HOME" home values "TT" time tables "COSY" cosy lists
<class>	memory class in which method is stored 1 RAM 2 NVRAM 3 FLASH
<size>	size of method in bytes
<time>	time of last storage (seconds since 1970)
<#ttlines>	number of time table lines stored with the method
<#cosylists>	number of cosy lists stored with the method

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0200 not existing

Examples

(a) Comment: query method handler status
Instruction: `FILE:STAT? "meth42"`
Reply: `RA 0000 FILE:STAT "METH42","HOME,TT,COSY",2,513,825693510,33,7`

Name

FILE:SAVE
save method
FILE:FSAV
force saving method

Synopsis

Instruction:	`FILE:SAVE <"name">[,<"components">[,<class>]]`
Reply:	`<rc> FILE:SAVE <"name">,<"components">,<class>`
Instruction:	`FILE:FSAV <"name">[,<"components">[,<class>]]`
Reply:	`<rc> FILE:FSAV <"name">,<"components">,<class>`

Description

Save the currently used parameters to the specified method. Use FSAV to overwrite a method that already exists.

Parameters

<"name">	method name
<"components">	components of the method: "SYS" system variables "HOME" home values "TT" time tables "COSY" cosy lists "COSY:33" cosy list #33 and all related lists
<class>	memory class in which method shall be stored 1 RAM 2 NVRAM 3 FLASH, transient area

Default Values

Default components are the home values, time tables and the main cosy list (HOME, TT and COSY).

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0104 full
RE 0201 can't overwrite
RE 0202 class protected
RE 0203 internal error
RE 0204 mismatch (some of the specified components are not available, method is stored anyway)

Examples

(a) Comment: save actual home values, main cosy list and cosy list #5 as method named "meth42" into RAM
Instruction: `FILE:SAVE "meth42","HOME,COSY,COSY:5",1`
Reply: `RA 0000 FILE:SAVE "METH42","HOME,COSY,COSY:5",1`

(b) Comment: save home values into the default memory class ("meth42" is already existing)
Instruction: `FILE:SAVE "meth42","HOME"`
Reply: `RE 0201 FILE:SAVE "METH42","HOME",2`

(c) Comment: force saving the default components into the default memory class
Instruction: `FILE:FSAV "meth42"`
Reply: `RA 0000 FILE:FSAV "METH42","HOME,TT,COSY",2`

Name

FILE:LOAD
load method
FILE:FLOD
force loading method

Synopsis

Instruction:	`FILE:LOAD <"name">[,<"components">]`
Reply:	`<rc> FILE:LOAD <"name">,<"components">`
Instruction:	`FILE:FLOD <"name">[,<"components">]`
Reply:	`<rc> FILE:FLOD <"name">,<"components">`

Description

Loads the specified method. If the actual method has been modified before, use FLOD to overwrite it.

Parameters

<"name"> method name
<"components"> components to be loaded (if part of the method saved):

"SYS"	system variables
"HOME"	home values
"TT"	time tables
"COSY"	cosy lists stored with the method

Default Values

If no components are specified, all available components are loaded.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0200 not existing
RE 0201 can't overwrite
RE 0203 internal error
RE 0204 mismatch (some of the components to be loaded are not accepted by the module, method is loaded partially)

Examples

(a) Comment: load all components of a method named "meth42"
Instruction: `FILE:LOAD "meth42"`
Reply: `RA 0000 FILE:LOAD "meth42","HOME,COSY"`

(b) Comment: load only a part of that method
Instruction: `FILE:LOAD "meth42","HOME"`
Reply: `RA 0000 "meth42","HOME"`

Name

`FILE:ANNO`

set/query annotation of the actual method

Synopsis

Instruction:	`FILE:ANNO <"annotation">`
Reply:	`<rc> FILE:ANNO <"annotation">`
Instruction:	`FILE:ANNO?`
Reply:	`<rc> FILE:ANNO <"annotation">`

Description

Set the annotation string for the actual method.
Query the annotation string of the actual method. If no annotation is available a zero string is returned.

Parameters

`<"annotation">` annotation string

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set the annotation for the actual method
	Instruction:	`FILE:ANNO "method by M.King"`
	Reply:	`RA 0000 FILE:ANNO "method by M.King"`
(b)	Comment:	get the annotation of the actual method
	Instruction:	`FILE:ANNO?`
	Reply:	`RA 0000 FILE:ANNO "method by M.King"`

Name

FILE:FIRS?
list method names, get first

Synopsis

Instruction: `FILE:FIRS?`
Reply: `<rc> FILE:FIRS <"name">`

Description

List the names of all methods stored in the module. Query the name of the first method.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0300 end of list

Examples

(a) Comment: get the first method
Instruction: `FILE:FIRS?`
Reply: `RA 0000 FILE:FIRS "METH42"`

Name

FILE:NEXT?

list method names, get next

Synopsis

Instruction: FILE:NEXT? <"previous">

Reply: <rc> FILE:NEXT <"next">

Description

List the names of all methods stored in the module. Query the name of the method following the method specified.

Parameters

<"previous">	name of previous method
<"next">	name of next method

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0200 not existing
RE 0300 end of list

Examples

(a) Comment: get the name of the method following the method named "meth42"
Instruction: FILE:NEXT? "meth42"
Reply: RA 0000 FILE:NEXT "METH43"

Name

FILE:AMET

define/query auto-on method

Synopsis

Instruction:	`FILE:AMET <"name">,<"attributes">`
Reply:	`<rc> FILE:AMET <"name">,<"attributes">`
Instruction:	`FILE:AMET?`
Reply:	`<rc> FILE:AMET <"name">,<"attributes">`

Description

Define a method that's activated when the module is turned on automatically (see instruction ATON).

Query the auto-on method name and attributes.

Attributes and method name are stored in NVRAM.

Parameters

`<"name">`	name of auto-on method	
`<"attributes">`	method attributes	
	"LOAD"	load method (if not specified the actual method will be kept)
	"OFF"	turn off module
	"ON"	turn on module
	"RUN"	start a run
	"CAN"	initiate auto-on at all other modules connected via CAN

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	define method "meth42" to be loaded and run at auto-on time
	Instruction:	`FILE:AMET "meth42","LOAD,ON,RUN"`
	Reply:	`RA 0000 FILE:AMET "meth42","LOAD,ON,RUN"`
(b)	Comment:	query the name and attributes of the auto-on method
	Instruction:	`FILE:AMET?`
	Reply:	`RA 0000 FILE:AMET "meth42","LOAD,ON,RUN"`

Name

`FILE:EMET`
define/query error method

Synopsis

Instruction:	`FILE:EMET <"name">,<"attributes">`
Reply:	`<rc> FILE:AMET <"name">,<"attributes">`
Instruction:	`FILE:EMET?`
Reply:	`<rc> FILE:AMET <"name">,<"attributes">`

Description

Define a method that's activated when the module detects an error condition.
Query the error method name and attributes.
Attributes and method name are stored in NVRAM.

Parameters

`<"name">`	name of error method	
`<"attributes">`	method attributes	
	"LOAD"	load method (if not specified the actual method will be kept)
	"OFF"	turn off module
	"ON"	turn on module
	"RUN"	start local run

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	define method "meth43" to be loaded and module to be turned off in case of an error
	Instruction:	`FILE:EMET "meth43","LOAD,OFF"`
	Reply:	`RA 0000 FILE:EMET "meth43","LOAD,OFF"`
(b)	Comment:	query the name and attributes of the error method
	Instruction:	`FILE:EMET?`
	Reply:	`RA 0000 FILE:EMET "meth43","LOAD,OFF"`

10.2. Data management

Name

FILE:DATA
store/read out data

Synopsis

Instruction:	`FILE:DATA <"section">,<"keyword">,<"data">`
Reply:	`<rc> FILE:DATA <"section">,<"keyword">,<"data">`
Instruction:	`FILE:DATA? <"section">,<"keyword">`
Reply:	`<rc> FILE:DATA <"section">,<"keyword">,<"data">`

Description

Store a data string under a user-defined keyword in an user-defined section.
Read out data entry referenced by section and keyword.

Parameters

`<"section">`	name of the section that keeps the specified keyword
`<"keyword">`	keyword that references the stored data
`<"data">`	data to be stored

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0104 full
RE 0205 no such entry

Examples

(a)	Comment:	store the string "M.King" under keyword "ID42" in section "USERS"
	Instruction:	`FILE:DATA "USERS","ID42","M.King"`
	Reply:	`RA 0000 FILE:DATA "USERS","ID42","M.King"`
(b)	Comment:	read out data string referenced by section "USERS" and keyword "ID42"
	Instruction:	`FILE:DATA? "USERS","ID42"`
	Reply:	`RA 0000 FILE:DATA "USERS","ID42","M.King"`

Name

`FILE:DATA:DEL`

delete single entry / whole section / all data

Synopsis

Instruction:	`FILE:DATA:DEL <"section">,<"keyword">`
Reply:	`<rc> FILE:DATA:DEL <"section">,<"keyword">`
Instruction:	`FILE:DATA:DEL <"section">`
Reply:	`<rc> FILE:DATA:DEL <"section">`
Instruction:	`FILE:DATA:DEL`
Reply:	`<rc> FILE:DATA:DEL`

Description

Delete a single data entry and the keyword that’s referencing it.
Delete all entries of the specified section and the section itself.
Delete all sections and all entries.

Parameters

`<"section">`	name of the section that keeps the entries to be deleted
`<"keyword">`	keyword that references the data entry to be deleted

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0205` no such entry

Examples

(a)	Comment:	delete data string referenced by section "USERS" and keyword "ID42"
	Instruction:	`FILE:DATA:DEL "USERS","ID42"`
	Reply:	`RA 0000 FILE:DATA:DEL "USERS","ID42"`
(b)	Comment:	delete section "USERS"
	Instruction:	`FILE:DATA:DEL "USERS"`
	Reply:	`RA 0000 FILE:DATA:DEL "USERS"`
(c)	Comment:	delete all
	Instruction:	`FILE:DATA:DEL`
	Reply:	`RA 0000 FILE:DATA:DEL`

Name

`FILE:DATA:STAT?`

query status of data handler

Synopsis

Instruction: `FILE:DATA:STAT?`

Reply: `<rc> FILE:DATA:STAT <#sections>,<nvram>`

Description

Query the data handler status information. Returns the number of sections currently defined and the amount of NVRAM used for data storage.

Parameters

`<#sections>`	number of sections defined
`<nvram>`	amount of NVRAM used (size in bytes)

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get data handler status

Instruction: `FILE:DATA:STAT?`

Reply: `RA 0000 FILE:DATA:STAT 5,798`

Name

`FILE:DATA:STAT?`
query status of a section

Synopsis

Instruction: `FILE:DATA:STAT? <"section">`
Reply: `<rc> FILE:DATA:STAT <#entries>,<time>`

Description

Query the status information of a section. Returns the number of data entries currently available in the section and the creation time of the section.

Parameters

`<"section">`	name of section
`<#entries>`	number of data entries
`<time>`	creation time of section (seconds since 1970)

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0205` no such entry (section not found)

Examples

(a) Comment: get section status
Instruction: `FILE:DATA:STAT? "USERS"`
Reply: `RA 0000 FILE:DATA:STAT 10,825693510`

Name

FILE:DATA:FIRS?
list sections, get first

Synopsis

Instruction: `FILE:DATA:FIRS?`
Reply: `<rc> FILE:DATA:FIRS <"firstsect">`

Description

List all sections defined in the data handler. Get the name of the first section.

Parameters

`<"firstsect">` name of first section

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0300 end of list

Examples

(a) Comment: get first section
Instruction: `FILE:DATA:FIRS?`
Reply: `RA 0000 FILE:DATA:FIRS "CONFIG"`

Name

`FILE:DATA:NEXT?`

list sections, get next

Synopsis

Instruction:	`FILE:DATA:NEXT? <"previoussect">`
Reply:	`<rc> FILE:DATA:NEXT <"nextsect">`

Description

List all sections defined in the data handler. Get the name of the section that follows the specified one.

Parameters

`<"previoussect">`	name of previous section
`<"nextsect">`	name of following section

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0205` no such entry (section not found)
`RE 0300` end of list

Examples

(a) Comment: get the section following the section named "CONFIG"

Instruction:	`FILE:DATA:NEXT? "CONFIG"`
Reply:	`RA 0000 FILE:DATA:NEXT "USERS"`

Name

FILE:DATA:FIRS?
list keywords of a section, get first

Synopsis

Instruction: `FILE:DATA:FIRS? <"section">`
Reply: `<rc> FILE:DATA:FIRS <"section">,<"firstkey">`

Description

List all keywords available in a section. Get the name of the first keyword.

Parameters

<"section">	name of the section
<"firstkey">	name of the first keyword in the section

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0205 no such entry (section not found)
RE 0300 end of list

Examples

(a) Comment: get first keyword of section "USERS"
Instruction: `FILE:DATA:FIRS? "USERS"`
Reply: `RA 0000 FILE:DATA:FIRS "USERS","ID1"`

Name

FILE:DATA:NEXT?
list keywords of a section, get next

Synopsis

Instruction: `FILE:DATA:NEXT? <"section">,<"previous">`
Reply: `<rc> FILE:DATA:NEXT <"section">,<"next">`

Description

List all keywords available in a section. Get the name of the keyword that follows the specified one.

Parameters

<"section">	name of the section
<"previous">	name of previous keyword
<"next">	name of following keyword

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0205 no such entry (section/keyword not found)
RE 0300 end of list

Examples

(a) Comment: get the keyword the keyword named "ID2"
Instruction: `FILE:DATA:NEXT? "USERS","ID2"`
Reply: `RA 0000 FILE:DATA:NEXT "USERS","ID3"`

11. Method Parameter

11.1. Principles of Using Methods

The overall idea of using a method is to allow the following control scheme of a Agilent 1200 system:

FIRST LOAD a method to all modules THEN START the analysis

This clearly means that during an analysis a controller (Chemstation, HandHeld, ...) does not need and should not interfere with the execution of the analysis. All jobs that contribute to the reproducibility of the analysis (in other words: all the realtime stuff) is handled by the Agilent 1200 modules.
All parameters needed for an analysis are stored in the modules non volatile (battery buffered) RAM. The modules never modify parameters by themselves.
For repeating the same analysis, it is allright to just start the analysis.

Parts of a method are:
Home Values: parameters that are typically changed according to the chemistry in use
like Flow, Composition, Wavelength, Temperature, Injection volume...
Time Table: parameters that are typically used in a time program
like Composition, Wavelength, External Contacts ...
Application List: How the analysis is done, the order of jobs to carry out, timing, synchronization
like Injector program and COSY Lists

The Agilent 1200 series modules always have a set of active method parameters. After power on the ones used before turning off are re-activated. Additionaly the different parts of a method can be stored with a user definable name. See chapter 'Method Handling'. This has nothing to do with saving a method on a controller's massstorage device (disc..) and is typically not used if a controller has its own memory available (like chemstation).

The different parts of a method can be read from a module either by quering each individual parameter/list entry or much easier by using the LIST CU. See also subchapter List CU of chapter 'Communication Units'. The instructions to list are

LIST "item, item, .."

where items are HOME for homevalues, TT for timetable entries, COSY:<number> for injector program and application lists.

11.2. Common Method Parameter

Name

STPT

Stoptime value for analysis

Synopsis

Instruction:	`STPT <time>`
Reply:	`<rc> STPT <time>`
Instruction:	`STPT?`
Reply:	`<rc> STPT <time>`
Instruction:	`ACT:STPT?`
Reply:	`<rc> ACT:STPT <actualtime>`

Description

The STPT parameter defines the time (in minutes for 1200 modules, in seconds for G3010A), after which the run time will stop automatically. A value of 0 means OFF (infinite run).
The ACT:STPT? query returns the actual runtime (in minutes for 1200 modules, in seconds for G310A), starting from zero. If not in run, 0 is returned.

STPT <time>....................sets stoptime for analysis
STPT?...............................query stoptime of analysis
ACT:STPT?.......................query actual analysis time

Parameters

`<time>`	0 .. 99999.00 [min] for 1200 modules 0 .. 6000000.0 [sec] for G3010A 0 means OFF (infinite run).
`<actualtime>`	0 .. 99999.00 [min] for 1200 modules 0 .. 6000000.0 [sec] for G3010A 0 means ’not in run’

Default Values

0 [min] (OFF)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set stoptime parameter to 5 minutes (1200 modules)
	Instruction:	`STPT 5`
	Reply:	`RA 0000 STPT 5.00`
(b)	Comment:	Query stoptime parameter setting (1200 modules)
	Instruction:	`STPT?`
	Reply:	`RA 0000 STPT 0`
(c)	Comment:	Query for actual runtime (1200 modules)
	Instruction:	`ACT:STPT?`
	Reply:	`RA 0000 ACT:STPT 2.14`

Name

POST

Posttime value for analysis

Synopsis

Instruction:	`POST <time>`
Reply:	`<rc> POST <time>`
Instruction:	`POST?`
Reply:	`<rc> POST <time>`
Instruction:	`ACT:POST?`
Reply:	`<rc> ACT:POST <actualtime>`

Description

The POST parameter defines the time (in minutes for 1200 modules, in seconds for G3010A), after which the postrun time elapses automatically.
The ACT:POST? query returns the actual posttime (in minutes for 1200 modules, in seconds for G310A), counting downto zero.
If no postrun is active, 0 is returned.
POST <time>........post analysis time
ACT:POST?...........query actual post time of analysis

Parameters

`<time>`	0 .. 99999.00 [min] for 1200 modules
	0 .. 6000000.0 [sec] for G3010A
`<actualtime>`	0 .. 99999.00 [min] for 1200 modules
	0 .. 6000000.0 [sec] for G3010A
	0 means 'no posttime'

Default Values

0 [min]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set posttime parameter (1200 modules)
	Instruction:	`POST 5`
	Reply:	`RA 0000 POST 5.00`
(b)	Comment:	Query posttime parameter setting (1200 modules)
	Instruction:	`POST?`
	Reply:	`RA 0000 POST 0`
(c)	Comment:	Query for actual posttime (1200 modules)
	Instruction:	`ACT:POST?`
	Reply:	`RA 0000 ACT:POST 2.14`

Name

ECn

Sets/query external contact 0,1,2 or 3

Synopsis

Instruction:	`ECn <on/off>`
Reply:	`<rc> ECn <on/off>`
Instruction:	`ECn?`
Reply:	`<rc> ECn <on/off>`
Instruction:	`AT:ECn <time>,<on/off>`
Reply:	`<rc> AT:ECn <time>, <on/off>`
Instruction:	`AT:ECn? <time>`
Reply:	`<rc> AT:ECn <time>,<on/off>`
Instruction:	`ACT:ECn?`
Reply:	`<rc> ACT:ECn <on/off>`

Description

ECn set contact n (0,1,2, or 3) to on or off. ECn? queries for the setpoint of contact n. ACT:ECn? queries for the actual state of the contact n. AT:ECn enters a time table entry for the contact n. These comands work with the optional contacts/BCD board.

ECn <on/off>....................external contact #n

ECn?.................................query external contact #n

Parameters

`<on/off>`	0 for not closed 1 for closed
`<time>`	0 .. 99999.00 [min]

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RA 0752 option not installed

Examples

(a)	Comment:	Set external contact 3 to closed
	Instruction:	`EC3 1`
	Reply:	`RA 0000 EC3 1`
(b)	Comment:	Query state of external contact 3
	Instruction:	`EC3?`
	Reply:	`RA 0000 EC3 1`
(c)	Comment:	Query for actual state of EC3
	Instruction:	`ACT:EC3?`
	Reply:	`RA 0000 ACT:EC3 1`
(d)	Comment:	Enter time table value for external contact 3, to open 5 minutes after start tt.
	Instruction:	`AT:EC3 5.00, 0`
	Reply:	`RA 0000 AT:EC3 5.00, 0`

12. Common System Parameter

Name

`TOUT`

Sets timeout value for the analysis

Synopsis

Instruction:	`TOUT <time>`
Reply:	`<rc> TOUT <time>`
Instruction:	`TOUT?`
Reply:	`<rc> TOUT <time>`
Instruction:	`ACT:TOUT?`
Reply:	`<rc> ACT:TOUT <actual>`

Description

The TOUT parameter defines the time (in minutes for 1200 modules, in seconds for G3010A) for a clock that is started with the end of an analysis. The time is counted back to 0. If no new analysis is stared until the counter reached 0 the time out error event is generated. For G3010A the timeout is only active, if a batch is turned on (see command BAT).

The ACT:TOUT? query returns the actual timout timer value (in minutes for 1200 modules, in seconds for G3010A). If not active, 0 is returned.

Parameters

`<time>`	0 .. 99999.00 [min] for 1200 modules 0 .. 6000000.0 [sec] for G3010A 0 means OFF (clock is not started and/or stopped).
`<actual>`	0 .. 99999.00 [min] for 1200 modules 0 .. 6000000.0 [sec] for G3010A 0 means ’not in run’

Default Values

0 [min] (OFF)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

`RE 0502` out of range

Examples

(a)	Comment:	set timeout parameter to 15 minutes (1200 modules)
	Instruction:	`TOUT 15`
	Reply:	`RA 0000 TOUT 15`
(b)	Comment:	Query the timeout parameter setting (1200 modules)
	Instruction:	`TOUT?`
	Reply:	`RA 0000 TOUT 15`
(c)	Comment:	Query for actual timeout value (1200 modules)
	Instruction:	`ACT:TOUT?`
	Reply:	`RA 0000 ACT:TOUT 2.14`

Name

REMC

APG Remote Configuration

Synopsis

Instruction:	`REMC <config>`
Reply:	`<rc> REMC <config>`
Instruction:	`REMC?`
Reply:	`<rc> REMC <config>`

Description

The REMC parameter defines the APG remote configuration. This has no efect at all on the module's behaviour if the modules are connected via CAN.

Parameters

`<config>` -1, 0.. 3

-1 means AUTO , for detectors and the column compartment this is the same as 0, for the pumps and the autosampler this is the same as 1 . For the autosampler with revision 2.03 (sept/oct 1996) or greater the AUTO value is the same as value 3.

0 means: no stop pulse will be generated if a time table was started via the APG remote start pulse

1 means: a stop pulse will be generated if a time table was started via the APG remote start pulse

2 means: a start request pulse will be generated if the STRT command is received via an interface (no stop pulse will be generated)

3 means: a APG remote startrequest pulse will be interpreted like the STRT command via an interface (e.g. the main COSY list is started).

4 means: a APG remote startrequest pulse generates the event "EV 0161 controller start request", stop pulse is generated according to 1.

Default Values

-1

for the Autosampler G1313A with revision 2.03 or greater, the default is set to 3.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

`RE 0502` out of range

Examples

(a) Comment: set remote configfruation to AUTO

Instruction: `REMC -1`

Reply: `RA 0000 REMC -1`

(b) Comment: Query the remote configuration

Instruction: `REMC?`

Reply: `RA 0000 REMC -1`

Name

NCFG
BCD/BIN Output Configuration (not for column comp)

Synopsis

Instruction:	`NCFG <BCD-source>,<BCD-format>`
Reply:	`<rc> NCFG <BCD-source>,<BCD-format>`
Instruction:	`NCFG?`
Reply:	`<rc> NCFG <BCD-source>,<BCD-format>`

Description

Defines the format and the source used to write a number out to the pins of the BCD Connector.Format can be either binary or BCD-coded, source can be either the current vial number of the autosampler or the value set with the next NOUT instruction. The column compartement can not be equiped with the optional harware, therfore the reply to NCFG is allways RE 0712 (option not installed).

Parameters

`<BCD-source>`	0 .. 1 0 : use current vial number (meaningless for non-autosamplers) 1 : use the value set with NOUT
<BCD-format>	0 .. 1 0 : output in BCD-coded format 1 : output in binary format

Default Values

<BCD-source>	0 for autosampler, 1 for non-autosamplers
<BCD-format>	0 (BCD-coded)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0502` out of range

Examples

(a)	Comment:	set binary output format; use value set with NOUT
	Instruction:	`NCFG 1,1`
	Reply:	`RA 0000 NCFG 1,1`
(b)	Comment:	Query the BCD-configuration
	Instruction:	`NCFG?`
	Reply:	`RA 0000 NCFG 1,1`
(c)	Comment:	write the number 42 in binary format to BCD-connector
	Instruction:	`NOUT 42`
	Reply:	`RA 0000 NOUT 42`

13. Commands & Queries

13.1. Controlling the Agilent 1200 Series modules

'Controlling' in this chapter is ment in a sense of turning on and off the modules's hardware (see LAMP 1, PUMP 1, THRM 1), starting and stopping an analysis (see STRT, STOP, ABRT) and invoking statechanges (see CLER, RSET)

13.2. Common Commands & Queries

Name

STRT
Start analysis
STRS
Start Analysis without other CAN Devices

Synopsis

Instruction: `STRT`
Reply: `<rc> STRT`

Description

The STRT and STRS command start the analysis. The module carries out the COSY list defined by the instruction 'MAIN <listname'. All modules connected via CAN start their analysis but in difference to the STRT command, at the STRS command they will neither start any COSYlist nor be able to prevent the analysis. The actual progres of the analysis can be monitored by 'ACT:ATIM?' with returns the time since the analysis was started. An analysis can be stopped with 'ABRT'.

Parameters

none

Default Values

none

Reply Codes <rc>

Common:
- RA 0000 no error, instruction has been accepted
- RE 0700 run already in progress
- RE 0701 no start while leak
- RE 0702 no start while shutdown
- RE 0703 no start while startup
- RA 0704 analysis already in progress
- RE 0708 no start while aborting

Pumps (binary-/isocratic-/quaternary-/capillary-/nano-):
- RE 2030 start rejected while pump off
- RE 2031 start rejected while purge in progress
- RE 2037 STRT while purging
- RE 2038 STRT while fast composition change

Column Compartment specific:
- RE 2801 start not allowed while off

Autosampler specific:
- RE 3036 start of an analysis during initialization
- RE 3045 start analysis while module not ready
- RE 3046 start analysis while module defect
- RE 3047 start analysis while module in service mode

Diode-Array Detector:
- RE 6043 No start while prepare

Variable Wavelength Detector:
RE 6041 start while reset
RE 6043 start while prep
RE 6044 start while scan

Fluorescence Detector:
RE 5609 C_ER_STARTWHILENOCELL
RE 5610 C_ER_STARTWHILEPMTCALC
RE 5611 C_ER_STARTWHILECALIB
RE 5614 C_ER_STARTWHILE_MISSING_FLF
RE 5615 C_ER_STARTWHILE_ADC_CALIB_FAILED
RE 5619 C_ER_STARTWHILE_LAMP_COVER_OPEN
RE 6043 C_ER_STARTWHILEPREP

Fraction Collector:
RE 3210 no start during initialization
RE 3701 no start while transport command is running
RE 3700 no start while door open

Well Plate Autosampler:
RE 3210 no start during initialization
RE 3211 no start while module not ready
RE 3210 no start while module defect
RE 3210 no start while module in service mode

Preparative Pump:
RE 2300 start while pump off
RE 2307 start while cover open
RE 2308 start while permanent encoder error
RE 2309 start while motor over temperature
RE 2310 start while permanent pressure read-out error
RE 2311 start while bottle error

Examples

(a) Comment: start analysis
Instruction: `STRT`
Reply: `RA 0000 STRT`

(b) Comment: start analysis private to one module
Instruction: `STRS`
Reply: `RA 0000 STRS`

Name

RUNL
start local run
RUNG
start global run
RPST
start postrun

Synopsis

Instruction:	RUNL
Reply:	<rc> RUNL
Instruction:	RUNG
Reply:	<rc> RUNG
Instruction:	RPST
Reply:	<rc> RPST

Description

These commands should be used as part of COSY lists to invoke the start of a gradient of the pump or to start rawdata generation of the detectors. The duration of a run is defined by the stoptime parameter (STPT), the duration of a postrun by the posttime parameter (POST)

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 2801 not allowed while column compartement is off
RE 0704 not allowed during analysis
RE 0703 not allowd during startup (after power is turned on)
RE 0702 not allowed while shutdown
RE 0701 not allowed while leak
RE 0700 not allowed while run

Examples

(a) Comment: start local run
Instruction: RUNL
Reply: RA 0000 RUNL

Name

STOP
stop analysis
STPR
stop run
STPP
stop postrun

Synopsis

Instruction:	STOP
Reply:	<rc> STOP
Instruction:	STPR
Reply:	<rc> STPR
Instruction:	STPP
Reply:	<rc> STPP

Description

Aborts the analysis (STOP), stops runtime (STPR) or stops postrun execution (STPP). STPR and STPP will only have an effect on the dedicated item to stop, that is, the analysis goes on. STOP will abort at once the tasks currently under way and will force a state change to 'no analysis'.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: stop analysis
Instruction: STOP
Reply: RA 0000 STOP

Name

ABRT
abort analysis

Synopsis

Instruction: `ABRT`
Reply: `<rc> ABRT`

Description

Aborts the analysis. All modules enter and leave abort synchronously i.e. the NotReady-condition "abort" is left when all modules are finished with their (private) abort actions.

Parameters

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: abort analysis
Instruction: `ABRT`
Reply: `RA 0000 ABRT`

Name

RSET
reset module

Synopsis

Instruction: RSET
Reply: <rc> RSET

Description

Resets the complete module. Only communication is stil alive. Turns off and reinitializes the hardware e.g temperature control in column compartment is turned off, starts sample transport initalization of autosamplers and fraction collector. In the DAD the spectrometer calibration information from the spectrometer's EEPROM is re-read and downloaded into the DSP. If this fails, an error event is generated. The actual wavelength Recalibration setting is not changed.

The command also aborts an analysis and timetable execution, clears the error state and restarts error checking mechanisms.

The command does not change parameters.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: reset module
Instruction: RSET
Reply: RA 0000 RSET

Name

CLER
clear error

Synopsis

Instruction: CLER
Reply: <rc> CLER

Description

Clears the error state and turns off the error LED

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: clear error
Instruction: CLER
Reply: RA 0000 CLER

Name

IDN

query identification of module

Synopsis

Instruction: IDN

Reply: <rc> IDN

Instruction: IDN?

Reply: <rc> IDN "<manuf>,<model>,<sernum>,<fwrev>"

Description

IDN identify the module by the frontend LED.

IDN? returns the identification string of the module.

Parameters

<manuf>	manufacturer: AGILENT TECHNOLOGIES revisions before Rev. A.04.00: HEWLETT-PACKARD
<model>	model number: example: G1315B (Agilent 1200 series DAD)
<sernum>	CCdddnnnnn CC=Country Code (DE=Deutschland, JP=Japan, ...) ddd=Year and week code nnnnn=running number
<fwrev>	firmware revision number (example: A.06.02)

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query module identification (G1315B)

Instruction: IDN?

Reply: RA 0000 IDN "AGILENT TECHNOLOGIES,G1315B,DE00001889,A.06.02"

Name
REV?
firmware revision query

Synopsis
Instruction: `REV?`
Reply: `<rc> REV "<fwrev>"`
Instruction: `REV? <sys>`
Reply: `<rc> REV <sys>, "<fwrev>"`

Description
Returns the firmware revision number

Parameters
<sys>	system id 0 = main, 1 = resident, 2 = boot
<fwrev>	firmware revision number

Default Values
none

Reply Codes <rc>
`RA 0000` no error, instruction has been accepted

Examples
(a) Comment: query firmware revision number of running system
Instruction: `REV?`
Reply: `RA 0000 REV "A.01.12"`

(b) Comment: query firmware revision number of boot loader
Instruction: `REV? 2`
Reply: `RA 0000 REV 2,"1.20"`

Name

BLDN?
firmware build number query

Synopsis

Instruction: `BLDN?`
Reply: `<rc> BLDN "<fwbuild>"`

Description

Returns the firmware build number

Parameters

<fwbuild> firmware build number

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query firmware build number of target system
Instruction: `BLDN?`
Reply: `RA 0000 BLDN "[017]"`

Name

SER?
queries the instrument serial number

Synopsis

Instruction: SER?
Reply: <rc> SER "<sernum>"

Description

Parameters

<sernum>	CCdddnnnnn CC=Country Code (DE=Deutschland, JP=Japan, ...) ddd=Year and week code nnnnn=running number

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query instrument serial number
Instruction: SER?
Reply: RA 0000 SER "DE00001867"

Name

TYPE

queries the instrument type

Synopsis

Instruction: `TYPE?`

Reply: `<rc> TYPE "<instrument_type>"`

Description

Parameters

<instrument_type> example: G1315B (Agilent 1200 series DAD)

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query instrument type (Agilent 1200 series DAD)

Instruction: `TYPE?`

Reply: `RA 0000 TYPE "G1315B"`

Name

MFGD?

query the manufacturing date of the module

Synopsis

Instruction: MFGD?

Reply: <rc> MFGD <value>

Description

Queries the manufacturing date that was stored in non erasible memory.

Pareters

<value> time since 1970 in seconds

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the manufactioring date
Instruction: MFGD?
Reply: RA 0000 MFGD 880622983

Name

OPT?

query option identification

Synopsis

Instruction: `OPT?`

Reply: `<rc> OPT "<moduleID>[,<optid_1>,..,<optid_N>]"`

Description

Returns a string, containing a module identification (moduleID) and one option descriptor (optid_x) for each option installed; the identification and option descriptors are separated by commas.

module ID	description	product
ISO	isocratic pump	G1310
QUAT	quaternary pump	G1311
HPGR	high pressure gradient pump	G1312
ALS	autosampler	G1313, G1327
MWD	multiple wavelength detector	G1365
DAD	diode array detector	G1314
VWD	variable wavelength detector	G1315
THERMO	column compartment	G1316
SPM	spectrophotometer	G1103

option ID	description	used by module
THERMOSTAT	autosampler with vial thermostat	ALS, G1327
SSV	solvent selection valve	PUMP, G1312
SWPMP	sealwash pump	PUMP, G1310-12
CSV	column switching valve 6/2	COL.COMP, G1316
10Port2Pos	column switching valve 10/2	COL.COMP,G1316
CONTACTS	external contacts	ALL excpt. COL.COMP
MIO	MIO Card	ALL excpt. COL.COMP
MCT7	HP89073 multicell transport (7 pos)	G1103
MCT	G1120 multicell transport (8 pos	G1103

Parameters

at least the module identification string

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query option identification of high pressure gradient pump without options
Instruction: `OPT?`
Reply: `RA 0000 OPT "HPGR"`

(b) Comment: query option identification of high pressure gradient pump with solvent selection valve and external contacts
Instruction: `OPT?`
Reply: `RA 0000 OPT "HPGR,SSV,CONTACTS"`

(c) Comment: query option identification of als
Instruction: `OPT?`
Reply: `RA 0000 OPT "ALS"`

(d) Comment: query option identification of als with vial thermostat module
Instruction: `OPT?`
Reply: `RA 0000 OPT "ALS,THERMO"`

Name

DIP?

query actual informations from DIP switch settings

Synopsis

Instruction: `DIP?`

Reply: `<rc> DIP <hpib>,<rs232>,<rsvd>,<test>`

Description

query actual informations, which have been set with the help of the DIP switches

Parameters

<hpib>	HPIB address
<rs232>	RS232 settings
<rsvd>	reserved
<test>	test settings

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:

Instruction: `DIP?`

Reply: `RA 0000 DIP 25,52,0,0`

Name

DATE
set/query date/time

Synopsis

Instruction:	`DATE <year>,<month>,<day>,<hour>,<min>,<sec>`
Reply:	`<rc> DATE <year>,<month>,<day>,<hour>,<min>,<sec>`
Instruction:	`DATE?`
Reply:	`<rc> DATE <year>,<month>,<day>,<hour>,<min>,<sec>`

Description

Set date and time.

Parameters

<year>	1970 .. 205x ???
<month>	1 .. 12
<day>	1 .. 31
<hour>	0 .. 23
<min>	0 .. 59
<sec>	0 .. 59

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: ..

Instruction:	`DATE 1997,11,30,14,50,0`
Reply:	`RA 0000 DATE 1997,11,30,14,50,0`

Name

TIME

set/query module time [in seconds since 1970]

Synopsis

Instruction:	`TIME <SecSince1970>`
Reply:	`<rc> TIME <SecSince1970>`
Instruction:	`TIME?`
Reply:	`<rc> TIME <SecSince1970>`
Instruction:	`ACT:TIME?`
Reply:	`<rc> TIME <SecSince1970>`

Description

Sets time of the module to the specified value, the format used is "Seconds since 1970"

NOTE: Every Agilent 1200 module has a realtime clock ,therefore the internal clock is still running while the module is switched off.

The queries TIME? and ACT:TIME? both return the actual time in Seconds since 1970. Do not use TIME? in the MONI:STRT command, use ACT:TIME? instead.

Parameters

`<SecSince1970>` 0 .. 2^31 -1

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 0502 out of range

Examples

(a)	Comment:	set time to 8123432
	Instruction:	`TIME 8123432`
	Reply:	`RA 0000 TIME 8123432`
(b)	Comment:	query actual time
	Instruction:	`ACT:TIME?`
	Reply:	`RA 0000 ACT:TIME 8123438`

Name

ATON

set automatic turn on

Synopsis

Instruction:	`ATON <turnon_time>`
Reply:	`<rc> ATON <turnon_time>`
Instruction:	`ATON?`
Reply:	`<rc> ATON <turnon_time>`
Instruction:	`ACT:ATON?`
Reply:	`<rc> ACT:ATON <time_left>`

Description

Sets the time in seconds since 1970 when the module will be turned on. ATON? returns the value set with ATON, ACT:ATON? returns the time left before automatic turn on.

At turn on time the following is done:

1200 DAD:	lamps are turned on
1200 PUMPS:	pump turns on
1200 TCC	TCC turns on

NOTE: Module's power must be switched on!

Parameters

<turnon_time>	0 .. Seconds Since 1970 + 26 * 10^6	; 'approx 310 days from now' a value smaller than Seconds Since 1970 deactivates the Automatic On
<time_left>	0 .. 26 * 10^6	time in seconds left before automatic turn on

Default Values

0, If not active ACT:ATON? returns 0.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0502 out of range
RA 0706 for 'started'
RA 0707 for 'deactivated', if value is smaller than Seconds Since 1970

Examples

(a)	Comment:	sets automatic turn on to 8345765
	Instruction:	`ATON 8345765`
	Reply:	`RA 0000 ATON 8345765`
(b)	Comment:	queries automatic turn on time
	Instruction:	`ATON?`
	Reply:	`RA 0000 ATON 8345765`
(c)	Comment:	returns the remaining time until automatic turn on (in seconds)
	Instruction:	`ACT:ATON?`
	Reply:	`ACT:ATON 231`

Name

ECPn

generate puls at external contact 0,1,2 or 3

Synopsis

Instruction: `ECPn <duration>`

Reply: `<rc> ECPn <duration>`

Description

ECPn causes contact n to toggle its current state for the given duration. If the instruction ECn interferes with a puls, ECn is the winner.

These instructions work with the optional external contact board. (see OPT "xx, CONTACT")

A cosy sync is send upon completion of the puls.

Parameters

<duration>	0 .. 100.0 seconds

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RA 0752 option not installed

Examples

(a) Comment: puls contact 3 for 10.5 seconds

Instruction: `ECP3 10.5`

Reply: `RA 0000 ECP3 10.5`

Name

NOUT
set or query BCD output value

Synopsis

Instruction: `NOUT <value>`
Reply: `<rc> NOUT <value>`
Instruction: `ACT:NOUT?`
Reply: `<rc> ACT:NOUT <value>`

Description

Writes a number out to the pins of the BCD Connector, depending on the BCD-configuration set with NCFG. The number is only written to the connector, if the BCD-source (see NCFG) is set to 1. The number is written either in BCD-coded or binary format depending of the BCD-format set with NCFG.
ACT:NOUT? queries the actual value written to the BCD Connector. For an autosampler, if BCD-source is set to 0, this is the current vial number.
This instruction works with the optional External Contacts / BCD board.

Parameters

<value> 0 .. 4095

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0502 out of range

Examples

(a) Comment: query the current vial number
Instruction: `ACT:NOUT?`
Reply: `RA 0000 ACT:NOUT 42`

Name

CRDY
set controller ready/not ready

Synopsis

Instruction:	`CRDY <value>`
Reply:	`<rc> CRDY <value>`
Instruction:	`CRDY?`
Reply:	`<rc> CRDY <value>`
Instruction:	`ACT:CRDY?`
Reply:	`<rc> ACT:CRDY <value>`

Description

This instruction allows to set a mark for a controller that indicates that the controller is not ready or ready for an analysis. As part of the synchronization for an analysis, all connected modules of the CAN check their CRDY flag and delay the begin of the COSY lists till this flag indicates ready. The CRDY flags are controller specific, they are automatically set to ready as soon as a controller closes the LICOP connection (or 'power fails')
Example for the usage: the chemstation software sets CRDY 0 during data analysis and report generation and clears the flag to CRDY 1 upon completion. If a sequence is started on the handheld controller, each analysis is delayed (the modules show a 'nrdy controller') until all connected chemstations have done their data processing of the previous analysis.
The query CRDY? returns the controller specific flag. The actual listing ACT:CRDY? returns the or'd flags of all connected controllers.

Parameters

<value> 0 for not ready, 1 for ready

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set controller nrdy
	Instruction:	`CRDY 0`
	Reply:	`RA 0000 CRDY 0`
(b)	Comment:	query my private crdy
	Instruction:	`CRDY?`
	Reply:	`RA 0000 CRDY 0`
(c)	Comment:	query all crdy's
	Instruction:	`ACT:CRDY?`
	Reply:	`RA 0000 ACT:CRDY 0`

Name

WCTR
wait till controllers are ready

Synopsis

Instruction: WCTR
Reply: <rc> WCTR
Instruction: STOP:WCTR
Reply: <rc> STOP:WCTR

Description

Waits till all controllers indicate readyness (see CRDY). A COSY sync is sent as soon as ready is seen.

Pareters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: wait for controllers to become ready
Instruction: WCTR
Reply: RA 0000 WCTR

(b) Comment: stop waiting for controllers
Instruction: STOP:WCTR
Reply: RA 0000 WCTR

Name

WRMR

wait till remote line shows ready

Synopsis

Instruction:	`WRMR`
Reply:	`<rc> WRMR`
Instruction:	`STOP:WRMR`
Reply:	`<rc> STOP:WRMR`

Description

Waits till the remote line shows ready. A COSY sync is sent as soon as ready is seen. This instruction is part of the default cosy list, mainly to wait for remotely connected detectors to finish their 'balances'.

Pareters

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	wait for controllers to become ready
	Instruction:	`WRMR`
	Reply:	`RA 0000 WRMR`
(b)	Comment:	stop waiting for ready
	Instruction:	`STOP:WRMR`
	Reply:	`RA 0000 STOP:WRMR`

Name

NAME

sets/queries the symbolic module name

Synopsis

Instruction:	`NAME?`
Reply:	`<rc> NAME "<name>"`
Instruction:	`NAME "<name>"`
Reply:	`<rc> NAME "<name>"`

Description

Parameters

<name>	each series of printable characters, 30 chars. max.

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

`RE 0502` out of range

Examples

(a) Comment: set module name

Instruction: `NAME "MySweetALS"`

Reply: `RA 0000 NAME "MySweetALS"`

Name

BOOT:RSET
reset code execution

Synopsis

Instruction: `BOOT:RSET`
Reply: `<rc> BOOT:RSET`
Instruction: `BOOT:RSET <delay>`
Reply: `<rc> BOOT:RSET <delay>`

Description

Force processor to restart code execution at beginning of resident/main system (10020h/30020h).

Parameters

<delay> delay time to reset in seconds

Default Values

delay = 2s

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: execute reset after a delay of 5 seconds
Instruction: `BOOT:RSET 5`
Reply: `RA 0000 BOOT:RSET 5`

Name

BOOT:XFER
transfer to alternate system

Synopsis

Instruction:	`BOOT:XFER`
Reply:	`<rc> BOOT:XFER`
Instruction:	`BOOT:XFER <delay>`
Reply:	`<rc> BOOT:XFER <delay>`
Instruction:	`BOOT:XFER <delay>,<sys>`
Reply:	`<rc> BOOT:XFER <delay>,<sys>`

Description

Force processor to continue code execution at beginning of alternate system.

Parameters

<delay>	delay time to reset in seconds
<sys>	system to jump to 0 = main system 1 = resident system 2 = boot loader

Default Values

delay = 2s
sys = 1, if main is running
sys = 0, if resident is running

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0750` xfer failed

Examples

(a) Comment: jump to resident system (when currently running in main)
Instruction: `BOOT:XFER`
Reply: `RA 0000 BOOT:XFER 2,1`

Name

BOOT:STAT?
inquire boot message

Synopsis

Instruction: `BOOT:STAT?`
Reply: `<rc> BOOT:STAT "<msg>"`

Description

Inquire reason for last re-boot.

Parameters

<msg> boot message

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: get boot message
Instruction: `BOOT:STAT?`
Reply: `RA 0000 BOOT:STAT "Forced Transfer"`

Name

`BOOT:COLD`

force cold start

Synopsis

Instruction: `BOOT:COLD`

Reply: `<rc> BOOT:COLD`

Description

Force a cold start at next time module is re-booted.

Parameters

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: force cold start

Instruction: `BOOT:COLD`

Reply: `RA 0000 BOOT:COLD`

13.3. Communication Control Commands

Name

LIST

request listing

LIST:ABRT

abort listing

LIST:FRST

reset para changed flag

Synopsis

Instruction: `LIST <"item [,item[,...]]">`
Reply: `<rc> LIST <"item [,item[,...]]">[`
Instruction: `LIST:FRST <"item">`
Reply: `<rc> LIST:FRST <"item">`

Description

With the LIST istruction, a collection of instructions can be listed. For long lists like long Timetables, this results in multiple block of transfer messages. More than one item may be specified for listing. If a listing is in progress i.e. if not all blocks of all items have left the list CU, further attempts result in RE 0100 for 'list in progress'. With LIST:ABRT the actual listing in progress can be skipped.
With each item (except METH) there is a flag associated that controlles if a controller will receive a event if a paramter is this item is changed. The principle is: a event is sent when a paramter changes and the flag is set to 'do not sent event'. With each 'LIST item' the flag is reset to 'send event with next change'. The instruction LIST:FRST item allows to reset the flag even if no listing was done.
The following table summerizes the list capabilities:

item name	parameters
HOME	actual method's home values
TT	actual method's timetable
SYS	system variables
*	same as "HOME, TT, SYS"
COSY:<name>	the cosy list named <name>
COSY:*	list all cosy lists
METH	list all stored methods

Parameters

`item` ASCII identifier of what to list (see table)

Default Values

none

Reply Codes <rc>

`RA xxxx` no error, <xxxx> = number of listed items

`RE 0100` listing active

Examples

(a) Comment: list home values and time table of current method
Instruction: `LIST "HOME,TT"`
Reply: `RA 0002 LIST "HOME,TT"`

Name

`MONI:STRT`
start monitor
`MONI:STOP`
Stop monitor
`MONI:STAT?`
query monitor state

Synopsis

Instruction:	`MONI:STRT <period>,"<actual-items>"`
Reply:	`<rc> MONI:STRT <period>,"<actual-items>"`
Instruction:	`MONI:STOP`
Reply:	`<rc> MONI:STOP`
Instruction:	`MONI:STAT?`
Reply:	`<rc> MONI:STAT <on/off>,<period>,"<actual-items>"`

Description

The MONI:STRT instructions starts the CU to periodically generate messages every <period> seconds. The message numbering starts with 0000, after the message number of '9999' the num bering restarts with '0000'.
The <actual items> are a list of instructions. Allowed are only those instructions that start with ACT: . (These are the actual queries). The list is enbedded in double quotes, the instructions must be seperated by a colon.MONI:STRT implicitely stops and restarts a running monitor with new parameters.
MONI:STOP Stops the monitor from delivering periodically messages. One last message is generated containing the identification character 'E'in the message prefix, indicating that it is the last monitor message.
The MONI:STAT? query returns the actual monitor parameters (period and items) and the actual monitor state.

Parameters

<period>	1 .. 99 period in seconds when a monitor message is generated
<actual-items>	list of actual queries separated by colon, embedded in double quotes;
<on/off>	0 : monitor is switched off 1 : monitor is switched on

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	start monitor generating messages every 2 seconds
	Instruction:	`MONI:STRT 2,"ACT:TIME?;ACT:OUT?"`
	Reply:	`RA 0000 MONI:STRT 2,"ACT:TIME?;ACT:OUT?"`
(b)	Comment:	restart monitor with different parameters
	Instruction:	`MONI:STRT 1,"ACT:STPT?;ACT:POST?"`
	Reply:	`RA 0000 MONI:STRT 1,"ACT:STPT?;ACT:POST?"`
(c)	Comment:	query actual monitor state
	Instruction:	`MONI:STAT?`
	Reply:	`RA 0000 MONI:STAT 1,"ACT:STPT?;ACT:POST?"`

Name

`COM:CONF`

set/query communication interfaces

Synopsis

Instruction:	`COM:CONF <device>,<"configuration">`
Reply:	`<rc> COM:CONF <device>,<"configuration">`
Instruction:	`COM:CONF? <device>`
Reply:	`<rc> COM:CONF <device>,<"configuration">`

Description

Parameters

<com-id>	0 .. 3
	0 : CAN 1 : MIO 2 : HPIB 3 : RS232
<"configuration">	"nodenumber" for CAN-interface
	"cardtype" for MIO
	"address:secondary" for HPIB-interface
	"baud:databits:parity[:handshake]" for RS232-interface

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query HPIB-interface

Instruction: `COM:CONF? 2`

Reply: `RA 0000 COM:CONF 2,"26:0"`

13.4. Remote Interface Commands & Queries

Name

`REMO:PULS`

generate a pulse on a remote line

Synopsis

Instruction:	`REMO:PULS <line>, <duration>`
Reply:	`<rc> REMO:PULS <line>, <duration>`

Description

Generates a pulse on the specified line of the APG remote connector. The duration is in multiples of 12.5 ms. I.g a pulse of 1 second means a duration of 80.

Parameters

<line>	2 .. 9; 2 = prepare 3 = start 4 = shutdown 5 = unimplemented 6 = power on 7 = not ready 8 = stop 9 = start request
<duration>	5 .. 30000

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: remote start puls

Instruction: `REMO:PULS 3,8`

Reply: `RA 0000 REMO:PULS 3,8`

Name

REMO:LINE
set a remote line

Synopsis

Instruction: `REMO:LINE <line>, <level>`
Reply: `<rc> REMO:LINE <line>, <level>`

Description

Sets the specified line of the APG remote connector.

Parameters

<line>	2 .. 9; 2 = prepare 3 = start 4 = shutdown 5 = unimplemented 6 = power on 7 = not ready 8 = stop 9 = start request
<level>	0,1 0 = inactive, 1 = active

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: set remote not ready
Instruction: `REMO:LINE 7,1`
Reply: `RA 0000 REMO:LINE 7,1`

Name

REMO:STAT?

read remote line

Synopsis

Instruction: REMO:STAT?

Reply: <rc> REMO:STAT <lineset>

Description

Returns the current states of all remote lines.

Parameters

<lineset>	Every binaryl digit of the integer value returned represents the status of a single line (digit = 0: false, digit = 1: true). lowest digit: prepare digit 2: start digit 3: shutdown digit 4: unimplemented digit 5: power on digit 6: not ready digit 7: stop digit 8: start request

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query actual remote line status

Instruction: REMO:STAT?

Reply: RA 0000 REMO:STAT 11

Name

`REMO:WPLS`

wait for a pulse on a remote line

Synopsis

Instruction:	`REMO:WPLS <line>, <timeout>`
Reply:	`<rc> REMO:WPLS <line>, <timeout>`

Description

Waits for a pulse on the specified line of the APG remote connector. A cosy sync is generated once the pulse has been detected, an error is generated if the pulse has not been detected till the timeout value has been counted to 0.

Parameters

<line>	2 .. 9; 2 = prepare 3 = start 4 = shutdown 5 = unimplemented 6 = power on 7 = not ready 8 = stop 9 = start request
<timeout>	0 .. 99999.00 minutes for Agilent 1200 series modules 0 .. 6000000.0 seconds for G3010A

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: wait for a remote start pulse (abort after 5 minutes)

Instruction: `REMO:WPLS 3,5`

Reply: `RA 0000 REMO:WPLS 3,5`

Name

`REMO:WLNE`

wait for a remote line

Synopsis

Instruction:	`REMO:WLNE <line>, <level>, <timeout>`
Reply:	`<rc> REMO:WLNE <line>, <level>, <timeout>`

Description

Wait for the level of the specified line of the APG remote connector. A cosy sync is sent once the appropriate level is detected. An error is generated if the timeout value has been counted back to 0 before the level has been detected.

Parameters

<line>	2 .. 9;
	2 = prepare
	3 = start
	4 = shutdown
	5 = unimplemented
	6 = power on
	7 = not ready
	8 = stop
	9 = start request
<level>	0,1 0 for active (kept to ground), 1 for inactive
<timeout>	0 .. 99999.00 minutes for Agilent 1200 series modules
	0 .. 6000000.0 seconds for G3010A

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	wait for remote ready (error after 5 minutes)
	Instruction:	`REMO:WLNE 7,1,5`
	Reply:	`RA 0000 REMO:WLNE 7,1,5`

14. Actual Values

14.1. Introduction

An obvious example of actual values are measured values like pressure or temperature. But also a variety of parameters show different values from their setpoint at an instant of time. This is pretty intuitive for actual flow and actual composition values during a gradient interpolation, but also other reasons lead to differences like ’flow reduction’ due to a steep pressure increase, or actual temperature values during cool down/heat up phases.

For permanent monitoring of actual values, a user interface/controller should use the ’Monitor CU’. See also chapter ’Communication Units’. This CU is configurable to deliver with a constant time interval settable from 1 to 99 seconds, a free definable list of actual values. But the intention is not to deliver actual values as ’time precise data’ to be stored and process later. For those measurement data the ’Rawdata CU’ should be used.

As example the following command shows a monitor CU configuration of the pump:

MONI:STRT 5,"ACT:FLOW?; ACT:PRES?; ACT:NRDY?; ACT:RIPL?"

This command instructs the CU to deliver every 5 seconds a message that containes the actual flow, the actual pressure, the actual not ready state and the actual ripple of the pressure. Note the ’;’ as seperater of multiple actual values.

Actual values can also be directly queried from the modules (as all other queries), but this should only be used for a momentum overview of the module’s state and parameters, like the chemstation does with each start of an analysis. It should not be used to do polling for monitoring reasons (which is considered the most unintelligent way of collecting data in a multi processor/multi tasking/networking environment).

14.2. Common Actual Values

Name

STAT?
ACT:STAT?
query generic module states

Synopsis

Instruction: `STAT?`
Reply: `<rc> STAT <generic>,<analysis>,<error>,<nrdy> <test>`
Instruction: `ACT:STAT?`
Reply: `<rc> ACT:STAT <generic>,<analysis>,<error>, <nrdy>,<test>`

Description

The queries STAT? and ACT:STAT? return the generic module states.
The reply to STAT? is ASCII-coded, the reply to ACT:STAT? is decimal-coded.
The module specific status is reported using a different query (e.g. ACT:PUMP? for the Agilent 1200 series pump module).

Parameters

<generic>	"PRERUN" | "RUN" |"POSTRUN" | "SHUTDOWN" | "LEAK" | "STARTUP" 0 = prerun , 1 = run, 2 = postrun, 3 = shutdown, 4 = leak, 5 = startup
<analysis>	STARTUP" | "ANALYSIS" | "NO_ANALYSIS" 0 (= no analysis) , 1 (= analysis) , 2 (= startup)
<error>	either "STARTUP" or "NO_ERROR" or "ERROR" 0 = noerror, 1 = error, 2 = startup
<nrdy>	either "STARTUP" or "NOTREADY" or "READY" 0 = ready, 1 = notready, 2 = startup
<test>	either "STARTUP" or "NO_TEST" or "TEST" 0 = notest, 1 = test, 2 = startup

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: states typical for postrun:
Instruction: `STAT?`
Reply: `RA 0000 STAT "POSTRUN","ANALYSIS","NO_ERROR","READY", "NO_TEST"`

(b) Comment: states typical for postrun
Instruction: `ACT:STAT?`
Reply: `RA 0000 ACT:STAT 2,1,0,0,0`

Name

ACT:BTMP?
Board Temperature, LeakSensor Current and Leak State

Synopsis

Instruction: `ACT:BTMP?`
Reply: `<rc> ACT:BTMP <boardtemp>,<senscurrent>,<leakstate>`

Description

ACT:BTMP? returns the internal temperature of the PC-Board,the current of the PTC sensor which detect the leak flow and the state of the leak sensor evaluation.
NOTE: Not implemented for spectrophotometer 8453 (returns all zeros)!

Parameters

`<boardtemp>`	-40.0 ... 150.0 degree Celsius
`<senscurrent>`	sensor current in mA: 3.0 ... 500 mA
`<leakstate>`	-1 during warmup phase 0 for 'no leak' 1 for 'leak detected' 2 for 'leak hardware defect: NTC Board Sensor shorted 3 for 'leak hardware defect: NTC Board Sensor open 4 for 'leak hardware defect: PTC Leak Sensor shorted 5 for 'leak hardware defect: PTC Leak Sensor open / or not connected

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual board temperature of 30.1 degC and the leak sensor current of 34.7 mA na dthe actual leak state that contents 'no leak':
Instruction: `ACT:BTMP?`
Reply: `RA 0000 ACT:BTMP 30.1,34.7,0`

Name

ACT:ATIM?

queries actual analysis time

Synopsis

Instruction: ACT:ATIM?

Reply: <rc> ACT:ATIM <analysis-time>

Description

Parameters

<analysis-time> 0 .. 99999.00[min] for Agilent 1200 series modules
0 .. 6000000.0 [sec] for G3010A
analysis-time is counted upward from begin of analysis

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual analysis-time (Agilent 1200 series modules)
Instruction: ACT:ATIM?
Reply: RA 0000 ACT:ATIM 12.62

14.3. Not Ready

14.3.1. Instruction

Name

ACT:NRDY?
Not Ready States

Synopsis

Instruction: ACT:NRDY?
Reply: <rc> ACT:NRDY <Conditions>

Description

ACT:NRDY? queries the actual condition(s) that are causing the module to show 'NotReady' (corresponding to the state event ES 0108 not ready on page 884).
The reply is 0 if the module is ready (corresponding to the state event)

Parameters

<conditions> a number, where each bit represents a not ready reason.
Bits 1 .. 8 represent the same reason with all Agilent 1200 series modules.
Starting with bit 9, the reasons are different for each module.
(see Not Ready Condition Table page 219)

Default Values

0 for ready

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Top Cover of a module is open
Instruction: ACT:NRDY?
Reply: RA 0000 ACT:NRDY 16

14.3.2. Common Not Ready Conditions

NOT READY CONDITION TABLE: Agilent 1200 Series

Bit	hex	decimal	REASON
1	0x0000001	1	Startup
2	0x0000002	2	Reset
3	0x0000004	4	Leak
4	0x0000008	8	Shutdown
5	0x0000010	16	Cover open
6	0x0000020	32	Wait Remote
7	0x0000040	64	Wait Controller (Controller Notready)
8	0x0000080	128	Aborting active

14.3.3. Not Ready (Pumps)

NOT READY CONDITION TABLE: Pump (G1310A,G1311A,G1312A,G1312B)

Bit	hex	decimal	REASON
9	0x0000100	256	Init
10	0x0000200	512	Sync piston
11	0x0000400	1024	Drive off
12	0x0000800	2048	Wait piston
13	0x0001000	4096	Wait volume
14	0x0002000	8192	Wait pressure
15	0x0004000	16384	Bottle A not ready
16	0x0008000	32768	Bottle B not ready
17	0x0010000	65536	Bottle C not ready
18	0x0020000	131072	Bottle D not ready
19	0x0040000	262144	Column Flow
20	0x0080000	524288	Purge A1
21	0x0100000	1048576	Purge A2
22	0x0200000	2097152	Purge B1
23	0x0400000	4194304	Purge B2
24	0x0800000	8388608	Purge
25	0x1000000	16777216	Comp Change
26	0x2000000	33554432	Mode
27	0x4000000	67108864	Flow Init
28	0x8000000	134217728	Elasticity Tuning
29	0x10000000	268435456	Comprss. Tuning
30	0x20000000	536870912	Flow Reduction

14.3.4. Not Ready (TCC)

NOT READY CONDITION TABLE: Column Compartment (G1316)

Bit	hex	decimal	REASON
9	0x0000100	256	Thermo off
10	0x0000200	512	Left too low
11	0x0000400	1024	Left too high
12	0x0000800	2048	Left heat up
13	0x0001000	4096	Left cool down
14	0x0002000	8192	Right too low
15	0x0004000	16384	Right too high
16	0x0008000	32768	Right heat up
17	0x0010000	65536	Right cool down
18	0x0020000	131072	Left wait temp
19	0x0040000	262144	Right wait temp
20	0x0080000	524288	Left remote temp partner missing
21	0x0100000	1048576	Right remote temp partner missing
22	0x0200000	2097152	
23	0x0400000	4194304	
24	0x0800000	8388608	
25	0x1000000	16777216	

14.3.5. Not Ready (Autosampler)

NOT READY CONDITION TABLE:
Autosampler (G1313A, G1329A, G1389A, G2260A)

Bit	hex	decimal	REASON
9	0x00000100	256	Init
10	0x00000200	512	Door
11	0x00000400	1024	Motor
12	0x00000800	2048	Error
13	0x00001000	4096	Service
14	0x00002000	8192	No tray
15	0x00004000	16384	
16	0x00008000	32768	
17	0x00010000	65536	Temp off
18	0x00020000	131072	Cooling
19	0x00040000	262144	Heating
20	0x00080000	524288	
21	0x00100000	1048576	
22	0x00200000	2097152	
23	0x00400000	4194304	
24	0x00800000	8388608	
25	0x01000000	16777216	

14.3.6. Not Ready (DAD, MWD)

NOT READY CONDITION TABLE: DAD (G1315A/B) AND MWD (G1365A/B)

Bit	hex	decimal	REASON
9	0x0000100	256	Prepare
10	0x0000200	512	Deuterium lamp ignition
11	0x0000400	1024	
12	0x0000800	2048	Deuterium lamp not ready
13	0x0001000	4096	Tungsten lamp not ready

14.3.7. Not Ready (VWD)

NOT READY CONDITION TABLE: VWD (G1314A, G1314B, G1314C)

Bit	hex	decimal	REASON
9	0x0000100	256	Prepare
10	0x0000200	512	Deuterium lamp ignition
11	0x0000400	1024	Lamp warm up
12	0x0000800	2048	Deuterium lamp not ready
13	0x0001000	4096	Tungsten lamp not ready
14	0x0002000	8192	
15	0x0004000	16384	Sample scan
16	0x0008000	32768	Blank scan
17	0x0010000	65536	Scan
18	0x0020000	131072	
19	0x0040000	262144	Calibrate
20	0x0080000	524288	Calibration lost
21	0x0100000	1048576	
22	0x0200000	2097152	Holmium check
23	0x0400000	4194304	
24	0x0800000	8388608	
25	0x1000000	16777216	

14.3.8. Not Ready (FLD)

NOT READY CONDITION TABLE: Fluorescence Detector (G1321A)

Bit	hex	decimal	REASON
9	0x0000100	256	Prepare
10	0x0000200	512	
11	0x0000400	1024	Lamp warm up
12	0x0000800	2048	UV lamp not ready
13	0x0001000	4096	
14	0x0002000	8192	Calibrating
15	0x0004000	16384	Calibration lost
16	0x0008000	32768	No cell
17	0x0010000	65536	Scan
18	0x0020000	131072	3-D Scan
19	0x0040000	262144	PMT gain test
20	0x0080000	524288	EX go rot
21	0x0100000	1048576	EX go pos
22	0x0200000	2097152	EM go rot
23	0x0400000	4194304	EM go pos
24	0x0800000	8388608	EX mot init
25	0x1000000	16777216	EM mot init
26	0x2000000	33554432	ADC overflow

14.3.9. Not Ready (RID)

NOT READY CONDITION TABLE: RI Detector (G1362A)

Bit	hex	decimal	REASON
9	0x0000100	256	reference liquid exchange time (purge)
10	0x0000200	512	temperature wait
11	0x0000400	1024	recycle time
12	0x0000800	2048	wait after automatic purge
13	0x0001000	4096	wait for RI signal
14	0x0002000	8192	light low
15	0x0004000	16384	light high
16	0x0008000	32768	simulated data
17	0x0010000	65536	unbalanced diodes
18	0x0020000	131072	
19	0x0040000	262144	
20	0x0080000	524288	
21	0x0100000	1048576	
22	0x0200000	2097152	
23	0x0400000	4194304	
24	0x0800000	8388608	
25	0x1000000	16777216	

14.3.10. Not Ready (HPLC-Chip Cube Interface)

NOT READY CONDITION TABLE: HPLC-Chip Cube Interface (G4240A)

Bit	hex	decimal	REASON
9	0x0000100	256	Load in Progress
10	0x0000200	512	Door Open
11	0x0000400	1024	Not Connected to MS
12	0x0000800	2048	Trapping in Progress
13	0x0001000	4096	No Chip Loaded
14	0x0002000	8192	Trap Pump Missing
15	0x0004000	16384	Gradient Pump Missing
16	0x0008000	32768	Unload in progress
17	0x0010000	65536	Valve Position Mismatch
18	0x0020000	131072	Wait Pressure
19	0x0040000	262144	Valve Calibration in Progress
20	0x0080000	524288	
21	0x0100000	1048576	
22	0x0200000	2097152	
23	0x0400000	4194304	
24	0x0800000	8388608	
25	0x1000000	16777216	

14.4. Start Not Ready

14.4.1. Instruction

Name

ACT:SRDY?
Reasons for start not ready.

Synopsis

Instruction: `ACT:SRDY?`
Reply: `<rc> ACT:SRDY <Conditions>`

Description

ACT:SRDY? queries the actual condition(s) that are causing the module to reject an analysis start. If and only if a start is allowed, the reply is 0.
Only if the module is already in analysis this command returns 0 although an additional STRT-command will be rejected (with "RA 0704 analysis already in progress"). Therefore the module generally stays in "start ready" if it's within analysis state.

Parameters

<conditions> a number, where each bit represents a reason for rejecting an analysis. Bits 1 .. 10 represent the same reason with all Agilent 1200 series modules. Starting with bit 11, the reasons are different for each module. (see following common start not ready conditions and all he private condition tables.)

Default Values

0 for "start is allowed"

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Module has a leak.
Instruction: `ACT:SRDY?`
Reply: `RA 0000 ACT:SRDY 4`

14.4.2. Common Start Not Ready Conditions

START NOT READY CONDITION TABLE: Agilent 1200 Series

Bit	hex	decimal	REASON
1	0x00000001	1	Startup
2	0x00000002	2	Reserved
3	0x00000004	4	Leak
4	0x00000008	8	Shutdown
5	0x00000010	16	Abort
6	0x00000020	32	Reserved
7	0x00000040	64	Reserved
8	0x00000080	128	No fan rotation

START NOT READY CONDITION TABLE: Agilent 1200 Series

Bit	hex	decimal	REASON
9	0x00000100	256	Incompatible firmware or artificial permanent error
10	0x00000200	512	Error in loading FPGA

14.4.3. Start Not Ready (Pumps)

START NOT READY CONDITION TABLE:
Pump (G1310A,G1311A,G1312A, G1312B)

Bit	hex	decimal	REASON
11	0x0000400	1024	Drive off
12	0x0000800	2048	Drive init

14.4.4. Start Not Ready (Column Compartment)

START NOT READY CONDITION TABLE:
Column Compartment (G1316A, G1316B)

Bit	hex	decimal	REASON
11	0x00000400	1024	Thermo off

14.4.5. Start Not Ready (Autosampler)

START NOT READY CONDITION TABLE:
Autosampler (G1313A, G1329A, G1389A, G2260A)

Bit	hex	decimal	REASON
11	0x00000400	1024	Initializing
12	0x00000800	2048	Motor temperature or no tray
13	0x00001000	4096	Fatal error
14	0x00002000	8192	Service mode

14.4.6. Start Not Ready (DAD, MWD)

START NOT READY CONDITION TABLE:
DAD (G1315A/B) AND MWD (G1365A/B)

Bit	hex	decimal	REASON
11	0x00000400	1024	Device Reset
12	0x00000800	2048	Prepare
13	0x00001000	4096	Scan
14	0x00002000	8192	Temperature hardware failure
15	0x00004000	16384	DSP not okay
16	0x00008000	32768	Test

14.4.7. Start Not Ready (VWD)

START NOT READY CONDITION TABLE: VWD (G1314A, G1314B, G1314C)

Bit	hex	decimal	REASON
11	0x00000400	1024	Device Reset
12	0x00000800	2048	Prepare
13	0x00001000	4096	Scan

14.4.8. Start Not Ready (FLD)

START NOT READY CONDITION TABLE: Fluorescence Detector (G1321A)

Bit	hex	decimal	REASON
11	0x00000400	1024	Prepare
12	0x00000800	2048	Calibration
13	0x00001000	4096	Scan
14	0x00002000	8192	Calc PMT
15	0x00004000	16384	No cell
16	0x00008000	32768	Missing frontend board
17	0x00010000	65536	Adc calibration failed
18	0x00020000	131072	Lamp cover open

14.4.9. Start Not Ready (RID)

START NOT READY CONDITION TABLE: RI Detector (G1362A)

Bit	hex	decimal	REASON
11	0x00000400	1024	Reference Cell Purge
12	0x00000800	2048	Lamp Defect
13	0x00001000	4096	Temperature Sensor Defect

14.4.10. Start Not Ready (HPLC-Chip Cube Interface)

START NOT READY CONDITION TABLE: HPLC-Chip Cube Interface (G4240A)

Bit	hex	decimal	REASON
11	0x00000400	1024	Missing MS
12	0x00000800	2048	Not in operate mode

15. Occupy/Lock instructions

15.1. Introduction

If more than one controller is connected to the instrument one might prevent other controllers from interacting with the instrument. This is especially interesting in regulated environment.

Exemplar usage:
At begin of a sequence of operations with the instrument the controller may lock others from manipulating the instrument. Only monitoring may be allowed. After end of operation the controller releases the lock.

Instructions may be locked using the instruction OCPY <bitfield>. Locking means that only the controller that has set the lock is accepted to use the instructions defined by the <bitfield>. All other controllers' attempts end with the reply RE 0504 <instruction> , indicating 'access denied'. If a controller tries to occupy or release instructions classes that are already locked by another controller the reply sent is RE 505 OCPY <bitfield>.
A controller may release his locks using the instruction RLSE <bitfield>.
The bits of the bitfield may be set independently from each other. For example:
OCPY 4 ====> OCPY? ====> OCPY 4
OCPY 8 ====> OCPY? ====> OCPY 12
that means that both bits are set!

15.2. Instructions to lock and release

Name

OCPY
Occupy instructions classes

Synopsis

Instruction:	`OCPY <bitfield>`
Reply:	`<rc> OCPY <bitfield>`
Instruction:	`OCPY?`
Reply:	`<rc> OCPY  <bitfield>`

Description

The instructions OCPY allows to set a lock on instruction classes. The parameter <bitfield> defines the instruction classes to be locked. The query OCPY? returns the instruction classes locked by the quering controller. see also RLSE. Each time the locked instruction classes changes the event EV 171 is sent. The parameter attached to the event represents the new locked instruction classes.

Parameters

<bitfield

Table 1:

Bitvalue	Decimal Value	Class Name
0x00000000	0	Class None
0x00000001	1	Query
0x00000002	2	Others
0x00000004	4	Parameter Homevalues
0x00000008	8	System Variables
0x00000010	16	Time tables
0x00000020	32	Method Handling Instructions
0x00000040	64	’unused’
0x00000080	128	Cosy lists
0x00000100	256	Communication settings
0x00000200	512	Boot
0x00000400	1024	Non Officials
0x00000800	2048	Starts

Table 1:

Bitvalue	Decimal Value	Class Name
0x00001000	4096	Stops
0x00002000	8192	Instrument ID
0x00004000	16384	Control
0x00008000	32768	On/Off
0x00010000	65536	File/Data
0x00020000	131072	EMF Counters
0x00040000	262144	Rawdata
0x00080000	524288	Calibration
0x00100000	1048576	Maintenance
0x00200000	2097152	Miscellaneous
0x00400000	4194304	Tags

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0505` class of instructions already occupied

Examples

(a) Comment: Lock Homevalues, System Parameters, Timetables and Cosylists (4+8+16+128)
Instruction: `OCPY 156`
Reply: `RA 0000 OCPY 156`

(b) Comment: Query locks
Instruction: `OCPY?`
Reply: `RA 0000 OCPY 156`

Name

RLSE
Release Locks

Synopsis

Instruction: `RLSE <bitfield>`
Reply: `<rc> RLSE <bitfield>`

Description

RLSE releases the instruction classes locked with the instruction OCPY.

Parameters

`<bitfield>` see instruction OPCY

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0505 class of instructions already occupied

Examples

(a) Comment: Release Locks for the system variables:
Instruction: `RLSE 8`
Reply: `RA 0000 RLSE 8`

Name

AOCP?

Queries for all locks

Synopsis

Instruction: `AOCP?>`

Reply: `<rc> AOCP <bitfield>`

Description

While the OCPY? instructions returns only those locks set by the queriyng controller, the AOCP instruction rturns all bits locked, disregarding wich controller has set the lock.

Parameters

`<bitfield>` see instruction OCPY

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: Query all locks:

Instruction: `AOCP?`

Reply: `RA 0000 AOCP 8`

Name

ROCP

Release all locks when the controller has gone

Synopsis

Instruction:	`ROCP 0/1`
Reply:	`<rc> ROCP 0/1`
Instruction:	`ROCP?`
Reply:	`<rc> ROCP 0/1`

Description

When ROCP is set to "0" all locks are released at once when the controller has gone. Set ROCP to "1" the locks are not released before end of the current analysis.

Parameters

`0/1`	release locks at once or at end of current analysis

Default Values

0	release locks at once

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: When controller has gone release locks not before end of analysis:

Instruction: `ROCP 1`

Reply: `RA 0000 ROCP 1`

15.3. Instruction Classes

Non Classified

OCPY
RLSE
ROCP

Queries

all instructions that end with a '?'.

STPT?
....
DIAG:LIST
IDN
MONI:...

Others

unused.

Parameter Homevalues

Active Instruments Setpoints before and after an Analysis

STPT	Stop Time Value
POST	Post Time Value

System Variables

Instrument setpoints that define hardware related and non-analysis behaviour.

TOUT	Time Out
NCFG	

G1313/29 ALS: THRM:SIG1, THRM:SIG2

Time Tables

AT:<instr> <time>[,<param>...]

Method Handling instructions

Instructions to store and activate setpoints

FILE:...

COSY lists

Instructions to set up and maintain COSY lists, all instructions with the Node COSY

COSY:.....

Communication settings

CONF
PRN:...

Boot

Instructions to switch to the 'resident' system and back.

BOOT:XFER

LOAD:....

Non Officials

Instructions used during production, for special applications ....

Starts

Instruction to start an analysis
STRT
STRS

Stops

Instructions to stop/abort/reset

STOP
ABRT
RSET
STPR
STPP
STOP:WRMR
STOP:WCTR
STOP:TOUT
G1313/29 ALS: TRAY:RSET, SUSP, RESU

Instrument ID

IDN
TYPE
SER
Ri Detector: OSER
G1313/29 ALS: THRM:SER, THRM:TYPE, THRM:REW,

Control

ECP0 .. ECP3
CRDY
RUNL
RUNG
RPST
Ri Detector: VREF, VREC, APRG, AZOT, ATE, SSEQWTMP, WRI
G1310/11/12 Pumps: WVOL, WPRS,
G1313/29 ALS: TRAY:HOME, TRAY:PARK, TRAY:MOVV, TRAY:MOVE, TRAY:MOVR, TRAY:PICK, TRAY:PUT, TRAY:RELV, DRAW, EJECT, VLVE, NDLE, WASH, SRNG, INJT, SMIX, MOVV, NDLE:DOWN, NDLE:UP, MONA:NDLE, MONA:TRQ,

On/Off

Pump: PUMP
Detectors: LAMP
Column Comp: THRM
RI Detector: HEAT
G1313/29 ALS: OINJ

File/Data

File Data

DATA

DATA:..

EMF Counters

DIAG:CLMT
DIAG:CRST

Rawdata

RAWD:...

Calibration

RI Detector: NRIU
Column Comp:
G1313/29 ALS: TRAY:SET, TRAY:TWST, TRAY:ALIG, TRAY:CORR:RSET, TRAY:CORR:SET, TRAY:CORR:USE, THRM:TCAL

Maintenance

DIAG:MTNC
DIAG:EXB
G1313/29 ALS: CPON, CPOF, CNON, CNOF, CGON, CGOF

Miscelanious

DATE
TIME

Tag

16. State Diagrams

16.1. Common State Diagrams

Name

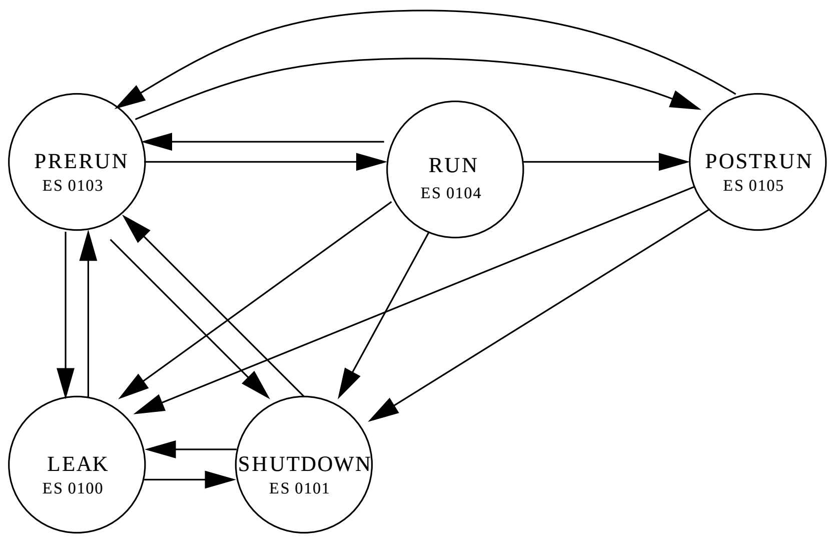
RuntimeSTM

Statemachine of Runtime Behaviour

Description

This statemachines build up the states during a chromatographic run.

State Diagram



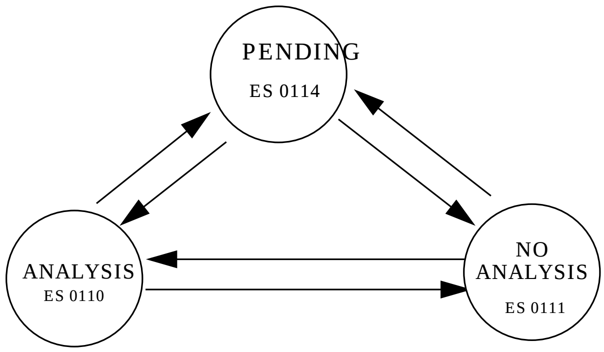
Name

AnalysisSTM
Statemachine of Analysis States

Description

This statemachines build up the analysis states.

State Diagram



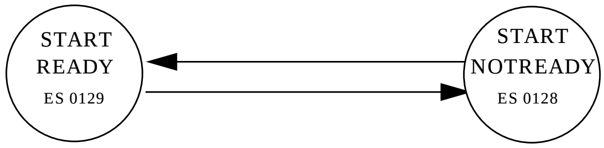
Name

StartReadySTM
Statemachine of Ready-For-Start Behaviour

Description

This statemachine builds up the ready-for-start states.

State Diagram



Name

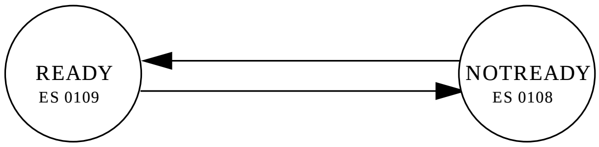
ReadySTM

Statemachine of Ready Behaviour

Description

This statemachines build up the ready states.

State Diagram



Name

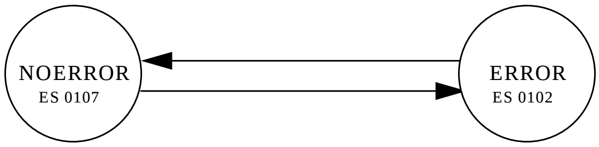
ErrorSTM

Statemachine of Error Behaviour

Description

This statemachines build up the error states.

State Diagram



Name

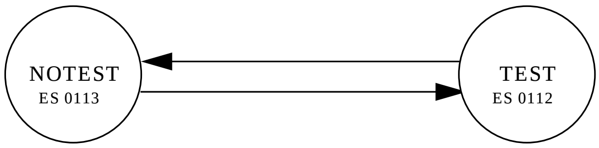
TestSTM

Statemachine of Test Behaviour

Description

This statemachines build up the test states.

State Diagram



17. Rawdata File

17.1. Introduction

The Agilent 1200 Series Rawdata file contains measurement data, which usually are generated during an analysis. The data can be a equidistant, 1-dimensional data points (called signal) or a set of array data, which also contains paramter data. Examples for signal data are detector signals or temperature over time; a spectrum measured by a detector is an example for a set of array data.

A rawdata file is generated automatically for every run. In addition, a rawdata file can be generated on request (see instruction RAWD:STRT).

The records of the rawdtata file are stored within the module, until they are read by the controlling device via the rawdata CU. After a record is read, its memory is freed (recycled) and can later be (re-)used for storing new data.

Every module can hold only one rawdata file at a time. Starting a new run or resetting the rawdata file will cause the loss of almost all the data stored within the module (data already prepared for transmission are not affected). Nevertheless, necessary stop records are generated to guarantee correspondig pairs of start/stop records.

Rawdata are NOT powerfail protected! Data already saved within the module are lost at power fail or power off.

The amount of records, that can be stored within the Agilent 1200 series module depends on the module. If all records are in use, the rawdata file overflows. This will cause loss of data, and an event is generated, which describes the type of loss. Only one overflow event for a given type is generated per run.

Amongst the different types of data, that can be stored in the rawdata file, the signal data do have priority. This will cause other data records to be discarded in case of an overflow. In this case, always the latest data records of a specific type will be discarded.
Because the signal data in the rawdata file are organized as equidistant data points without any time stamp, a loss of signal data due to a rawdata file overflow will cause the rawdata file generation to stop. Rawdata stored up to this time are still available.

17.2. Related Instructions

RAWS <signalset>
: select the chromatographic signals to be stored (page 258)

RAWF <format>, <number_of_datapoints>
: select rawdata file format (page 259)

RAWD:STRT
: start storing rawdata (monitor) (page 254)

RAWD:STOP
: stop storing rawdata (page 255)

RAWD:STAT?
: query rawdata file status (page 257)

RAWD:RSET
: reset rawdata file and stop storing signal data (page 256)

17.3. Rawdata File Commands

Name

RAWD:STRT
start storing rawdata (monitor)

Synopsis

Instruction: RAWD:STRT
Reply: <rc> RAWD:STRT

Description

Starts storing the selected signal data into the internal rawdata file to be used for monitoring purposes (outside an analytical run). Not allowed during a run.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0305 not allowed during run

Examples

(a) Comment: start storing rawdata
Instruction: STRT
Reply: RA 0000 STRT

Name

RAWD:STOP
stop storing rawdata

Synopsis

Instruction: RAWD:STOP
Reply: <rc> RAWD:STOP

Description

Stops storing rawdata into the internal rawdata file. Does not stop an analytical run! Storing signal data is also stopped at end of run.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: start storing rawdata
Instruction: RAWD:STOP
Reply: RA 0000 RAWD:STOP

Name

RAWD:RSET
reset rawdata file and stop storing signal data

Synopsis

Instruction: RAWD:RSET
Reply: <rc> RAWD:RSET

Description

Resets the internal rawdata file, stored data are discarded. Automatically stops storing rawdata if in monitor mode. Not allowed during run.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0305 not allowed during run

Examples

(a) Comment: resets internal rawdata file
Instruction: RAWD:RSET
Reply: RA 0000 RAWD:RSET

Name

RAWD:STAT?
query rawdata file status

Synopsis

Instruction:	`RAWD:STAT?`
Reply:	`<rc> RAWD:STAT <state>, <free>, <used>`

Description

Returns the status of the internal rawdata file.

Parameters

`state`	0 means IDLE 1: MONITOR 2: MONITOR OVERFLOW 3: RUN 4: RUN OVERFLOW 5: WAIT
free, used	Number of data points free (used). The sum of <free> plus< used> is the total capacity of data points the internal rawdata file can hold.

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query actual rawdata status
Instruction: `RAWD:STAT?`
Reply: `RA 0000 RAWD:STAT 2,17508,2543`

17.4. Rawdata File Parameter

Name

RAWS
select the chromatographic signals to be stored

Synopsis

Instruction:	`RAWS <signalset>`
Reply:	`<rc> RAWS <signalset>`
Instruction:	`RAWS?`
Reply:	`<rc> RAWS <signalset>`

Description

The RAWS setting selects the signals to be stored. Although the RAWS setting is always accepted, it becomes active only at the time when the rawdata storing begins. Storing signals starts when the RAWD:STRT instruction is accepted or when an analytical run starts.
The RAWS setting is not affected when changing the method.
The number of signals available and the definition of the different signals depends on the different Agilent 1200 Series modules. See corresponding description, e.g. "VWD Rawdata File Description (page 640)".

Parameters

`<signalset>` 0 .. N
(N depends on the number of signals avialable, module dependent)

1 (bit 0 set) means: store signal A
2 (bit 1 set) means: store signal B
4 (bit 2 set) means: store signal C
...
any combination of bits is allowed,
0 means: no signal will be stored (OFF)
31 means: five signals (A,B,C,D,E) will be stored.

Default Values

1 (only signal A is stored)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0502` out of range

Examples

(a) Comment: select signal A and B to be stored
Instruction: `RAWS 3`
Reply: `RA 0000 RAWS 3`

(b) Comment: query signal select setting
Instruction: `RAWS?`
Reply: `RA 0000 RAWS 3`

Name

RAWF
select rawdata file format

Synopsis

Instruction:	`RAWF <format>, <number_of_datapoints>`
Reply:	`<rc> RAWF <format>, <number_of_datapoints>`
Instruction:	`RAWF?`
Reply:	`<rc> RAWF <format>, <number_of_datapoints>`

Description

The signal data output (digital) of the rawdata file is organized as a sequential data file, separated into multiple consecutive records. The data format of these records and the number of data points whithin each record can be selected by the RAWF settings.

The setting can be changed while reading the rawdata file. Athough a new setting is effective immediately, there might be one or two records already prepared using the previous format setting. The RAWF setting is not affected when changing the method.

NOTE: The RAWF setting also affects the format of the array data output of the rawdata file (e.g. spectrum records). Nevertheless, the record size of array data records cannot be changed.

Parameters

`<format>`
- 0 means binary format
- 1 means hexadecimal code ASCII format
- 2 means decimal coded ASCII format

`<number_of_datapoints>`
- range 1 .. 240 for binary format
- range 1 .. 120 for hexadecimal format
- range 1 .. 80 for decimal format

Default Values

format: 1 (hexadecimal coded ASCII)
number_of_datapoints: 120

Reply Codes <rc>

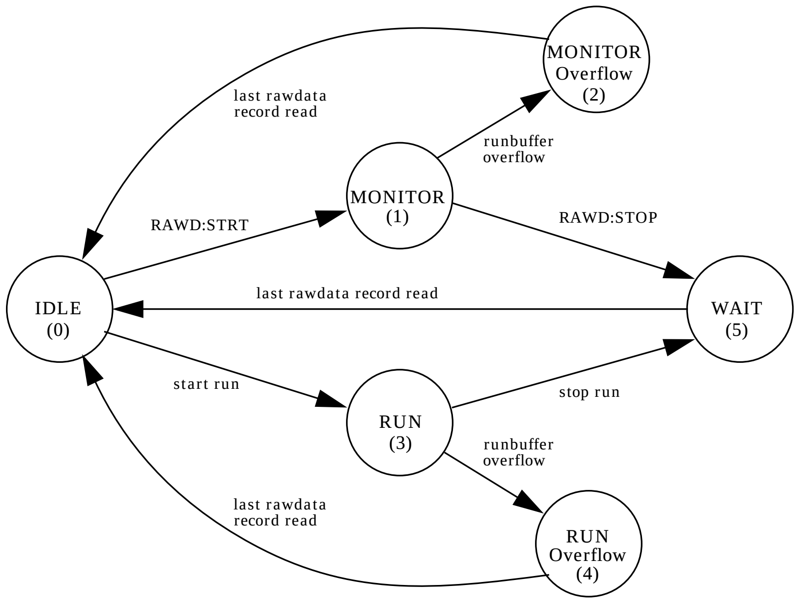
`RA 0000` no error, instruction has been accepted
`RE 0502` out of range

Examples

(a) Comment: select decimal format, 10 datapoints per record
Instruction: `RAWF 2,10`
Reply: `RA 0000 RAWF 2,10`

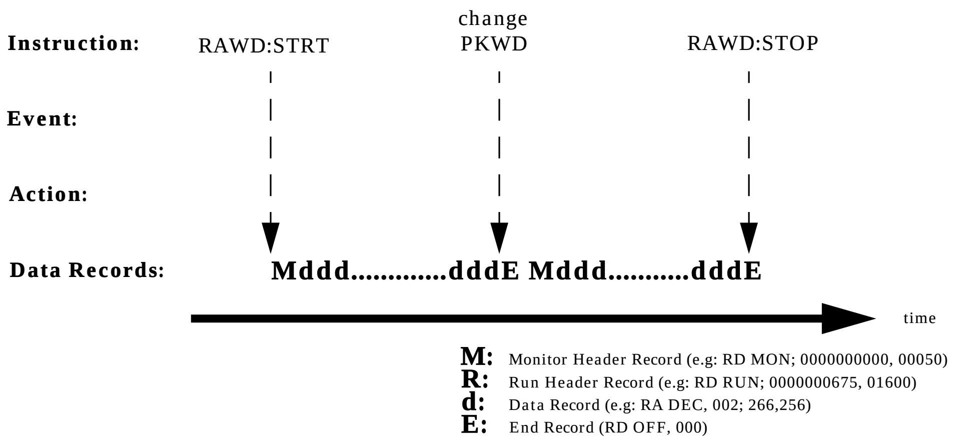
(b) Comment: query actual format
Instruction: `RAWF?`
Reply: `RA 0000 RAWF 2,10`

17.5. Rawdata File State Diagram

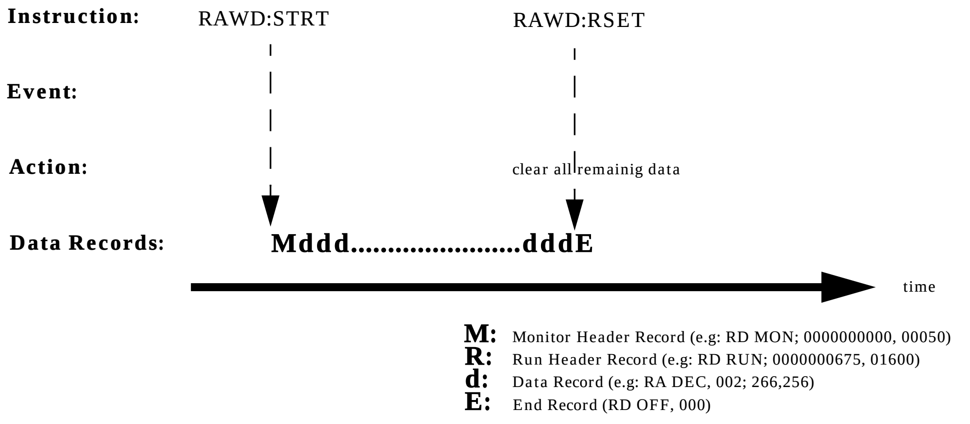


17.6. Starting and Stopping Rawdata Generation

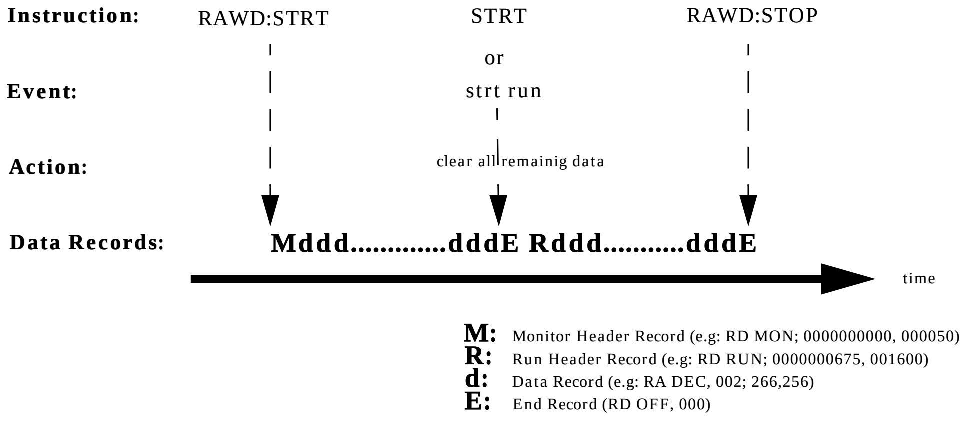
Peakwidth change during monitor datat generation



Resetting the rawdatafile generation



Issue 'start run' during monitor data generation



17.7. Rawdata Record Format

The Agilent 1200 Series rawdata file is organized as a file, always starting wiht a HEADER record, followed by zero or more DATA records and terminated by an END record.
The HEADER record contains information about the type (RUN data, MONitor data or SIMulation data), the time of the first signal data point relative to the start of the run (offset time) and the time between two adjacent signal data points.
The representation of the data in the DATA records can be binary (BIN), hexadecimal ASCII (HEX) or decimal (DEC). See also RAWF instruction.
The type of the data, which will go into the rawdata file are different for the different Agilent 1200 Series modules. Selection of these data is controlled by separate settings, see e.g. the RAWS instruction for signal data.

<**file**> ::= <headerrecord> { <data record> } * <stop record>

<**header record**> ::= RD <header id>; <relative time>, <interval>
(example: RD RUN; 0000000142, 003200
RD MON; 0000000000, 003200)

<header id> ::= RUN | MON | SIM

<relative itme> ::= time of first signal data point in this file, integer value of 10 decimals, unit: 1 ms (NOTE: time of first data point can be negative!)

<interval> ::= time between two samples, integer value, repreented by 6 decimals, unit: 0.1 ms

<**data record**> ::= <signal data record> | <array data record>

<**stop record**> ::= RD OFF, 0000;

<signal data record> ::= <signalheader> {<integer>}*

examples:
RA BIN,0240;<d><d>...
RB HEX,0002;0000000A0000000B
RC DEC,0005; -30,1,20,237,1282

<signalheader> ::= R<signal id> <format>, <intcount>;
(12 bytes fix: e.g "`RA BIN,0240;`")

<signal id> ::= A | B | C | ...

<format> ::= BIN | HEX | DEC
(defines the format of the data following the signalheader or the arrayheader: <binary int>, <hex int> or <decimal int>)

<intcount> ::= <decimal int>
(number of data points in this record, 4 bytes fix)

<integer> ::= <binary int> | <hex int> | <decimal int>

<binary int> ::= (4 byte signed integer, MOTOROLA format)

<hex int> ::= (8 byte hexadecimal ascii signed integer, MOTOROLA format)

<decimal int> ::= (decimal signed integer separated by comma, 10 bytes max, optional minus sign)

<**array data record**> ::= <arrayheader><numpara><para><numdata><arraydata>

```
examples for a whole spectrum:
SL BIN,0240;<d><d>...
SL BIN,1043;<d><d>...
SL HEX,0523;00000015328C9A6B000020D5000012DE...
SL DEC,0080;-353,34,234,-423,23,234,-42,...

example of a spectrum in 3 parts:
SL0DEC,0348;21,848075283,7605,4030,200,190000,...
SL1DEC,0348;73350,213105,-233435,-42419,145665,...
SLEDEC,0037;258782,-27054,28516,169113,-53848,...
```

<arrayheader> ::= <id><recordpart> <format>,<intcount>;
(12 bytes fix: e.g "`SL BIN,0240;`")

<numpara> ::= <integer>
(number of parameter following)

<para> ::= {<integer>}*

<numdata> ::= <integer>
(number of data points following)

<arraydata> ::= {<integer>}*

<id> ::= (two ASCII characters, specifying the type of data; module specific)

```
examples:
SV: variance spectrum
SL: normal spectrum
IL: normal intensity scan
```

<recordpart> ::= <(blank> | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | E
(<blank>: whole record
0: first part of an array data record
1: second part of an array data record
..
E: last part of an array data record.)

18. Diagnostics

18.1. General Description

An Agilent 1200 Series module holds a variety of data in its internal memory. This chapter describes the data and access mechanism for diagnostic purposes.

18.1.1. Principles

Diagnostic Data is always available, it cannot be cleared or removed (the history principle). Each type of data is collected in an individual buffer that has a name of up to 4 characters. Each buffer has a maximum number of entries. Once full the following entry overwrites the oldest. The time frame covered by a buffer depends on the frequency of the entries, it may range from a few milli seconds up to years.
Error events for excample are collected in the buffer named ERR. The maximum number of entries is 40, this means the last 40 error events are always available, each entry of an error event is accompanied by a time stamp when the event was generated, and by a parameter that may help to further decipher what caused the error event.
Two different types of diagnostic information are available:
  Diagnostic Buffers: hold historical information as explained above
  Diagnostic Counters: entries are summed up. See sub chapter below.

18.1.2. How to read buffers, message headers

The listing of the contents of a buffer is initiated with the instruction DIAG:LIST <name[, name]>.If the reply is acknowledged (RA ...), buffers of the name list have been found and are provided at the diagnosis CU (see also the description of the CUs). The messages generated by the CU, always start with a header :
DI nnnn name, TimeOfList, TimeOfLastEntry, XSpace, XAxisStart, XAxisStartValid, Info

DI is the CU identifier
nnnn is the number of the message for the currently listed buffer. The number is 0 is the buffer is empty, the header of the last message for the buffer starts with **DIXnnnn**.
TimeOfList is the time in seconds since 1970 when the listing was generated.
TimeOfLastEntry is the time in seconds since 1970 when the last entry to the buffer was done.
XSpace may contain a number if the buffer holds equidistant data. Same is for **XAxisStart** and **XAxisStartValid**.
Info may contain some short explanation for the buffer, by default it is 'NoInfo' (without quotes).

18.1.3. Storage types

Three different storage types are used for diagnostic data, depending on the type of information to be held:

volatile (RAM)
: used to store data only needed while the module is running; data are lost after a power cycle of the module.

non-volatile (battery-buffered RAM)
: used to store data that needs to be available after a power cycle of the module; data are lost at a coldstart of the intrument

non-erasable (Flash ROM)
: used to store data needed over the lifetiem of the module

18.1.4. Get a list of all buffers

To get a list of all buffers use DIAG:LIST "LS" as instruction, then read the buffers provided at the DIagnosis CU.
To get a list of all counters use DIAG:LIST "CNT" as instruction, then read the buffer provided at the DIagnosis CU.

18.1.5. External buffers

Normally buffers are 'read only' from the controller's point of view. Some buffers (EXBn,MTNC) can be read and written by the controller e.g. for maintenance purposes. Specific commands (DIAG:EXBn , DIAG:MTNC) can be used to write data to these buffers.

18.1.6. Counters

Counters have names like Buffers. The listing is not done via the Diagnosis CU (except when listed all together: see buffer descriotion for CNT) but via the instruction **ACT:CNT? <name>**. The reply is **ACT:CNT <name>, <total>, <erase>, <limit>**.

As the reply shows a counter consists of three values:

<total> is the part of the counter that cannot be re- or pre-set.

<erase> is the part of the counter that can be preset with the instruction **DIAG:CRST <name>, <value>**. This part is also used for limit comparison. If the <erase>value is larger than the limit an event is generated (see counter descrioptions). <total> and <erase> are updated synchronously.

<limit> can be set with the instruction **DIAG:CLMT <name>, <limit>**. If the storage type of the counter is none-volatile or non-erasable, the limit detection is still active after power cycling (warm start).A limit value of 0 deactivates the limit comparison. The Event is used twice: first if the limit is activated, the event is sent with parameter 0; secondly the event is sent with parameter 1 if the erase value is greater than the limit.

Some counters have an associated buffer that holds the history of when the counter was reset/preset. This history is a normal diagnosis buffer with the same name as the counter. See the description of each buffer to get the information if or if not a history is available.

18.2. Diagnostic Commands

Name

`DIAG:LIST`
initiate the listing of buffers

Synopsis

Instruction: `DIAG:LIST <bufnamelist>`
Reply: `<rc> DIAG:LIST <bufnamelist>`

Description

Causes the module to invoke a listing of the contents the buffers named in <bufnamelist>. The buffers are available at the Diagnosis CU. See also Communication documentation.

Parameters

<bufnamenlist> at least one buffer name, multiple buffer must be seperated by comma. The complete list has to be embedded in double quotes.

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
if the reply is rejected: no buffers are listed!

Examples

(a) Comment: List the buffer named ERR
Instruction: `DIAG:LIST "ERR"`
Reply: `RA 0001 DIAG:LIST "ERR"`

(b) Comment: List the buffers named ERR and STT
Instruction: `DIAG:LIST "ERR, STT"`
Reply: `RA 0002 DIAG:LIST "ERR, STT"`

Name

`DIAG:MTNC`
enter string to MTNC buffer

Synopsis

Instruction: `DIAG:MTNC <any text>`
Reply: `<rc> DIAG:MTNC <any text>`

Description

Enters <any text> to the buffer called MTNC.

Parameters

<any text> The maximum number of characters allowed are 20. (Truncated to 20).

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: Enter 'hello world' to the MTNC buffer
Instruction: `DIAG:MTNC "hello world"`
Reply: `RA 0000 DIAG:MTNC "hello world"`

Name

DIAG:EXBn

enter string to EXBn buffer

Synopsis

Instruction:	`DIAG:EXB0 <any text>`
Reply:	`<rc> DIAG:EXB0 <any text>`
Instruction:	`DIAG:EXB1 <any text>`
Reply:	`<rc> DIAG:EXB1 <any text>`
Instruction:	`DIAG:EXB2 <any text>`
Reply:	`<rc> DIAG:EXB2 <any text>`
Instruction:	`DIAG:EXB3 <any text>`
Reply:	`<rc> DIAG:EXB3 <any text>`

Description

Enters <any text> to the buffer called EXB0, EXB1, EXB2 or EXB3.

Parameters

<any text>	The maximum number of characters allowed are 20. (Truncated to 20).

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Enter 'hello world' to the EXB2 buffer
	Instruction:	`DIAG:EXB2 "hello world"`
	Reply:	`RA 0000 DIAG:EXB2 "hello world"`

Name

DIAG:CRST
preset counter

Synopsis

Instruction: `DIAG:CRST <counter name>, <preset value>`
Reply: `<rc> DIAG:CRST <counter name>, <preset value>`

Description

Sets the erasable part of a counter to the <preset value>. This command also caused an entry to the counter history. Entered is the total counter value, the new erasable value and the limit.

Parameters

<counter name>	max 4 characters embedded in double quotes.
<preset value>	0 .. Uint

Default Values

<preset value> is cleared to 0 with each cold start.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Set Erasable part of OnTimeCounter to 0
Instruction: `DIAG:CRST "TONT", 0`
Reply: `RA 0000 DIAG:CRST "TONT", 0`

Name

DIAG:CLMT
enter limit for a counter

Synopsis

Instruction:	`DIAG:CLMT <counter name>, <limit>`
Reply:	`<rc> DIAG:CLMT <counter name>, <limit>`
Instruction:	`DIAG:CLMT? <counter name>`
Reply:	`<rc> DIAG:CLMT <counter name>, <limit>`

Description

Sets the limit value to <limit>. Once per second the limit is check with the erasable value of the counter. If the limit is smaller then the counter's event is generated and the limit function is disabled. See the individual counter description for the event numbers.

Parameters

<counter name>	up to 4 characters embedded in double quotes.
<limit>	0 .. Uint; if 0 the event generation is disabled.

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Set Event limit of the OnTimeCounter to 60 seconds
Instruction: `DIAG:CLMT "TONT", 60`
Reply: `RA 0000 DIAG:CLMT "TONT", 60`

Name

`DIAG:ABRT`
aborts current listing of buffers

Synopsis

Instruction: `DIAG:ABRT`
Reply: `<rc> DIAG:ABRT`

Description

Parameters

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:
Instruction: `DIAG:ABRT`
Reply: `RA 0000 DIAG:ABRT`

Name

DIAG:TERM
defines termination character

Synopsis

Instruction: `DIAG:TERM <"termination">`
Reply: `<rc> DIAG:TERM <"termination">`

Description

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:
Instruction: `DIAG:TERM ","`
Reply: `RA 0000 DIAG:TERM ","`

Name

ACT:CNT?

query the actual counter values

Synopsis

Instruction: `ACT:CNT? <counter name>`

Reply: `<rc> ACT:CNT <counter name>, <tot>, <era>, <limit>`

Description

queries for the actual counter values.

Parameters

<counter name>	up to 4 characters embedded in double quotes.
<tot>	the total (none resettable) part of the counter
<era>	the erasable part of the counter
<limit>	the value for checking with <era> for event generation

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the OnTimeCounter

Instruction: `ACT:CNT? "TONT"`

Reply: `RA 0000 ACT:CNT "TONT", 1533, 324, 60`

18.3. Common Diagnose Buffers

Name

`LS (diag)`
buffer summary

Characteristics

max entries	as much as there are different buffers
max characters	20
storage	non erasible
availability	always

Contents

name, time stamp, max entries

name,	the name of a diagnostic buffer
time stamp,	0 for No time stamp, 1 for time stamp with each entry to the buffer
max entries	the maximun number of entries the buffer can hold. A buffer holds less or maximum this number of entries. Once a buffer is full, the actual number of entries equals the max number of entries. New entries lead to loosing the oldest.

Description

The LS buffer shows a list of all available buffers of the module. This buffer is never empty.

Name

`CNT (diag)`
counter summary

Characteristics

max entries	one for each counter
max characters	38
storage	volatile but always actual
availability	always

Contents

counter name, total counts, erasible counts, limit

counter name	up to four characters for the counter identification
total counts	none erasible counter value at the list time
erasible counts	erasible value of the counter at list time
limit	settible value to be used for comparison with the erasible counter value.

Description

See chapter Counters for a description of each counter. See Instruction DIAG:CLMT for setting and quering limits and see ACT:CNT? <name> for sinlge counter query. The CNT buffer lists all counters with their actual values and limits.

Name

ERR (diag)
error logbook

Characteristics

max entries	20
max characters	35
storage	non volatile
availability	always

Contents

time, event number, parameter

time,	seconds since 1970, the time when the event occured
event number,	error event number, see document *Error Events* for more detail
parameter	optional parameter to go with the event. An example is the vial number with the event 'missing vial'.

Description

An error event is generated each time a device recognizes a hardware failure (like sonsor signal missing) or a condition that is considered to disturb and analysis (like pressure exeeds limit). Each of the events is send to the event CU to allow all controllers to react upon. In paralel the events are kept in diagnostic buffers to preserve their occurence for diagnostic purposes. Different types of events are kept in different buffers, see also STT, NFO and LMT. Additionly the events are copied to the analysis logbooks AERR, ASTT, ANFO and ALMT if they occur during an analysis. The purpose their is to relate the occurence to an analysis as GLP feature.

Name

STT (diag)
state change history

Characteristics

max entries	20
max characters	35
storage	none volatile
availability	always

Contents

time, event number, parameter

time,	seconds since 1970, the time when the event occured
event number,	statechange event number, see document *Statechange Events* for more detail
parameter	optional parameter to go with the event.

Description

Each state a device can be in has a uniqeu id (the statechange event number). Each time a state is entered the apropriate events is generated. The diagnostic purpose of the buffer is to consider the sequence of statechanges when trying to find the reason for an unexpected result.

Name

NFO (diag)

info event history

Characteristics

max entries	10
max characters	35
storage	non volatile
availability	always

Contents

time, event number, parameter

time,	seconds since 1970, the time when the event occured
event number,	info event number, see document *Info Events* for more detail
parameter	optional parameter to go with the event.

Description

Info events are those events that do neither denote an error nor a statechange. They notify asynchronous occurences that are considered worth for a chemstation and/or user to be informed.

Name

LMT (diag)

Limit Event History

Characteristics

max entries	10
max characters	35
storage	non volatile
availability	always

Contents

time, event number, parameter

time,	seconds since 1970, the time when the event occured
event number,	limit event number, see document *Info Events* for more detail
parameter	optional parameter to go with the event.

Description

Limit events are those events that result from a comparison of a diagnosis counters' limit with the diagnosis counters' erasible part. Typical usage is 'early maintenace feedback'.

Name

AERR (diag)
error logbook of last analysis

Characteristics

max entries	5
max characters	35
storage	volatile
availability	always

Contents

time, event number, parameter

time,	seconds since the analysis was started
event number,	error event number, see document *Error Events* for more detail
parameter	optional parameter to go with the event. An example is the vial number with the event 'missing vial'.

Description

See also description of ERR. With each start of an analysis this buffer restarts from empty. After the analysis is done, no more entries are done. Typically an error causes the analysis to abort, therefore only few entries are provided.

Name

ASTT (diag)
state change history of last analysis

Characteristics

max entries	20
max characters	35
storage	volatile
availability	always

Contents

time, event number, parameter

time,	seconds since the analysis was started
event number,	statechange event number, see document *State Change Events* for more detail
parameter	optional parameter to go with the event.

Description

With each start of an analysis this buffer is refilled from empty. After the analysis is done no more entries are done. A typical analysis has only 5 to 10 state changes, but the larger a special application the larger the amount of state changes.

Name

`ANFO (diag)`

messages of last analysis

Characteristics

max entries	5
max characters	35
storage	volatile
availability	always

Contents

time, event number, parameter

time,	seconds since the analysis was started
event number,	info event number, see document *Info Events* for more detail
parameter	optional parameter to go with the event.

Description

With each start of an analysis, this buffer is refilled from empty. After the end of the analysis no more entries are done.

Name

`ALMT (diag)`

Limit events during last analysis

Characteristics

max entries	5
max characters	35
storage	volatile
availability	always

Contents

time, event number, parameter

time,	seconds since the analysis was started
event number,	limit event number, see document *Info Events* for more detail
parameter	optional parameter to go with the event.

Description

With each start of an analysis, this buffer is refilled from empty. After the end of the analysis no more entries are done.

Name

ACHG (diag)

parameters changed during the analysis

Characteristics

max entries	10
max characters	20
storage	volatile
availability	always

Contents

time, controller id, instruction

time	seconds since the analysis was started
controller id	identification number (LICOP) of the controller that send the parameter
instruction	any instruction of class 'parameter', 'sysvar' or 'time table' that was sent to the device during the analysis. Only the first 20 characters of the instruction are kept.

Description

With each start of an analysis, this buffer is refilled from empty. After the end of the analysis no more entries are done. The purpos is to explain deviations from an automated predefined progress of the analysis. After an analysis is started a chemstation has no need at all to send instructions to the device (except for abort/stop/reset/queries.. wich is quite instresting for diagnostics). All data exchange during an analysis is handled via dedicated communication channels like monitor CU, event CU and rawdata CU.

Name

INST (diag)

instruction history

Characteristics

max entries	20
max characters	32
storage	non volatile
availability	allways

Contents

controller id, instruction

controller id	identification number (LICOP) of the controller that send the instruction
instruction	any instruction sent to the in CU. Only 32 character of an instruction are kept.

Description

See also REPL for the reply error to the instruction

Name

REPL (diag)
reply error history

Characteristics

max entries	5
max characters	30
storage	non volatile
availability	allways

Contents

time, reply

time,	seconds since 1970, the time when the reply was available at the incu
reply	the reply error to an instruction. Only the first 30 characters of a reply string are kept.

Description

See also INST for the instruction history. Reply errors are generated when the parsing routine for an instruction detects a syntax error, a parameter range violation.... A reply error allways means that the instruction was rejected and had no influence neither on a parameter nor on an execution of what ever.

Name

DCHG (diag)
date change history

Characteristics

max entries	10
max characters	22
storage	non erasible
availability	always

Contents

previous second, next second

previous second	the last second of the previous date
next second	the first second of the new date

Description

The last 10 date changes can be recalculated (not unambigously) from the two entries. An entry is only done if the time values of previous and new date differ by more than 10 seconds. Smaller differences are assumed to be corrections rather than changes (corrections of the date are done with each start of an analysis to synchronize the date of all CAN wired devices, and are done by controllers if a difference from devices to PC clocks are found!)

Name

PFL (diag)
short powerfail history

Characteristics

max entries	15
max characters	12
storage	volatile
availability	always

Contents

time

time	seconds since 1970

Description

Short powerfailures (<200 ms) are detected and the time of the occurence is kept. The buffer is empty after turn on.

Name

TON (diag)
turn on/off history

Characteristics

max entries	10
max characters	30
storage	non volatile
availability	always

Contents

time, on/off, param

time — seconds since 1970, the time when power was turned off or on, see **on/off**.

on/off —
0 for power off,
1 for power on,
2 for power on without power off detection,
3 for power off without information (because of coldstart power on next time)

param — not for external use
case **on/off** is off (0,3), the 'ResetStatusRegister' with the actual 'on reason' was reported:
bit7: External, bit6: Power-Up, bit5: Software Watchdog, bit4: Halt Monitor
bit3: Reserved, bit2: Loss of Clock, bit1: System, bit0: Test Submodule
case **on/off** is on (1,2), the 'GetSysContext' with the actual on context was reported: **param** defines what type of start up was used:
if the number is odd, a *cold start* was performed, that means that all method data are set to their default values (also most diagnostic buffers are empty). If the number is even a *warm start* was done, all method data, and most diagnostic buffers are still available; see storage types.
Context: 0xX000: IRQ6, MUSIC2, TABLE, FLASH
0x0X00: SIMUL, DEBUG, EMUL, TARGET
0x000X: unused, HOT, WARM, COLD

Description

Keeps time and reason when the device was turned off and when it was turned on.

Name

MSG (diag)
turn off message

Characteristics

max entries	5
max characters	32
storage	non volatile
availability	debug only

Contents

time, message

oldsecond	seconds since 1970, the time when the system was booted
message	the message that was generated during turn off

Description

Turn off messages are generated by the processor in case a tracable exception was encountered, like bus errors and address errors, missing dtacks....

Name

RUN (diag)
run history

Characteristics

max entries	15
max characters	30
storage	non volatile
availability	always

Contents

start time, end time, start mode, end mode

start time	seconds since 1970, the time when the analysis was started
end time	seconds since 1970, the time when the analysis was terminated
start mode	0 for local start; 1 for global start (e.g. with remote start puls generation)
end mode	0 for normal end; 1 for aborted

Description

One entry per start of a time table is done as soon as the run is done

Name

AHIS (diag)

analysis history

Characteristics

max entries	15
max characters	20
storage	non volatile
availability	always

Contents

start time, start/stop, type of start/stop, #of modules, analysis ID

start time	seconds since 1970, the time when the analysis was started
start/stop	1 for start, 0 for stop
type/end	if start: 3 for normal analysis, 4 for APG remote start pulse analysis if stop: 1 normal stop, 2 abort by this module, 3 abort by other module 4 abort due to disconnected module
#of modules	the number of Agilent 1200 Series modules that responded to the CAN start synchronization.
analysis ID	a 12 bit number, incremented with each start of an analysis

Description

Name

MTNC (diag)

maintenance history

Characteristics

max entries	20
max characters	60
storage	non erasible
availability	always

Contents

time, maintenance string

time	seconds since 1970
maintenance string	externally defined string of maximum 50 characters

Description

remark: Any nonesens of contents is possible; ask your chemstation how the buffers are used. The instruction to enter a string is *DIAG:MTNC <string>*. See also instruction list.

Name

EXB0 (diag)
EXB1, EXB2, EXB3 (diag)
buffers for external diagnostic

Characteristics

max entries	20
max characters	40
storage	non erasible
availability	always

Contents

time, external string

time	seconds since 1970
external string	externally defined string of maximum 40 characters

Description

remark: Any nonesens of contents is possible; ask your chemstation how the buffers are used. The command to enter a string into EXB0 is *DIAG:EXB0 <string>*. See also the instruction lists..

Name

BTMS (diag)
boardtemperature short time buffer

Characteristics

max entries	90
max characters	12
storage	non volatile
availability	always
storage frequency	always: 1 Hz (->overall time is 1.5 minutes)

Contents

board temperature, leak sensor current

board temperature,	degree celsius of the pc-board in 0.1degC
leaksensor current,	current of the PTC leak sensor in 0.1mA

Description

Name

BTML (diag)

boardtemperature long time buffer

Characteristics

max entries	144
max characters	12
storage	non volatile
availability	always
storage frequency	always: 5 minutes (->overall time is 12 hours)

Contents

board temperature, leak sensor current

board temperature,	degree celsius of the pc-board in 0.1degC
leaksensor current,	current of the PTC leak sensor in 0.1mA

Description

Name

FAN (diag)

fan frequency and cover sensor

Characteristics

max entries	30
max characters	12
storage	volatile
availability	always
storage frequency	once per second

Contents

frequency, cover

frequency,	rotabtions of the fan per second
cover	1 for cover open, 0 for cover closed

Description

Name

TM1S (diag)
auxilliary temperature sensor 1, seconds

Characteristics

max entries	50
max characters	12
storage	volatile
availability	always
storage frequency	once per second

Contents

temperature

temperature	-40 to 140 degrC in 0.01 units

Description

Sensor function must be turned on with "TMP1 1"

Name

TM1M (diag)
auxilliary temperature sensor 1, minutes

Characteristics

max entries	50
max characters	12
storage	volatile
availability	always
storage frequency	once per second

Contents

temperature

temperature,	-40 to 140 degrC in 0.01 units

Description

Sensor function must be turned on with "TMP1 1"

Name

TM2S (diag)
auxilliary temperature sensor 2, seconds

Characteristics

max entries	50
max characters	12
storage	volatile
availability	always
storage frequency	once per second

Contents

temperature

temperature,	-40 to 140 degrC in 0.01 units

Description

Sensor function must be turned on with "TMP2 1"

Name

TM2M (diag)
auxilliary temperature sensor 2, minutes

Characteristics

max entries	50
max characters	12
storage	volatile
availability	always
storage frequency	once per second

Contents

temperature

temperature,	-40 to 140 degrC in 0.01 units

Description

Sensor function must be turned on with "TMP2 1"

18.4. Common Diagnose Counters

Name

CRUN (diag)

timetable start counter

storage type: non volatile
history size: 2
limit event: EV 402

increments by one with each start of a time table, multiple timetable starts may occur during on analysis. A timetable may be started with the instructions RUN or RUNG (typically from the main COSY list) or remotely by a start puls or with a remote start command directly via CAN.

Name

CNAN (diag)

analysis start counter

storage type: on volatile
history size: 2
limit event: EV 403

increments by one with each start of an analysis. An analysis may be started with the instructions STRT or STRS or remotely by a start request puls or with an analysis start command directly via CAN (the normal way in a CAN cluster).

Name

TONT (diag)

On time counter

storage type: non erasible
history size: 5
limit event: EV 401

increments by one after each second the module is in the power on state.
The Erasable part is set to 0 with each cold start,
The limit detection is still available after a normal power on.

19. Pumps (G1310A, G1311A, G1312A, G1312B)

19.1. Pumps Method Parameter

Name

FLOW
Set flow value

Synopsis

Instruction:	`FLOW <flow value>`
Reply:	`<rc> FLOW <flow value>`
Instruction:	`FLOW?`
Reply:	`<rc> FLOW <flow value>`
Instruction:	`ACT:FLOW?`
Reply:	`<rc> ACT:FLOW <flow value>`
Instruction:	`AT:FLOW <time>, <flow value>`
Reply:	`<rc> AT:FLOW <time>, <flow value>`

Description

The FLOW parameter defines the total flow to be pumped after the pump is turned on.
The ACT:FLOW? query returns the actual flow. If the pump is not on, a value of 0 is returned.

Parameters

<flow value>	for Isocratic and Quaternary (G1310A, G1311A): 0 .. 10.000 [ml/min] for Binary and Binary-SL (G1312A, G1312B): 0 .. 5.000 [ml/min] a flow value of greater than 5ml/min is rejected if a high pressure limit is set to more than 200bar.
`<time>`	0 .. 99999.00 [min]

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 2032` pump on rejected while leak or shutdown or permanent erro or license code

Examples

(a) Comment: set flow to 0.222 ml/min
Instruction: `FLOW 0.222`
Reply: `RA 0000 FLOW 0.222`

(b) Comment: Query the flow setpoint
Instruction: `FLOW?`
Reply: `RA 0000 FLOW 0.222`

(c) Comment: Query for actual flow
Instruction: `ACT:FLOW?`
Reply: `RA 0000 ACT:FLOW 0.222`

(d) Comment: Set Time Table entry at 1.5 min with flow 2 ml/min
Instruction: `AT:FLOW 1.5, 2`
Reply: `RA 0000 AT:FLOW 1.5, 2.000`

Name

COMP

Set Composition Value

Synopsis

Instruction:	`COMP <%B>, <%C>, <%D>`
Reply:	`<rc> COMP <%B>, <%C>, <%D>`
Instruction:	`COMP?`
Reply:	`<rc> COMP <%B>, <%C>, <%D>`
Instruction:	`ACT:COMP?`
Reply:	`<rc> ACT:COMP <%B>, <%C>, <%D>`
Instruction:	`AT:COMP <time>, <%B>, <%C>, <%D>`
Reply:	`<rc> AT:COMP <time>, <%B>, <%C>, <%D>`

Description

The COMP instruction defines the Composition of the mobile phase. The value for channel A is defined to be 100% - %B -%C -%D. If %B + %C > 100%, %C is set to 100 - %B. If %B + %C +%D > 100% then %D is set to 100 - %B - %C. If a value is set to -1, the corresponding channel is off. For the normal home value -1 is identical to 0. But for the timetable execution, all entries specified for the channel that is set to off, are considered to be 0.

The ACT:COMP? query returns the actual composition.

Parameters

<%B>	-1, 0 .. 100.0 [%] ignored if isocratic, -1 means off
<%C>	-1, 0 .. 100.0 [%] ignored if isocratic or binary, -1 means off
<%D>	-1, 0 .. 100.0 [%] ignored if isocratic or binary, -1 means off
<time>	0 .. 99999.00 [min]

Default Values

0,0,0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set composition to equal partion for quaternary pump
	Instruction:	`COMP 25,25,25`
	Reply:	`RA 0000 COMP 25,25,25`
(b)	Comment:	Query composition setpoint
	Instruction:	`COMP?`
	Reply:	`RA 0000 COMP 25,25,25`
(c)	Comment:	Query for actual composition
	Instruction:	`ACT:COMP?`
	Reply:	`RA 0000 ACT:COMP 25,25,25`
(d)	Comment:	define time table entry for composition gradient
	Instruction:	`AT:COMP 5, 50,50,0`
	Reply:	`RA 0000 AT:COMP 5.00, 50.0, 50.0, 0`

Name

`PMCH`

Set Primary Channel

Synopsis

Instruction:	`PMCH <channel>`
Reply:	`<rc> PMCH <channel>`
Instruction:	`PMCH?`
Reply:	`<rc> PMCH <channel>`
Instruction:	`ACT:PMCH?`
Reply:	`<rc> ACT:PMCH <channel>`

Description

The PMCH instruction defines the Primary Channel. This is the Channel to be used first and last in the sequence of solvent channel during takein. This instruction is only sensful for quaternary pumps. The primary channel is split in half: the first half is taken at the start of the take in phase the second half is taken at the end of the takein phase.

If the parameter is set to 0, (automatic mode) the greatest channel is selected as primary channel.

Parameters

<channel>	0 .. 4, 0 for automatic, 1 for channel A, 2 for channel B, 3 for channel C and 4 for channel D

Default Values

0 (automatic)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set primary channel to channel C
	Instruction:	`PMCH 3`
	Reply:	`RA 0000 PMCH 3`
(b)	Comment:	Query the setpoint
	Instruction:	`PMCH?`
	Reply:	`RA 0000 PMCH 3`
(c)	Comment:	Query for actual primary channel
	Instruction:	`ACT:PMCH?`
	Reply:	`RA 0000 ACT:PMCH 3`

Name

HIPR
Set High Pressure Limit

Synopsis

Instruction:	`HIPR <pressure>`
Reply:	`<rc> HIPR <pressure>`
Instruction:	`HIPR?`
Reply:	`<rc> HIPR <pressure>`
Instruction:	`ACT:HIPR?`
Reply:	`<rc> ACT:HIPR <pressure>`
Instruction:	`AT:HIPR <time>, <pressure>`
Reply:	`<rc> AT:HIPR <time>, <pressure>`

Description

The HIPR instruction sets and queries the high pressure limit. Once the actual pressure is larger than the limit, an error event is generated and the pump stops from delivering mobile phase.

Parameters

<pressure>	0 ... 400.0 [bar] if flow is below or equal 5 ml/min, otherwise see reply error 0 ... 200.0 [bar] if flow is greater 5 ml/min 0 ... 600.0 [bar] for G1312B
<time>	0 .. 99999.00 [min]

Default Values

400

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 2001 flow too high for high pressure limit

Examples

(a)	Comment:	set pressure limit to 150 bar
	Instruction:	`HIPR 150`
	Reply:	`RA 0000 HIPR 150.0`
(b)	Comment:	Query high pressure limit parameter setting
	Instruction:	`HIPR?`
	Reply:	`RA 0000 HIPR 150`
(c)	Comment:	Query for actual high pressure limit
	Instruction:	`ACT:HIPR?`
	Reply:	`RA 0000 ACT:HIPR 150.0`
(d)	Comment:	define time table at 3 minutes to high pressure limit of 300 bar
	Instruction:	`AT:HIPR 3.00, 300`
	Reply:	`RA 0000 AT:HIPR 3.00, 300`

Name

CBLT

Set Compressibility for isocratic and quaternary pump

Synopsis

Instruction:	`CBLT <compressibility>`
Reply:	`<rc> CBLT <compressibility>`
Instruction:	`CBLT?`
Reply:	`<rc> CBLT <compressibility>`
Instruction:	`ACT:CBLT?`
Reply:	`<rc> ACT:CBLT <compressibility>`

Description

The CBLT instruction set the compressibility value for the compensation calculation of the left pump assembly. See CCBL for the right pump assembly/

Parameters

<compressibility> 40 .. 150 [1/bar * 10E-6]

Default Values

100

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	set compressibility to 100
	Instruction:	`CBLT 100`
	Reply:	`RA 0000 CBLT 100`
(b)	Comment:	Query compressibility
	Instruction:	`CBLT?`
	Reply:	`RA 0000 CBLT 100`
(c)	Comment:	Query for actual compressibility
	Instruction:	`ACT:CBLT?`
	Reply:	`RA 0000 ACT:CBLT 100`

Name

CCBL

Set Compressibility for both drive assemblise of the binary pump

Synopsis

Instruction:	`CCBL <left>, <right>`
Reply:	`<rc> CCBL <left>, <right>`
Instruction:	`CBLT?`
Reply:	`<rc> CCBL <left>,<right>`
Instruction:	`ACT:CCBL?`
Reply:	`<rc> ACT:CCBL <left>,<right>`

Description

The CCBL instruction sets the compressibility value for the compensation calculation of the left and right pump assembly for the binary pump. See CBLT for the other pump types

Parameters

<left>	40 .. 150 [1/bar * 10E-6] for G1312B: -1 for user defined solvent -2 for "H2O" -3 for "ACN" -4 for "MeOH" -5 for "IPA" -6 for "premixed ACN(95%) - H2O(5%)" -7 for "premixed ACN(90%) - H20(10%)"
<right>	same values as left

Default Values

50 for <left>, 115 for <right>

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set compressibility to 75
	Instruction:	`CCBL 75,75`
	Reply:	`RA 0000 CCBL 75,75`
(b)	Comment:	Query compressibility
	Instruction:	`CCBL?`
	Reply:	`RA 0000 CCBL 75,75`
(c)	Comment:	Query for actual compressibility
	Instruction:	`ACT:CCBL?`
	Reply:	`RA 0000 ACT:CCBL 75,75`

Name

STRK

Set Stroke Value

Synopsis

Instruction:	`STRK <stroke value>`
Reply:	`<rc> STRK <stroke value>`
Instruction:	`STRK?`
Reply:	`<rc> STRK <stroke value>`
Instruction:	`ACT:STRK?`
Reply:	`<rc> ACT:STRK <stroke value>`

Description

The STRK instruction defines the stroke value for the left drive assembly of the pump. See CSTR instruction for the right drive assembly.

Any stroke value in the range below is accepted, but if the minimum stroke value for a given flow is higher than the setpoint, the actual stroke will be the stroke value.

Parameters

<stroke value>	-1, 20 .. 100 [micro liter] -1 means automatic stroke calculation

Default Values

-1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set stroke value to 75 ul
	Instruction:	`STRK 75`
	Reply:	`RA 0000 STRK 75`
(b)	Comment:	Query stroke parameter setting
	Instruction:	`STRK?`
	Reply:	`RA 0000 STRK 75`
(c)	Comment:	Query for actual posttime
	Instruction:	`ACT:STRK?`
	Reply:	`RA 0000 ACT:STRK 75`

Name

CSTR
Set Stroke Value for the Right Drive Assembly

Synopsis

Instruction:	`CSTR <stroke value>`
Reply:	`<rc> CSTR <stroke value>`
Instruction:	`CSTR?`
Reply:	`<rc> CSTR <stroke value>`
Instruction:	`ACT:CSTR?`
Reply:	`<rc> ACT:CSTR <stroke value>`

Description

The CSTR instruction defines the stroke value for the right drive assembly of the pump. See STRK instruction for the left drive assembly.
Any stroke value in the range below is accepted, but if the minimum stroke value for a given flow is higher than the setpoint, the actual stroke will be the stroke value.

Parameters

<stroke value>	-1, 20 .. 100 [micro liter] -1 means automatic stroke calculation

Default Values

-1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set stroke value to 75 ul
	Instruction:	`CSTR 75`
	Reply:	`RA 0000 CSTR 75`
(b)	Comment:	Query stroke parameter setting
	Instruction:	`CSTR?`
	Reply:	`RA 0000 CSTR 75`
(c)	Comment:	Query for actual posttime
	Instruction:	`ACT:CSTR?`
	Reply:	`RA 0000 ACT:CSTR 75`

Name

LOPR

Set Low Pressure Limit

Synopsis

Instruction:	`LOPR <pressure>`
Reply:	`<rc> LOPR <pressure>`
Instruction:	`LOPR?`
Reply:	`<rc> LOPR <pressure>`

Description

The LOPR instruction sets the low pressure limit. Once the pressure drops below this limit (this means if the pressure was greater! before) an error event is generated and the pump stops from delivering mobile phase.

Parameters

<pressure>	0 .. 400.0 [bar], 0 .. 600.0 for G1312B 0 means off

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set low pressure limit to 80 bar
	Instruction:	`LOPR 80`
	Reply:	`RA 0000 LOPR 80`
(b)	Comment:	Query low pressure parameter setting
	Instruction:	`LOPR?`
	Reply:	`RA 0000 LOPR 80.0`

Name

FRMP
Set Maximum Flow Ramp

Synopsis

Instruction:	`FRMP <ramp>`
Reply:	`<rc> FRMP <ramp>`
Instruction:	`FRMP?`
Reply:	`<rc> FRMP <ramp>`

Description

The FRMP instruction defines the maximum change of the flow within one minute.

Parameters

<ramp>	0.100 .. 100.000 [ml/(min * min)]

Default Values

100 [the fastest]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set Flow ramp to max 20 ml/(min*min)
	Instruction:	`FRMP 20`
	Reply:	`RA 0000 FRMP 20.000`
(b)	Comment:	Query flow ramp parameter setting
	Instruction:	`FRMP?`
	Reply:	`RA 0000 FRMP 20.000`

Name

SSVA

Switches the solvent selection valve for the left drive assemble

Synopsis

Instruction:	`SSVA <Channel>`
Reply:	`<rc> SSVA <Channel>`
Instruction:	`SSVA?`
Reply:	`<rc> SSVA <Channel>`

Description

The SSVA instruction switches the solvent selection valve to one out of two sovent channels.

Parameters

<Channel>	0 for A1, 1 for A2

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0752` option not installed

Examples

(a)	Comment:	set solvent selection valve to 1
	Instruction:	`SSVA 1`
	Reply:	`RA 0000 SSVA 1`
(b)	Comment:	Query solvent selection valve parameter setting
	Instruction:	`SSVA?`
	Reply:	`RA 0000 SSVA 1`

Name

SSVB

Switches the solvent selection valve for the right drive assemble

Synopsis

Instruction: `SSVB <Channel>`
Reply: `<rc> SSVB <Channel>`
Instruction: `SSVB?`
Reply: `<rc> SSVB <Channel>`

Description

The SSVB instruction switches the solvent selection valve to one out of two sovent channels.

Parameters

<Channel> 0 for B1, 1 for B2

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0752 option not installed

Examples

(a) Comment: set solvent selection valve to 1
Instruction: `SSVB 1`
Reply: `RA 0000 SSVB 1`

(b) Comment: Query solvent selection valve parameter setting
Instruction: `SSVB?`
Reply: `RA 0000 SSVB 1`

Name

POSC

Defines the Postconditioning cosy list

Synopsis

Instruction:	`POSC <list number>`
Reply:	`<rc> POSC <list number>`
Instruction:	`POSC?`
Reply:	`<rc> POSC <actual list number>`

Description

The POSC instruction defines the cosy list to be used as the post condition list. The default list will be used if a number < 0 is used.

Parameters

<list number>	< 0 for default, > 0 for privately defined once

Default Values

-21

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set post conditioning list to 5
	Instruction:	`POSC 5`
	Reply:	`RA 0000 POSC 5`
(b)	Comment:	Query the post condition list
	Instruction:	`POSC?`
	Reply:	`RA 0000 POSC 5`

Name

PREC

Defines the Preconditioning cosy list

Synopsis

Instruction: `PREC <list number>`

Reply: `<rc> PREC <list number>`

Instruction: `PREC?`

Reply: `<rc> PREC <actual list number>`

Description

The PREC instruction defines the cosy list to be used as the precondition list. The default list will be used if a number < 0 is used.

Parameters

<list number> < 0 for default, > 0 for privately defined once

Default Values

-21

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set preconditioning list to 5
Instruction: `PREC 5`
Reply: `RA 0000 PREC 5`

(b) Comment: Query the precondition list
Instruction: `PREC?`
Reply: `RA 0000 PREC 5`

Name

MODF

Modulates the flow value (not part of the default behaviour)

Synopsis

Instruction:	`MODF <Period>, <Amplitude>`
Reply:	`<rc> MODF <Period>, <Amplitude>`
Instruction:	`MODF?`
Reply:	`<rc> MODF <Period>, <Amplitude>`

Description

The MODF instruction defines the a period and amplitude to modulate the flow value with a sinus.
REMARK: Only for Test purposes, results are unpredictable, not recommended for use.

Parameters

<Period>	1.0 .. 1000.0 seconds
<Amplitude>	-100.0 .. 100.0 percent of the actual method setpoint.

Default Values

Period 1.0
Amplitude 0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	set flow modulation
	Instruction:	`MODF 20, 50`
	Reply:	`RA 0000 MODF 20, 50`
(b)	Comment:	Query the modulation set points
	Instruction:	`MODF?`
	Reply:	`RA 0000 MODF 20,50`

Name

RMPC

Defines the Slope for a Composition Ramp (not part of the default behaviour)

Synopsis

Instruction:	`RMPC <B>,<C>,<D>`
Reply:	`<rc> RMPC <B>,<C>,<D>`
Instruction:	`RMPC?`
Reply:	`<rc> RMPC <B>,<C>,<D>`

Description

The RMPC instruction defines the slope for the composition ramp functionality (see also instructions (RMPC:STRT, RMPC:STOP, RMPC:HOLD, RMPC:CONT and RCL0 and RCL1).

Remark: The implementation allows only a binary ramp, even though three parameters are present.

Parameters

<B>,<C>,<D>	< 0 for default, > 0 for privately defined once

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set composition ramp to 20% per minute
	Instruction:	`RMPC 20,0,0`
	Reply:	`RA 0000 RMPC 20.0,0,0`
(b)	Comment:	Query the slope
	Instruction:	`RMPC?`
	Reply:	`RA 0000 RMPC 20.0,0,0`

Name

RCL0

Defines a limit for the composition ramp (not part of the default behaviour)

Synopsis

Instruction:	`RCL0 <%B>,<%C>,<%D>`
Reply:	`<rc> RCL0 <%B>,<%C>,<%D>`
Instruction:	`RCL0?`
Reply:	`<rc> RCL0 <%B>,<%C>,<%D>`

Description

The RCL0 instruction defines a limit for the composition ramp function.

Parameters

<%B>,<%C>,<%D> 0 .. 100.0 percent

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set limit for composition ramp
	Instruction:	`RCL0 5`
	Reply:	`RA 0000 RCL0 5`
(b)	Comment:	Query the precondition list
	Instruction:	`RCL0?`
	Reply:	`RA 0000 RCL0 5`

Name

RCL1

Defines a limit for the composition ramp (not part of the default behaviour)

Synopsis

Instruction:	`RCL1 <%B>,<%C>,<%D>`
Reply:	`<rc> RCL1 <%B>,<%C>,<%D>`
Instruction:	`RCL1?`
Reply:	`<rc> RCL1 <%B>,<%C>,<%D>`

Description

The RCL0 instruction defines a limit for the composition ramp function.

Parameters

<%B>,<%C>,<%D> 0 .. 100.0 percent

Default Values

100.0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set limit for composition ramp
	Instruction:	`RCL1 95`
	Reply:	`RA 0000 RCL1 95`
(b)	Comment:	Query the precondition list
	Instruction:	`RCL1?`
	Reply:	`RA 0000 RCL1 95`

Name

RMPF

Sets value for the flow ramp (not part of the default behaviour)

Synopsis

Instruction:	`RMPF <Slope>`
Reply:	`<rc> RMPF <Slope>`
Instruction:	`RMPF?`
Reply:	`<rc> RMPF <Slope>`

Description

The RMPF instruction defines the Slope value for the flow ramp functionaliy. When started (see instruction RMPF:STRT) the flow ramp progresses from the actual flow value with the given slope till either a limit is reached (see RFL0 and RFL1) or the ramp is suspended (see the instructions RMPF:STOP or RMPF:HOLD...). When used within a COSY list, the sync is sent as soon as a limit is reached.

Parameters

<Slope>	-100.000 ... 100.000 ml/(min*min)

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set slope to 50 ml/(min*min)
	Instruction:	`RMPF 50`
	Reply:	`RA 0000 RMPF 50`
(b)	Comment:	Query the slope
	Instruction:	`RMPF?`
	Reply:	`RA 0000 RMPF 50`

Name

RFL0

Limit for Flow Ramp (not part of default behaviour)

Synopsis

Instruction:	`RFL0 <limit>`
Reply:	`<rc> RFL0 <limit>`
Instruction:	`RFL0?`
Reply:	`<rc> RFL0 <limit>`

Description

The RFL0 instruction defines a limit for the flow ramp functionality. The ramp will not exceed this limit.

Parameters

<limit>	0 .. 10.000 ml/min

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set ramp limit to 0.5 ml/min
	Instruction:	`RFL0 0.5`
	Reply:	`RA 0000 RFL0 0.500`
(b)	Comment:	Query the limit
	Instruction:	`RFL0?`
	Reply:	`RA 0000 RFL0 0.500`

Name

RFL1

Limit for Flow Ramp (not part of default behaviour)

Synopsis

Instruction:	`RFL1 <limit>`
Reply:	`<rc> RFL1 <limit>`
Instruction:	`RFL1?`
Reply:	`<rc> RFL1 <limit>`

Description

The RFL1 instruction defines a limit for the flow ramp functionality. The ramp will not exceed this limit.

Parameters

<limit>	0 .. 10.000 ml/min

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	set ramp limit to 4 ml/min
	Instruction:	`RFL1 4`
	Reply:	`RA 0000 RFL1 4.000`
(b)	Comment:	Query the limit
	Instruction:	`RFL1?`
	Reply:	`RA 0000 RFL1 0.500`

Name

CTA1

Compressibility table (G1312B only))

Synopsis

Instruction:	`CTA1 <cblt>, <cblt>, <cblt>`
Reply:	`<rc> CTA1 <cblt>, <cblt>, <cblt>`
Instruction:	`CTA1?`
Reply:	`<rc> CTA1 <cblt>, <cblt>, <cblt>`

Description

Defines the first three values of the user defined compressibility table.

Parameters

<cblt>	0 .. 250

Default Values

45

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set compressibility table entries
	Instruction:	`CTA1 115,106,98`
	Reply:	`RA 0000 CTA1 115, 106, 98`
(b)	Comment:	Query table entries
	Instruction:	`CTA1?`
	Reply:	`RA 0000 CTA1 115,106,98`

Name

CTA2

Compressibility table (G1312B only))

Synopsis

Instruction:	`CTA2 <cblt>, <cblt>, <cblt>`
Reply:	`<rc> CTA2 <cblt>, <cblt>, <cblt>`
Instruction:	`CTA2?`
Reply:	`<rc> CTA2 <cblt>, <cblt>, <cblt>`

Description

Defines the second three values of the user defined compressibility table.

Parameters

<cblt>	0 .. 250

Default Values

45

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set compressibility table entries
	Instruction:	`CTA2 92,87,82`
	Reply:	`RA 0000 CTA2 92,87,82`
(b)	Comment:	Query table entries
	Instruction:	`CTA2?`
	Reply:	`RA 0000 CTA2 92,87,82`

Name

CTA3

Compressibility table (G1312B only))

Synopsis

Instruction:	`CTA3 <cblt>`
Reply:	`<rc> CTA3 <cblt>`
Instruction:	`CTA3?`
Reply:	`<rc> CTA3 <cblt>`

Description

Defines the last value of the user defined compressibility table.

Parameters

<cblt>	0 .. 250

Default Values

45

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set compressibility table entry	
	Instruction:	`CTA3 78`	
	Reply:	`RA 0000 CTA3 78`	
(b)	Comment:	Query table entries	
	Instruction:	`CTA3?`	
	Reply:	`RA 0000 CTA3 78`	

Name

CTB1

Compressibility table (G1312B only))

Synopsis

Instruction:	`CTB1 <cblt>, <cblt>, <cblt>`
Reply:	`<rc> CTB1 <cblt>, <cblt>, <cblt>`
Instruction:	`CTB1?`
Reply:	`<rc> CTB1 <cblt>, <cblt>, <cblt>`

Description

Defines the first three values of the user defined compressibility table.

Parameters

<cblt> 0 .. 250

Default Values

45

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set compressibility table entries
	Instruction:	`CTB1 115,106,98`
	Reply:	`RA 0000 CTB1 115, 106, 98`
(b)	Comment:	Query table entries
	Instruction:	`CTB1?`
	Reply:	`RA 0000 CTB1 115,106,98`

Name

CTB2

Compressibility table (G1312B only))

Synopsis

Instruction:	`CTB2 <cblt>, <cblt>, <cblt>`
Reply:	`<rc> CTB2 <cblt>, <cblt>, <cblt>`
Instruction:	`CTB2?`
Reply:	`<rc> CTB2 <cblt>, <cblt>, <cblt>`

Description

Defines the second three values of the user defined compressibility table.

Parameters

<cblt>	0 .. 250

Default Values

45

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set compressibility table entries
	Instruction:	`CTB2 92,87,82`
	Reply:	`RA 0000 CTB2 92,87,82`
(b)	Comment:	Query table entries
	Instruction:	`CTB2?`
	Reply:	`RA 0000 CTB2 92,87,82`

Name

CTB3

Compressibility table (G1312B only))

Synopsis

Instruction:	`CTB3 <cblt>`
Reply:	`<rc> CTB3 <cblt>`
Instruction:	`CTB3?`
Reply:	`<rc> CTB3 <cblt>`

Description

Defines the last value of the user defined compressibility table.

Parameters

<cblt>	0 .. 250

Default Values

45

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set compressibility table entry
	Instruction:	`CTB3 78`
	Reply:	`RA 0000 CTB3 78`
(b)	Comment:	Query table entries
	Instruction:	`CTB3?`
	Reply:	`RA 0000 CTB3 78`

Name

CTHA

Solvent thermal specification (G1312B only))

Synopsis

Instruction:	`CTHA <vol>, <delay>, <duration>`
Reply:	`<rc> CTHA <vol>, <delay>, <duration>`
Instruction:	`CTHA?`
Reply:	`<rc> CTHA <vol>, <delay>, <duration>`

Description

Defines the thermal solvent specification.

Parameters

<vol>	0 .. 3000.000 [nl]
<delay>	0 .. 2^32 [ticks]
<duration>	0 .. 2^32 [ticks]

Default Values

0,0,0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	set solvent parameters
	Instruction:	`CTHA 50,5,30`
	Reply:	`RA 0000 CTHA 50,5,30`
(b)	Comment:	Query solvent parameters
	Instruction:	`CTHA?`
	Reply:	`RA 0000 CTHA 50,5,30`

Name

CTHB

Solvent thermal specification (G1312B only))

Synopsis

Instruction:	`CTHB <vol>, <delay>, <duration>`
Reply:	`<rc> CTHB <vol>, <delay>, <duration>`
Instruction:	`CTHB?`
Reply:	`<rc> CTHB <vol>, <delay>, <duration>`

Description

Defines the thermal solvent specification.

Parameters

<vol>	0 .. 3000.000 [nl]
<delay>	0 .. 2^32 [ticks]
<duration>	0 .. 2^32 [ticks]

Default Values

0,0,0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set solvent parameters
	Instruction:	`CTHB 50,5,30`
	Reply:	`RA 0000 CTHB 50,5,30`
(b)	Comment:	Query solvent parameters
	Instruction:	`CTHB?`
	Reply:	`RA 0000 CTHB 50,5,30`

Name

OPRS

Set Pressure Value for Operating Pressure Control (G1312B)

Synopsis

Instruction:	`OPRS <Pressure>`
Reply:	`<rc> OPRS <Pressure>`
Instruction:	`OPRS?`
Reply:	`<rc> OPRS <Pressure>`
Instruction:	`ACT:OPRS?`
Reply:	`<rc> ACT:OPRS <Pressure>`
Instruction:	`AT:OPRS <time>, <Pressure>`
Reply:	`<rc> AT:OPRS <time>, <Pressure>`

Description

The ’Operating Pressure Control’ limis the flow to a value that prevents high pressure limit violation. This eases up method load for unconditioned systems. If the pressure given is 0, the used value for controlling is defined as the actual high pressure limit bar.

Parameters

<Pressure>	0 [for automatic] .. 600 bar
<time>	0 .. 99999.0 minutes

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	set operating pressure to 145
	Instruction:	`OPRS 145`
	Reply:	`RA 0000 OPRS 145`
(b)	Comment:	Query operating pressure
	Instruction:	`OPRS?`
	Reply:	`RA 0000 ORPS 145`

Name

OPM
Operating pressure mode (G1312B)

Synopsis

Instruction:	`OPM <mode>`
Reply:	`<rc> OPM <mode>`
Instruction:	`OPM?`
Reply:	`<rc> OPM <mode>`

Description

The OPM instruction defines the operating pressure control mode, that is, the time when the operatin pressure control is active. Once activ, the system pressure will not exceed the limit defined by the OPRS instruction, and thus will prevent an overpressure error generation (if the OPRS limit is lower than the high pressure limit). Instead the flow will be controlled down to keep the pressure at the OPRS-defined limit. As long as flow is reduced the pump shows 'not ready flow reduction'.

Parameters

<mode>	0 for off, 1 for active if not in analysis, 2 for always active

Default Values

1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set operating pressure mode to 1
	Instruction:	`OPM 1`
	Reply:	`RA 0000 OPM 1`
(b)	Comment:	Query operating pressure mode
	Instruction:	`OPM?`
	Reply:	`RA 0000 OPM 1`

19.2. Pumps System Parameter

Name

ELAS

Set Elasticity for Channel A (not G1312B)

Synopsis

Instruction: `ELAS <Elasticity>`

Reply: `<rc> ELAS <ELasticity>`

Instruction: `ELAS?`

Reply: `<rc> ELAS <Elasticity>`

Description

The Elasticity of the mechanical pump hardware (mainly the aluminium parts and the seals) are used for the compensation calculation. With each deliver phase a new compensation value is calculated with the actual pressure, the actual stroke and the compressibility of the mobile phase.

Parameters

<Elasticity> 0 .. 20.0 [nl/bar]

Default Values

8.0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: Set Elasticity to 8.4
Instruction: `ELAS 8.4`
Reply: `RA 0000 ELAS 8.4`

(b) Comment: Query the current setpoint
Instruction: `ELAS?`
Reply: `RA 0000 ELAS 8.4`

Name

CELA

Set Elasticity for Channel B (not G1312B)

Synopsis

Instruction:	`CELA <Elasticity>`
Reply:	`<rc> CELA <ELasticity>`
Instruction:	`CELA?`
Reply:	`<rc> CELA <Elasticity>`

Description

The Elasticity of the mechanical pump hardware (mainly the aluminium parts and the seals) are used for the compensation calculation. With each deliver phase a new compensation value is calculated with the actual pressure, the actual stroke and the compressibility of the mobile phase.

Parameters

<Elasticity>	0 .. 20.0 [nl/bar]

Default Values

8.0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set Elasticity to 8.4
	Instruction:	`CELA 8.4`
	Reply:	`RA 0000 CELA 8.4`
(b)	Comment:	Query the current setpoint
	Instruction:	`CELA?`
	Reply:	`RA 0000 CELA 8.4`

Name

DVOL

Set DeadVolume for Channel A(not G1312B)

Synopsis

Instruction:	`DVOL <Volume>`
Reply:	`<rc> DVOL <Volume>`
Instruction:	`DVOL?`
Reply:	`<rc> DVOL <Volume>`

Description

The dead volume of the pump drive assembly is used for the compensation calculation and for the actual stroke length calculation. With each deliver phase a new compensation value is calculated with the actual pressure, the actual stroke, the dead volume and the compressibility of the mobile phase.

Parameters

<Volume>	0 .. 200.0 [ul]

Default Values

40.0 for G1310A,G1311A,G1312A,G1376A,G2226A
80.0 for G1312B

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Set dead volume to 44
	Instruction:	`DVOL 44`
	Reply:	`RA 0000 DVOL 44.0`
(b)	Comment:	Query the current setpoint
	Instruction:	`DVOL?`
	Reply:	`RA 0000 DVOL 44.0`

Name

CDVO
Set DeadVolume for Channel B (not G1312B)

Synopsis

Instruction:	`CDVO <Volume>`
Reply:	`<rc> CDVO <Volume>`
Instruction:	`CDVO?`
Reply:	`<rc> CDVO <Volume>`

Description

The dead volume of the pump drive assembly is used for the compensation calculation and for the actual stroke length calculation. With each deliver phase a new compensation value is calculated with the actual pressure, the actual stroke, the dead volume and the compressibility of the mobile phase.

Parameters

<Volume>	0 .. 200.0 [ul]

Default Values

40.0 for G1310A,G1311A,G1312A,G1376A,G2226A
80.0 for G1312B

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Set dead volume to 44
	Instruction:	`CDVO 44`
	Reply:	`RA 0000 CDVO 44.0`
(b)	Comment:	Query the current setpoint
	Instruction:	`CDVO?`
	Reply:	`RA 0000 CDVO 44.0`

Name

PRMP

Set Maximum Pressure Ramp

Synopsis

Instruction:	`PRMP <Ramp>`
Reply:	`<rc> PRMP <Ramp>`
Instruction:	`PRMP?`
Reply:	`<rc> PRMP <Ramp>`

Description

PRMP sets the value for the maximum pressure increment per second. If this value is violated the pump automatically reduces the flow till a pressure increment below the given value is reached. The main use for the limited pressure increment is the protection of the analytical column.

Parameters

<Ramp>	0 .. 100	[bar/sec]

Default Values

20

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set pressure ramp to 25 bar/second
	Instruction:	`PRMP 25`
	Reply:	`RA 0000 PRMP 25`
(b)	Comment:	Query the current setpoint
	Instruction:	`PRMP?`
	Reply:	`RA 0000 PRMP 25`

Name

ESER
Enable Solvent Error

Synopsis

Instruction:	`ESER <on/off>`
Reply:	`<rc> ESER <on/off>`
Instruction:	`ESER?`
Reply:	`<rc> ESER <on/off>`

Description

ESER enables the generation of an error in case the limit of a bottle counter is reached. Depending on the type of pump, different channels are affected.
The generation of an error depends on:
a) Error generation is enabled (ESER 1),
b) the channel is not off (COMP x,x,x where x != -1), channel A is always on
c) the limit of a bottle counter is reached (DIAG:CLMT name, limit)
d) the limit of a bottle counter is greater than 0.
The error event generated is EE 2055,<bottle number>
the bottle numbers are 1 for BOTA counter, 2 for BOTB, 3 for BOTC, 4 for BOTD.
For a binary pump without Solvent Selection Valve, the counters BOTA and BOTB are used, in case a Solvent Selection Valve is present, the counters BOTA and BOTB are used for SSVA and the counter BOTC and BOTD are used for SSVB.

Parameters

<on/off>	0 for disabled(off) and 1 for enabled

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Enable Solvent Error generation
	Instruction:	`ESER 1`
	Reply:	`RA 0000 ESER 1`
(b)	Comment:	Query the current setpoint
	Instruction:	`ESER?`
	Reply:	`RA 0000 ESER 1`

Name

SNRL
Solvent Not Ready Limit

Synopsis

Instruction:	`SNRL <on/off>,<LimitDelta>`
Reply:	`<rc> SNRL <on/off>,<LimitDelta>`
Instruction:	`SNRL?`
Reply:	`<rc> SNRL <on/off>,<LimitDelta>`

Description

SNRL enables or disables the not ready generation due to the approch of a bottle counter to the limit set for this counter. The feedback event mechanism of the counter is not affected by SNRL. Each bottle counter not ready is represented also as not ready bit, see instruction ACT:NRDY?.

A not ready is generated when all the following cases are true:
a) The function is enabled (SNRL 1,x)
b) The limit of the bottle counter is > 0 (see DIAG:CLMT name, limit)
c) The resettable part of the counter + LimitDelta > counter limit
d) The composition channel that corresponds to the counter is not -1

Parameters

<on/off>	0 for disabled, 1
<LimitDelta>	0 .. signed 32 (units of µl)

Default Values

0 for on/off (disabled)
100000 for LimitDelta (equals 100 ml)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Enable not Ready Generation if 50 ml are left
	Instruction:	`SNRL 1,50000`
	Reply:	`RA 0000 SNRL 1,50000`
(b)	Comment:	Query the current setpoint
	Instruction:	`SNRL?`
	Reply:	`RA 0000 SNRL 1,50000`

Name

BFLW

Periodic Broadcast of Flow Value via the CAN Bus. (obsolete)

Synopsis

Instruction:	`BFLW <Period>`
Reply:	`<rc> BFLW <Period>`
Instruction:	`BFLW?`
Reply:	`<rc> BFLW <Period>`

Description

The BFLW instruction defines a period value for periodical broadcast of the actual flow value via the CAN bus. This allows other CAN Cluster members to implement flow dependant functionalities (for example: to readjust the controlling temperature for the column, as done by G1316A). In a cluster with multiple pumps this broadcast becomes ambigous if defined for multiple pumps.

Parameters

<Period	0 .. 100000 seconds, 0 means off

Default Values

0 for off

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Define a 30 second period for flow update
	Instruction:	`BFLW 30`
	Reply:	`RA 0000 BFLW 30`
(b)	Comment:	Query the current setpoint
	Instruction:	`BFLW?`
	Reply:	`RA 0000 BFLW 30`

Name

PISY

Synchronization of pistos (not part of the default behaviour)

Synopsis

Instruction:	`PISY <Piston>`
Reply:	`<rc> PISY <Piston>`
Instruction:	`PISY?`
Reply:	`<rc> PISY <Piston>`

Description

Defines the synchronisation of the pistons for the binary pump.

REMARK: Not for normal operation

Parameters

<Piston>	0,1

Default Values

1 for channel B piston

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set piston
	Instruction:	`PISY 0`
	Reply:	`RA 0000 PISY 0`
(b)	Comment:	Query the current setpoint
	Instruction:	`PISY?`
	Reply:	`RA 0000 PISY 0`

Name

STSY

Synchronization of piston position with start (not part of the default behaviour)

Synopsis

Instruction:	`STSY <Phase>,<Piston>`
Reply:	`<rc> STSY <Phase>,<Piston>`
Instruction:	`STSY?`
Reply:	`<rc> STSY <Phase>,<Piston>`

Description

STSY defines a phase and a piston for synchronization of the time table start.

REMARK: Not for normal operation.

Parameters

<Phase>

<Piston>

Default Values

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Set values
	Instruction:	STSY 25,0
	Reply:	RA 0000 STSY 25,0
(b)	Comment:	Query the values
	Instruction:	STSY?
	Reply:	RA 0000 STSY 25,0

Name

SWPD

Set duration for sealwash pump one shot operation

Synopsis

Instruction:	`SWPD <Time>`
Reply:	`<rc> SWPD <Time>`
Instruction:	`SWPD?`
Reply:	`<rc> SWPD <Time>`

Description

Set the duration of a single sealwash operation. This sysvar is used when instruction SWP 1 is executed.

Parameters

<Time>	0 .. 10.0 [minutes]

Default Values

1.0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set one shot sealwash duration to 8.4 minutes
	Instruction:	`SWPD 8.4`
	Reply:	`RA 0000 SWPD 8.4`
(b)	Comment:	Query the current setpoint
	Instruction:	`SWPD?`
	Reply:	`RA 0000 SWPD 8.4`

Name

SWPP

Set period and active time for sealwash pump periodical operation

Synopsis

Instruction:	`SWPP <Period>,<Active time`
Reply:	`<rc> SWPP <Period>,<Active time>`
Instruction:	`SWPP?`
Reply:	`<rc> SWPP <Period>,<Active time>`

Description

Set period and active time for sealwash pump periodical operation. Sealwashing is repeated every <Period> minutes for a duration of <Active time> minutes, where <Active time> is limited to 70% of the period. This sysvar is used when instruction SWP 2 is executed.

Parameters

<Period>	0.1 .. 99.0	[minutes]
<Active time>	0.1 .. 7.0	[minutes], limited to 70% of <Period>.

Default Values

1.0, 0.2 Sealwash pump turned on once per minute for 12 seconds.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set one shot sealwash duration to 8.4 minutes
	Instruction:	`SWPP 10.0,2.0`
	Reply:	`RA 0000 SWPP 10.0,2.0`
(b)	Comment:	Reply error due too long active time
	Instruction:	`SWPP 3.0,2.5`
	Reply:	`RE 0502`

19.3. Pumps Control Commands

Name

PUMP
Turn pump to on,off or standby

Synopsis

Instruction: PUMP <state>
Reply: <rc> PUMP <state>

Description

The PUMP instruction changes the pump state to either off, on or standby. After power on the pump is always in the 'Off' state. Before flow can be delivered the pump drive assembly needs to find the reference position for guaranteed reproducible operation (Initiated either with pump 1 or pump 2. The standby mode is like a flow zero mode, the pump can at once start delivering flow. During standby mode the module shows 'ready', during 'Off' and till 'On' mode is reached the module shows 'not ready'.

Parameters

<state> 0 for Off, 1 for On, 2 for Standby

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 2032 pump on rejected while leak or shutdown or permanent erro or license code

Examples

(a) Comment: Turn on pump
Instruction: PUMP 1
Reply: RA 0000 PUMP 1

Name

SWP

Sealwash pump turn on to single mode,periodic mode or turn off

Synopsis

Instruction: `SWP <state>`

Reply: `<rc> SWP <state>`

Description

The optional sealwash pump may be controlled via the SWP instruction.Instruction call results in no reply error if sealwash pump is not present.

Parameters

`<state>`	0	turns off sealwash operation.
	1	turns on sealwash pump for the time specified by the SWPD parameter.
	2	turns on sealwash pump for periodical operation as specified by the SWPP parameter.

Default Values

state 0, sealwash pump is off.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: Turn on SWP for single shot sealwashing

Instruction: `SWP 1`

Reply: `RA 0000 SWP 1`

Name

CTAN

Solvent Name of Channel A

Synopsis

Instruction:	`CTAN <name>`
Reply:	`<rc> CTAN <name>`
Instruction:	`CTAN?`
Reply:	`<rc> CTAN <name>`

Description

This instruction sets and queries the name of the solvent defined for channel A.

Parameters

`<name>`	string, maximum 32 characters

Default Values

"undefined"

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set name of solvent A
	Instruction:	`CTAN "Water with 5% ACN"`
	Reply:	`RA 0000 CTAN "Water with 5% ACN"`
(b)	Comment:	query
	Instruction:	`CTAN?`
	Reply:	`RA 0000 CTAN "Water with 5% ACN"`

Name

`CTBN`

Solvent Name of Channel A

Synopsis

Instruction:	`CTBN <name>`
Reply:	`<rc> CTAN <name>`
Instruction:	`CTBN?`
Reply:	`<rc> CTBN <name>`

Description

This instruction sets and queries the name of the solvent defined for channel B.

Parameters

`<name>`	string, maximum 32 characters

Default Values

"undefined"

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set name of solvent B
	Instruction:	`CTBN "Water with 5% ACN"`
	Reply:	`RA 0000 CTBN "Water with 5% ACN"`
(b)	Comment:	query
	Instruction:	`CTBN?`
	Reply:	`RA 0000 CTBN "Water with 5% ACN"`

19.4. Pumps Actuals

Name

ACT:PRES?
Actual Pressure Value

Synopsis

Instruction: `ACT:PRES?`
Reply: `<rc> ACT:PRES? <pressure>`

Description

The ACT:PRES? instruction queries for the last measured pressure value. The pressure value is periodically measured.

Parameters

<pressure> -5.00 .. 450.00 bar, (-5.00 .. 720.00 bar for G1312B)

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual pressure
Instruction: `ACT:PRES?`
Reply: `RA 0000 ACT:PRES? 381.64`

Name

ACT:FLOW?

Actual flow value

Synopsis

Instruction: ACT:FLOW?

Reply: <rc> ACT:FLOW? <flow value>

Description

The ACT:FLOW? instruction queries for the actual flow value.

Parameters

<flow value> 0.000 .. 10.000 [ml/min], (0 .. 5.000 for G1312B)

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query actual flow

Instruction: ACT:FLOW?

Reply: RA 0000 ACT:FLOW? 0.230

Name

ACT:COMP?
Actual composition value

Synopsis

Instruction: `ACT:COMP?`
Reply: `<rc> ACT:COMP? <%A>,<%B>,<%C>,<%D>`

Description

The ACT:COMP? instruction queries for the actual composition. In case a channel is set to 'Off' a value of -1 is returned. The instruction is the same for quaternary and binary pump. For the binary pump %C and %D are returned as 0.

Parameters

<%A>	-1, 0.00 .. 100 [%], -1 for channel off
<%B>	-1, 0.00 .. 100 [%], -1 for channel off
<%C>	-1, 0.00 .. 100 [%], -1 for channel off
<%D>	-1, 0.00 .. 100 [%], -1 for channel off

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual composition
Instruction: `ACT:COMP?`
Reply: `RA 0000 ACT:COMP? 0,20,30,50`

Name

`ACT:RIPL?`
Actual Pressure Ripple

Synopsis

Instruction: `ACT:RIPL?`
Reply: `<rc> ACT:RIPL? <%pressure>`

Description

The ACT:RIPL? instruction queries for the actual ripple calculation. The pressure is monitored continously and kept in RAM for evaluation with each stroke. The ripple value is calculated as the mean value of the last five percent values calculated for the strokes. A pressure % value is calculated as (2 * (last pressure value of last stroke - first pressure value of current stroke) / (first + last).

Parameters

<%pressure> 0 .. 10.00 [%]

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual ripple
Instruction: `ACT:RIPL?`
Reply: `RA 0000 ACT:RIPL? 0.3`

Name

ACT:SSVA?

Actual Position of Sovent Selcetion Valve

Synopsis

Instruction: ACT:SSVA?

Reply: <rc> ACT:SSVA? <position>

Description

The ACT:SSVA? instruction queries for the actual position of the left side of the solvent selection valve assembly.

Parameters

<position> 0 for channel A1, 1 for channel A2

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual channel A position

Instruction: ACT:SSVA?

Reply: RA 0000 ACT:SSVA? 1

Name

ACT:SSVB?
Actual Position of Sovent Selcetion Valve

Synopsis

Instruction: ACT:SSVB?
Reply: <rc> ACT:SSVB? <position>

Description

The ACT:SSVB? instruction queries for the actual position of the right side of the solvent selection valve assembly.

Parameters

<position> 0 for channel B1, 1 for channel B2

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual channel B position
Instruction: ACT:SSVB?
Reply: RA 0000 ACT:SSVB? 1

Name

`ACT:WPRS?`
Actual Time of Wait for Pressure Function

Synopsis

Instruction: `ACT:WPRS?`
Reply: `<rc> ACT:WPRS? <time>`

Description

The ACT:WPRS? instruction queries for the actual time till a time out error occurs of the 'wait for pressure function'. In case the given pressure is not reached till the time is counted back to 0 the error event EE 2053 is generated.

Parameters

<time> 0.00 .. 99999.99 minutes

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual wait time
Instruction: `ACT:WPRS?`
Reply: `RA 0000 ACT:WPRS? 8.22`

Name

`ACT:WVOL?`

Actual Time of Wait for Volume Function

Synopsis

Instruction: `ACT:WVOL?`

Reply: `<rc> ACT:WVOL? <time>`

Description

The ACT:WVOL? instruction queries for the actual time till a time out error occurs of the 'wait for volume function'. In case the given pressure is not reached till the time is counted back to 0 the error event EE 2053 is generated.

Parameters

<time> 0.00 .. 99999.99 minutes

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual wait time

Instruction: `ACT:WVOL?`

Reply: `RA 0000 ACT:WVOL 2.12`

Name

ACT:WSUC?

Actual Time of Wait for TakeIn Function

Synopsis

Instruction: `ACT:WSUC?`

Reply: `<rc> ACT:WSUC? <time>`

Description

The ACT:WSUC? instruction queries for the actual time till a time out error occurs of the 'wait for takeIn function'. In case the take in phase (of the first piston) is not reached till the time is counted back to 0 the error event EE 2053 is generated.

Parameters

<time> 0.00 .. 99999.99 minutes

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual wait time

Instruction: `ACT:WSUC?`

Reply: `RA 0000 ACT:WSUC 2.12`

Name

ACT:WDEL?

Actual Time of Wait for Deliver Function

Synopsis

Instruction: `ACT:WDEL?`

Reply: `<rc> ACT:WDEL? <time>`

Description

The ACT:WDEL? instruction queries for the actual time till a time out error occurs of the ’wait for deliver function’. In case the deliver phase (of the first piston) is not reached till the time is counted back to 0 the error event EE 2053 is generated.

Parameters

<time> 0.00 .. 99999.99 minutes

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual wait time

Instruction: `ACT:WDEL?`

Reply: `RA 0000 ACT:WDEL 2.12`

Name

`ACT:PIST?`
Actual Piston Movements

Synopsis

Instruction:	`ACT:PIST?`
Reply:	`<rc> ACT:PIST? <channelA>,<channelB>`

Description

The ACT:PIST? instruction queries for the actual directions the first piston of channel A and channel B are moving.

Parameters

<channelA>	0 for not applicable, 1 for the deliver phase, 2 for the takein phase
<channelB>	0 for not applicable, 1 for the deliver phase, 2 for the takein phase

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	query the actual piston movements
	Instruction:	`ACT:PIST?`
	Reply:	`RA 0000 ACT:PIST? 1,2`

Name

`ACT:PUMP?`

Actual state of the Pump Drive

Synopsis

Instruction:	`ACT:PUMP?`
Reply:	`<rc> ACT:PUMP? <state>`

Description

The ACT:PUMP? instruction queries for the actual state of the pump drive assemblies.

Parameters

<state>	0: off, 1: on, 2: standby, 3: init

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	query the actual drive states
	Instruction:	`ACT:PUMP`
	Reply:	`RA 0000 ACT:PUMP? 1`

Name

`ACT:SWPM?`

Actual Mode of the Sealwash Pump.

Synopsis

Instruction: `ACT:SWPM?`

Reply: `<rc> ACT:SWPM? <mode>`

Description

The ACT:SWPM? instruction queries for the actual mode of the sealwash pump, as set with the SWP instruction. When single mode was entered, this actual returns "single mode" while the sea-wash pump is running and "off" after single mode has finished automatically.

Parameters

<mode> 0: off, 1: single, 2: periodic

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the sealwash pump mode

Instruction: `ACT:SWPM?`

Reply: `RA 0000 ACT:SWPM 2`

Name

ACT:TYP?
Actual Type of Pump

Synopsis

Instruction: `ACT:TYP?`
Reply: `<rc> ACT:TYP? <type>`

Description

The ACT:TYP? instruction queries for the actual type of pump according to the evaluated hardware.

Parameters

<type>	0 for binary, 1 for quaternary, 2 for isocratic, 3 for binary with SSV, 4 n/a

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual composition
Instruction: `ACT:TYP?`
Reply: `RA 0000 ACT:TYP? 3`

Name

ACT:HIPR?
Actual High Pressure Limit

Synopsis

Instruction: `ACT:HIPR?`
Reply: `<rc> ACT:HIPR? <pressure limit>`

Description

The ACT:HIPR? instruction queries for the actual value of the high pressure limit. This value may be different from the setpoint if used during a timetable execution. Timetable entries are interpolated linearily.

Parameters

<pressure limit> 0 .. 400, (0 .. 600 for G1312B)

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual high pressure limit
Instruction: `ACT:HIPR?`
Reply: `RA 0000 ACT:HIPR? 233`

Name

ACT:PMCH?
Actual Primary Channel

Synopsis

Instruction: `ACT:PMCH?`
Reply: `<rc> ACT:PMCH? <channel>`

Description

The ACT:PMCH? instruction queries for the actual primary channel. In case the primary channel is set to 'automatic mode', the channel with the biggest composition part will be choosen automatically (not during timetable execution)

Parameters

<channel> 0 for channel A, 1 for Channel B, 2 for Channel C, 3 for Channel D

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual primary channel
Instruction: `ACT:PMCH?`
Reply: `RA 0000 ACT:PMCH? 3`

Name

ACT:STRK?
Actual Stroke Value of Left Drive

Synopsis

Instruction: `ACT:STRK?`
Reply: `<rc> ACT:STRK? <stroke value>`

Description

The ACT:STRK? instruction queries for the actual stroke value. In case the stroke setpoint is set to 'automatic mode' the stroke value is calculated according to the actual flow and pump type. The 'automatic mode' is also the lower limit for a settable stroke: allthough the setpoint for lower stroke values is accepted (flow might change during timetable), the calculated value is the actually used one. If the setpoint is higher then the 'automatic', the setpoint is the actually used.

Parameters

<stroke value> 20.0 .. 100 [micro liters]

Default Values

..

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual stroke value
Instruction: `ACT:STRK?`
Reply: `RA 0000 ACT:STRK? 62.4`

Name

ACT:CSTR?

Actual Stroke Value of Right Drive

Synopsis

Instruction: `ACT:CSTR?`

Reply: `<rc> ACT:CSTR? <stroke value>`

Description

The ACT:CSTR? instruction queries for the actual stroke value. In case the stroke setpoint is set to 'automatic mode' the stroke value is calculated according to the actual flow and pump type. The 'automatic mode' is also the lower limit for a settable stroke: allthough the setpoint for lower stroke values is accepted (flow might change during timetable), the calculated value is the actually used one. If the setpoint is higher then the 'automatic', the setpoint is the actually used.

Parameters

<stroke value> 20.0 .. 100 [micro liters]

Default Values

..

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual stroke value

Instruction: `ACT:CSTR?`

Reply: `RA 0000 ACT:STRK? 98.2`

Name

ACT:OPRS?

Actual Operating Pressure

Synopsis

Instruction: `ACT:OPRS?`

Reply: `<rc> ACT:OPRS <pressure>`

Description

Queries the actual operating pressure value.

Parameters

<pressure> 0 .. 600.00 bar

Default Values

..

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual operating pressure value

Instruction: `ACT:OPRS?`

Reply: `RA 0000 ACT:OPRS 580.00`

19.5. Pumps Rawdata File Description

This description is a supplement to: Rawdata File on page 253.

The Agilent 1200 PUMPs generates up to 3 (6, 7) signals (depending on pump type). The rawdata file can store these signals as records RA, RB, RC, RD, RE, RF and RG. To select, which signal should be stored, see instruction RAWS on page 258. For selection of the data representation (decimal, hexadecimal or decimal) see instruction RAWF on page 259.

signal id	Isocratic Pump	Quarternary Pump	Binary Pump Binary Pump SL
RA	pressure [1/100 bar]	pressure [1/100 bar]	pressure [1/100 bar]
RB	flow [ul]	flow [ul]	flow [ul]
RC	direction of piston [0 or 1]	%A [1/10%]	%A [1/10%]
RD	-	%B [1/10%]	%B [1/10%]
RE	-	%C [1/10%]	direction of piston A [0 or 1]
RF	-	%D [1/10%]	direction of piston B [0 or 1]
RG	-	direction of piston [0 or 1]	-

19.6. Pumps Diagnose Buffers

Name

ADOF (diag)

Close loop offset during deliver phase of channel A

Characteristics

max entries	15
max characters	10
storage	battery buffered
availability	always
entry	twice per second

Contents

Offset

Offset	mean value of the last 4 offset register readings from the 'pump control' chip

Description

Offset values are directly read from the 'square chips'. Twice per second a read sequence is started, consisting of three successive register accesses with 160 micro seconds time delay. The smallest of the three readings is used as half second value.

NameName

ATOF (diag)

Close loop offset during take in phase of channel A

Characteristics

max entries	15
max characters	10
storage	battery buffered
availability	always
entry	twice per second

Contents

Offset

Offset	mean value of the last 4 offset register readings from the ’pump control’ chip

Description

Offset values are directly read from the ’square chips’. Twice per second a read sequence is started, consisting of three successive register accesses with 160 micro seconds time delay. The smallest of the three readings is used as half second value.

Name

BDOF (diag)
Close loop offset during deliver phase of channel B

Characteristics

max entries	15
max characters	10
storage	battery buffered
availability	always
entry	twice per second

Contents

Offset

Offset	mean value of the last 4 offset register readings from the ’pump control’ chip

Description

Offset values are directly read from the ’square chips’. Twice per second a read sequence is started, consisting of three successive register accesses with 160 micro seconds time delay. The smallest of the three readings is used as half second value.

Name

BTOF (diag)

Close loop offset during take in phase of channel B

Characteristics

max entries	15
max characters	10
storage	battery buffered
availability	always
entry	twice per second

Contents

Offset

Offset	mean value of the last 4 half seconds readings from the 'pump control' chip

Description

Offset values are directly read from the 'square chips'. Twice per second a read sequence is started, consisting of three successive register accesses with 160 micro seconds time delay. The smallest of the three readings is used as half second value.

Name

CCHR (diag)

Correction History of the left drive assembly

Characteristics

max entries	20
max characters	10
storage	battery buffered
availability	always
entry	once per piston reversal

Contents

Correction

correction	(ulong) The calculated correction for each stroke

Description

the correction value is calculated for each stroke using the actual pressure, the stroke length, the compressibility of the mobile phase and the elasticity of the pump assy.

Name

CCUR (diag)
Correction History of the right drive assembly

Characteristics

max entries	20
max characters	10
storage	battery buffered
availability	always
entry	once per piston reversal

Contents

Correction

correction	(ulong) The calculated correction for each stroke

Description

the correction value is calculated for each stroke using the actual pressure, the stroke length, the compressibility of the mobile phase and the elasticity of the pump assy.

Name

CHCB (diag)

compressibility calculations of left drive assembly

Characteristics

max entries	10
max characters	8
storage	volatile
availability	always
entry	with each piston reversal

Contents

compressibility

compressibility	integer values

Description

Compressibility values are calculated from the pressure drop during piston reversal. If not in auto-compressibility mode, the last calculated values are entered.

Name

CHVO (diag)

Counter History for mobile phase pumped through Channel A.

Name

CHWC (diag)

Counter History for channel A seal wear.

Name

CIVC (diag)

Counter History for channel B inlet valve switches

Name

COVC (diag)

Counter History for channel B outlet valve switches

Name

OVC (diag)

Counter History for channel A outlet valve switches

Name

CUCB (diag)

compressibility calculations of right drive assembly

Characteristics

max entries	10
max characters	8
storage	volatile
availability	always
entry	with each piston reversal

Contents

compressibility

compressibility	integer values

Description

Compressibility values are calculated from the pressure drop during piston reversal. If not in auto-compressibility mode, the last calculated value is entered.

Name

CUVO(diag)

Counter History for mobile phase pumped through Channel B.

Name

CUWC (diag)

Counter History for channel B seal wear.

Name

EOB (diag)

History of Observation Events

Characteristics

max entries	20
max characters	10
storage	battery buffered
availability	always
entry	stochastic (with each observation)

Contents

time, event number, event parameter

time	seconds since 1970
event number	see seperate document for event descriptions
event parameter	(ulong) associated value to the event number

Description

The pressure profile during each stroke is evaluated. The results can be gas bubble problems or leakages. Other observations are changes of parameters that are relevant for the pressure profile. e.g. FLOW, CBLT, COMP, ON/OFF. The data used for the profile evaluation is stored in the SYDA buffer.

Name

`GCKA (diag)`

Counter History for switches of MCGV valve for Channel A.

Name

`GCKB (diag)`

Counter History for switches of MCGV valve for Channel B.

Name

`GCKC (diag)`

Counter History for switches of MCGV valve for Channel C.

Name

`GCKD (diag)`

Counter History for switches of MCGV valve for Channel D.

Name

`IVC (diag)`

Counter History for channel A inlet valve switches

Name

`LITR (diag)`

Counter History for totally pump volume

Name

MAH (diag)

hybrid temperature reading of left motor circuit

Characteristics

max entries	40
max characters	10
storage	volatile
availability	always
entry	once every 0.2 seconds (approximately)

Contents

temperature

temperature	temperature values with 0.01' resolution

Description

Readings directly from the MUX/ADC. Adc counts are converted to temperature values.

Name

MAHC (diag)

hybrid temperature reading of right motor circuit

Characteristics

max entries	40
max characters	10
storage	volatile
availability	always
entry	once every 0.2 seconds (approximately)

Contents

temperature

temperature	temperature values with 0.01 ’ resolution

Description

Readings directly from the MUX/ADC. Adc counts are converted to temperature values.

Name

MAUX (diag)
ADC readings of auxilliary channel

Characteristics

max entries	40
max characters	10
storage	volatile
availability	always
entry	once every 0.2 seconds (approximately)

Contents

Counts

coutns	counts from 0 to 4096

Description

Readings directly from the MUX/ADC. Adc counts are not converted.

Name

MODF (diag)

Flow Values of MODulated Flow

Characteristics

max entries	200
max characters	10
storage	volatile
availability	always
entry	once every 0.1 seconds

Contents

Flow

flow	0 .. 100% of the actual flow

Description

The values are only the amplitude of the sinusoidal modulated flow. The real flow is FLOW + mod.

Name

MVC (diag)

voltage control readings of left motor circuit

Characteristics

max entries	40
max characters	10
storage	volatile
availability	always
entry	once every 0.2 seconds (approximately)

Contents

ADCCounts

ADCCounts	0 .. 4096 ==> 0 .. 5V

Description

Readings directly from the MUX/ADC.

Name

MVCC (diag)

voltage control readings of right motor circuit

Characteristics

max entries	40
max characters	10
storage	volatile
availability	always
entry	once every 0.2 seconds (approximately)

Contents

ADCCounts

ADCCounts	0 .. 4096 ==> 0 .. 5V

Description

Readings directly from the MUX/ADC.

Name

PCHR (diag)

Stroke History of the left drive assembly

Characteristics

max entries	5
max characters	37
storage	volatile
availability	always
entry	once per piston reversal

Contents

Time, TimeOfGrad, Stroke, Correction, Direction

Time	The time it took to stroke, in units of 12,5 ms
TimeOfGrad	The actual time of the gradient, in units of 12.5 ms
Stroke	The actual stroke value, in micro liters
Correction	The compensation value calculated for the stroke
Direction	1 for takein, 0 for deliver

Description

..

Name

PCUR (diag)

Stroke History of the right drive assembly

Characteristics

max entries	5
max characters	37
storage	volatile
availability	always
entry	once per piston reversal

Contents

Time, TimeOfGrad, Stroke, Correction, Direction

Time	The time it took to stroke, in units of 12,5 ms
TimeOfGrad	The actual time of the gradient, in units of 12.5 ms
Stroke	The actual stroke value, in micro liters
Correction	The compensation value calculated for the stroke
Direction	1 for takein, 0 for deliver

Description

..

Name

PERC (diag)
Percentage History of the quaternary pump

Characteristics

max entries	10
max characters	37
storage	battery buffered
availability	always
entry	once per piston reversal

Contents

PercA, PercB, PercC, PercD

PercX	The composition percentage of the quaternary channel

Description

For each stroke a new composition is calculated in case a composition gradient is in progress.

Name

PERP (diag)

Strokeposition History of the quaternary pump

Characteristics

max entries	10
max characters	48
storage	battery buffered
availability	always
entry	once per piston reversal

Contents

Pos1, Pos2, Pos3, Pos4, Stroke length

Pos1 .. Pos4	The values of the valve pulse registers of the 'square chip'
Stroke length	The value of the stroke register of the 'square chip'

Description

The composition percentages are converted into positions for valve pulses.

Name

PRV (diag)

pressure data collected during piston reversal (left)

Characteristics

max entries	30
max characters	6
storage	volatile
availability	always
entry	with each piston reversal, see description

Contents

pressure

pressure	integer values

Description

This data collection is the basis for ripple and compressibility calculations. Data collection starts with each piston reversal. The first data point is the mean value of the previous calculation, the second data point is the mean value of the current values. The next 10 pressure values are acquired with a time interval of 2.5 ms, the rest of the values are acquired with 25 ms between. The maximum amount of data is 30. With extrem short stroke and extrem high flow there will be less values acquired. This buffer is reset to empty befor entering new pressure values.(with each piston reversal)

See also Buffer RIP and RIPC.

Name

PRVC (diag)

pressure data collected during piston reversal (right)

Characteristics

max entries	30
max characters	6
storage	volatile
availability	always
entry	with each piston reversal, see description

Contents

pressure

pressure	integer values

Description

This data collection is the basis for ripple and compressibility calculations. Data collection starts with each piston reversal. The first data point is the mean value of the previous calculation, the second data point is the mean value of the current values. The next ten pressure values are acquired with a time interval of 2.5 ms, the rest of the values are acquired with 25 ms between. The maximum amount out data is 30, with extrem short stroke and extrem high flow there will be less values acquired. This buffer is reset to empty befor entering new pressure values. (with each piston reversal)

See also Buffer RIP and RIPC.

Name

RDTA (diag)
history of ripple data values

Characteristics

max entries	10
max characters	46
storage	volatile
availability	always
entry	with each piston reversal

Contents

Ripple Value, Id, FirstPressure, Val0, Val1, Val2, LastPressure, Diff

Ripple Value	percent value of pressure ripple caused by piston reversal
Id	
FirstPressure	the first pressure reading after reversal
Val0 .. Val2	Val0 is the pressure value taken after 100ms after reversal Val1 and Val2 are taken at half of the time till end of stroke
LastPressure	the last pressure reading of the stroke
Diff	are the last 12.5 ms ticks between Val1 and Val2

Description

..

Name

RIP (diag)
history of ripple calculations

Characteristics

max entries	10
max characters	8
storage	battery buffered
availability	always
entry	with each piston reversal

Contents

Ripple Value

Ripple Value	percent value of pressure ripple caused by piston reversal

Description

..

Name

RIPC (diag)
history of ripple calculations

Characteristics

max entries	10
max characters	8
storage	none volatile
availability	always
entry	with each piston reversal

Contents

Ripple Value

Ripple Value	percent value of pressure ripple caused by piston reversal

Description

..

Name

SSVA (diag)
Counter History for switches of solvent selection valve A1/A2

Name

SSVB (diag)
Counter History for switches of solvent selection valve B1/B2

Name

SYDA (diag)
history of pressure symptom data

Characteristics

max entries	31
max characters	15
storage	volatile
availability	always
entry	once per piston reversal

Contents

Id of datum, value

Id of datum	0: left deliver phase
	1: left deliver, pressure value0
	2: left deliver, pressure value1
	3: left deliver, pressure value2
	4: left deliver, last pressure value
	10: left takein phase
	11: left takein, pressure value0
	12: left takein, pressure value1
	13: left takein, pressure value2
	14: left takein, last pressure value
	20: right deliver phase
	21: right deliver, pressure value0
	22: right deliver, pressure value1
	23: right deliver, pressure value2
	24: right deliver, last pressure value
	30: right takein phase
	31: right takein, pressure value0
	32: right takein, pressure value1
	33: right takein, pressure value2
	34: right takein, last pressure value
	40: no symptom
value	in case of ids 0,10,20, 30 and 40: actual ticks of 12.5 ms (since turn on!)
	for other ids: pressure value in units ADC counts (0.. 4096 => 0 .. 440bar)

Description

With the detection of a new piston direction, the pressure values of the previous phase are put into the buffer. The entries always complete (allway 5 successive ids). Pressure value0 is taken 100 ms after the piston reversal, values 1 and 2 are always equidistant.

Name

VMCG (diag)
Valve test switches during startup

Characteristics

max entries	12
max characters	45
storage	volatile
availability	always
entry	once per start up

Contents

Off status, On status, Off bit, On bit, Ok

Off status	hareware reading from MUSIC-GPIO-IN-A
On status	hareware reading from MUSIC-GPIO-IN-A
Off bit	1 if bit for valve
On bit	1 if not bit valve
Ok	1 if Off bit and On bit are 1

Description

During the start up period the MCGV valve or SSV valve are all switched one after the other. This is the typical click-click-click.. of the quaternary pump or binary pump with SSV. The hardware is monitored during the witches, the results of the switches are given above. Only if all switches are done with Ok == 1 the appropriate option/pump type is set. If all switches are done with Ok == 0, the option is considered to be not present, If only a part of the switches failes an appropriate error event is generated.

19.7. Pumps Diagnose Counters

Name

BOTA (diag)

Volume pumped thrugh channel A

Increments by %A of the stroke volume with each deliver phase of the piston. In case of a binary pump the volume through the left pump assembly is added, with a solvent selection valve volume through A1 is added

storage	battery buffered
Event	EF 2083
History	none

Name

BOTB (diag)

Volume pumped thrugh channe B

Increments by %B of the stroke volume with each deliver phase of the piston. In case of a binary pump the volume through the right pump assembly is added, with a solvent selection valve, the volume through A2 is added.

storage	battery buffered
Event	EF 2084
History	none

Name

BOTC (diag)

Volume pumped thrugh channel C

Increments by %C of the stroke volume with each deliver phase of the piston. In case of a binary pump with a solvent selection valve, the volume through B1 is added.

storage	battery buffered
Event	EF 2085
History	none

Name

BOTD (diag)

Volume pumped thrugh channel C

Increments by %D of the stroke volume with each deliver phase of the piston. In case of a binary pump with a solvent selection valve, the volume through B2 is added.

storage	battery buffered
Event	EF 2086
History	none

Name

CHVO (diag)

pumped volume of the left pump assembly

Increments by the stroke volume with each piston reversal.

storage	none volatile
Event	EF 2073
History	5

Name

CHWC (diag)

wear of left pump assembly

Increments by (pressure x flow of left pump assembly) every second. Unit is ml*bar. A pressure value below 20 bar is set to 20 bar.

storage	none volatile
Event	EF 2070
History	2

Name

CIVC (diag)

counter for inlet valve switches of right pump assembly

Increments by 1 with each stroke cycle.

storage	none volatile
Event	EF 2078
History	5

Name

COVC (diag)

counter for outlet valve switches of right pump assembly

Increments by 1 with each stroke cycle.

storage	none volatile
Event	EF 2076
History	5

Name

CNPI (diag)

Increments by one with each take in phase of right pump assembly

storage	battery buffered
Event	none
History	none

Name

CPIS (diag)

Increments by one with each deliver phase of right pump assembly

storage	battery buffered
Event	none
History	none

Name

CPOV (diag)

Overlap counter for pressure observation data (binary pump).

Increments by one if the second drive assembly reversed the piston direction within 200 ms after start of deliver phase.In this case auto compensation and ripple calculations are skipped.

storage	volatile
Event	none
History	none

Name

CUVO (diag)

pumped volume of the right pump assembly

Increments by the stroke volume with each piston reversal.

storge	none volatile
Event	EF 2074
History	5

Name

CUWC (diag)

wear of right pump assembly

Increments by (pressure x flow of right pump assembly) every second.A pressure value below 20 bar is set to 20 bar.

storage	none volatile
Event	EF 2071
History	2

Name

GCK (diag)

Counts the interrupts for MCGV switches

storage	battery buffered
Event	none
History	none

Name

GCKA (diag)

MCGV switches of valve for channel A

storage	none volatile
Event	EF 2079
History	5

Name

GCKB (diag)

MCGV switches of valve for channel B

storage	none volatile
Event	EF 2080
History	5

Name

GCKC (diag)

MCGV switches of valve for channel C

storage	none volatile
Event	EF 2081
History	5

Name

GCKD (diag)

MCGV switches of valve for channel D

storage	none volatile
Event	EF 2082
History	5

Name

IVC (diag)

counter for inlet valve switches of left pump assembly

Increments by 1 with each stroke cycle.

storage	none volatile
Event	EF 2077
History	5

Name

LADC (diag)

Counter of failed ADC readings.

With extrem processor load, interrupts to serve the ADC may be lost.

This is tolerated to about 10 successive failures before an error is generated.

storage	volatile
Event	none
History	none

Name

LITR (diag)

Sums up the total volume pumped

Increments by one with each pumped ml.

storage	none volatile
Event	2072
History	10

Name

NPIS (diag)

Increments by one with each take in phase of left pump assembly

storage	battery buffered
Event	none
History	none

Name

OVC (diag)

counter for outlet valve switches of left pump assembly

Increments by 1 with each stroke cycle.

storage	none volatile
Event	EF 2075
History	5

Name

PIS (diag)

Increments by one with each deliver phase of left pump assembly

storage	battery buffered
Event	none
History	none

Name

RADC (diag)

Counts restarts of ADC timer.

If ADC timer interrupts are lost for more than 5 times, the timer is restarted and the counter is incremented by 1.

storage	volatile
Event	none
History	none

Name

SSVA (diag)

Solvent Selection Valve A1/A2 Counter

Increments by one with each switching of solvent selection valve A1/A2.

storage	none volatile
Event	EF 2087
History	5

Name

SSVB (diag)

Solvent Selection Valve B1/B2 Counter

Increments by one with each switching of solvent selection valve B1/B2.

storage	none volatile
Event	EF 2088
History	5

19.8. Pump Instructions to build Applications

Name

WPRS

Wait Till Pressure value is reached

Synopsis

Instruction:	`WPRS <Pressure>, <time>`
Reply:	`<rc> WPRS <Pressure>, <time>`
Instruction:	`ACT:WPRS?`
Reply:	`<rc> ACT:WPRS <time>`
Instruction:	`STOP:WPRS`
Reply:	`<rc> STOP:WPRS`

Description

The WPRS instruction allows to synchronize any other command to the event that a given pressure value is reached. If the <pressure> value is not reached within the <time> parameter an appropriate error is generated.

STOP:WPRS ends waiting for pressure, but acts as if pressure value was reached. The 'wait pressure' function also reacts to the ABRT instruction.

Parameters

`<Pressure>`	`0 .. 400 [bar], (0 .. 600 bar for G1312B)`
<time>	0 .. 99999.00 [minutes]

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Wait till pressure reaches 380 bar, error after 2 minutes
	Instruction:	`WPRS 380,2`
	Reply:	`RA 0000 WPRS 380.0,2.00`
(b)	Comment:	Query the actual pressure wait time
	Instruction:	`ACT:WPRS?`
	Reply:	`RA 0000 WPRS 1.12`
(c)	Comment:	Stop waiting for pressure
	Instruction:	`STOP:WPRS`
	Reply:	`RA 0000 STOP:WPRS`

Name

WVOL

Wait till a defined Volume has been pumped

Synopsis

Instruction:	`WVOL <Volume>, <time>`
Reply:	`<rc> WVOL <Volume>, <time>`
Instruction:	`ACT:WVOL?`
Reply:	`<rc> ACT:WVOL <time>`
Instruction:	`STOP:WVOL`
Reply:	`<rc> STOP:WVOL`

Description

The WVOL instruction allows to synchronize any other command to the event that a given volume has been pumped. If the <Volume> value is not reached within the <time> parameter an appropriate error is generated.

STOP:WVOL ends waiting for volume, but acts as if the volume had been pumped. The 'wait for valume' function also reacts to the ABRT instruction.

Parameters

<Volume>	0 .. 1000.000 [ml]
<time>	0 .. 99999.00 [minutes]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Wait till 2.345 ml have been pumped, error after 10 minutes
	Instruction:	`WVOL 2.345, 10`
	Reply:	`RA 0000 WVOL 2.345,10.00`
(b)	Comment:	Query the actual volume wait time
	Instruction:	`ACT:WVOL?`
	Reply:	`RA 0000 WVOL 1.12`
(c)	Comment:	Stop waiting for volume
	Instruction:	`STOP:WVOL`
	Reply:	`RA 0000 STOP:WVOL`

Name

WDEL

Wait till Piston reverses to Deliver Phase

Synopsis

Instruction:	`WDEL <left/right>, <time>`
Reply:	`<rc> WDEL <left/right>, <time>`
Instruction:	`ACT:WDEL?`
Reply:	`<rc> ACT:WDEL <time>`
Instruction:	`STOP:WDEL`
Reply:	`<rc> STOP:WDEL`

Description

The WDEL instruction allows to synchronize any other command to the event that the pump drive's piston reverses the direction to 'deliver'. This means the first piston of the drive assembly reverses to deliver. For isocratic and quaternary pumps there is only a left drive assembly. If the reversal has not occured within the <time> parameter an appropriate error is generated.

STOP:WDEL ends waiting for the deliver phase, but acts as if the reversal was done. The 'wait deliver' function also reacts to the ABRT instruction.

Parameters

`<left/right>`	`0 for the left drive assemlby, 1 for the right drive assembly`
<time>	0 .. 99999.00 [minutes]

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Wait for the start of the left deliver phase, error after 0.5 minutes
	Instruction:	`WDEL 0,0.5`
	Reply:	`RA 0000 WDEL 0,0.5`
(b)	Comment:	Query the actual wait time
	Instruction:	`ACT:WDEL?`
	Reply:	`RA 0000 WDEL 0.12`
(c)	Comment:	Stop waiting for the deliver phase
	Instruction:	`STOP:WDEL 0`
	Reply:	`RA 0000 STOP:WDEL 0`

Name

WSUC

Wait till Piston reverses to Take In Phase

Synopsis

Instruction:	`WSUC <left/right>, <time>`
Reply:	`<rc> WSUC <left/right>, <time>`
Instruction:	`ACT:WSUC?`
Reply:	`<rc> ACT:WSUC <time>`
Instruction:	`STOP:WSUC`
Reply:	`<rc> STOP:WSUC`

Description

The WSUC instruction allows to synchronize any other command to the event that the pump drive's piston reverses the direction to 'TakeIn'. This means the first piston of the drive assembly reverses to take in. For isocratic and quaternary pumps there is only a left drive assembly. If the reversal has not occured within the <time> parameter, an appropriate error is generated.
STOP:WSUC ends waiting for the take in phase, but acts as if the reversal was done. The 'wait for takein' function also reacts to the ABRT instruction.

Parameters

`<left/right>`	`0 for the left drive assemlby, 1 for the right drive assembly`
<time>	0 .. 99999.00 [minutes]

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Wait for the start of the left take in phase, error after 0.5 minutes
	Instruction:	`WSUC 0,0.5`
	Reply:	`RA 0000 WSUC 0,0.5`
(b)	Comment:	Query the actual wait time
	Instruction:	`ACT:WSUC?`
	Reply:	`RA 0000 WSUC 0.12`
(c)	Comment:	Stop waiting for the deliver phase
	Instruction:	`STOP:WSUC 0`
	Reply:	`RA 0000 STOP:WSUC 0`

Name

KPRS:STRT

Start Pressure Controlled Delivery

Synopsis

Instruction:	`KPRS:STRT <pressure>, <max flow>, <time>`
Reply:	`<rc> KPRS:STRT <pressure>, <max flow>, <time>`
Instruction:	`KPRS?`
Reply:	`<rc> KPRS:STRT <pressure>, <max flow>, <time>`
Instruction:	`KPRS:STOP`
Reply:	`<rc> KPRS:STOP`

Description

The KPRS:STRT instruction starts delivery at the given (constant) pressure instead of a constant flow. If the <max flow> parameter is specified as a positiv value, a error is generated whenever this flow is exceeded. When specified as a negativ value, this flow will not be exceeded, but not error will be generated.

KPRS:STOP terminates the pressure controlled delivery, that is , the flow value of the method is reactivated and kept constant.

Parameters

<pressure>	-1, 0 .. 400.0 [bar]; -1 means to keep the actual pressure (-1, 0 .. 600.0 bar for G1312B)
<max flow>	-10.000 .. 10.000 [ml/min], (-5.000 .. 5.000 for G1312B)
<time>	0 .. 1,000,000 [seconds]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	keep a pressure of 80 bar with a max flow of 3 ml/min (without error) for 2 minutes
	Instruction:	`KPRS:STRT 80,-3,120`
	Reply:	`RA 0000 KPRS:STRT 80.0, -3.000, 120`
(b)	Comment:	Query the actual params
	Instruction:	`KPRS?`
	Reply:	`RA 0000 KPRS 80.0, 3.000, 120`
(c)	Comment:	Stop pressure control
	Instruction:	`KPRS:STOP`
	Reply:	`RA 0000 KPRS:STOP`

Name

RMPC

Operate on a Composition Ramp (binary pump only)

Synopsis

Instruction:	`RMPC:STRT`
Reply:	`<rc> RMPC:STRT`
Instruction:	`RMPC:STOP`
Reply:	`<rc> RMPC:STOP`
Instruction:	`RMPC:HOLD`
Reply:	`<rc> RMPC:HOLD`
Instruction:	`RMPC:CONT`
Reply:	`<rc> RMPC:CONT`

Description

With the RMPC:STRT instruction a composition ramp is started.For the slope of the ramp, see method parameter RMPC. When the ramp is stopped, the home value of before is restored. The instruction RMPC:HOLD allows to freeze the actual composition value until the instruction RMPC:CONT resumes the ramp or RMPC:STOP aborts the ramp.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	start a composition ramp
	Instruction:	`RMPC:STRT`
	Reply:	`RA 0000 RMPC:STRT`
(b)	Comment:	stop a composition ramp
	Instruction:	`RMPC:STOP`
	Reply:	`RA 0000 RMPC:STOP`
(c)	Comment:	hold a composition ramp
	Instruction:	`RMPC:HOLD`
	Reply:	`RA 0000 RMPC:HOLD`
(d)	Comment:	resume a composition ramp
	Instruction:	`RMPC:CONT`
	Reply:	`RA 0000 RMPC:CONT`

Name

RMPF

Operate on a Flow Ramp

Synopsis

Instruction:	RMPF:STRT
Reply:	<rc> RMPF:STRT
Instruction:	RMPF:STOP
Reply:	<rc> RMPF:STOP
Instruction:	RMPF:HOLD
Reply:	<rc> RMPF:HOLD
Instruction:	RMPF:CONT
Reply:	<rc> RMPF:CONT

Description

With the RMPF:STRT instruction a flow ramp is started.For the slope of the ramp, see method parameter RMPF. When the ramp is stopped, the home value of before is restored. The instruction RMPF:HOLD allows to freeze the actual flow value until the instruction RMPF:CONT resumes the ramp or RMPC:STOP aborts the ramp.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	start a flow ramp
	Instruction:	RMPF:STRT
	Reply:	RA 0000 RMPF:STRT
(b)	Comment:	stop a flow ramp
	Instruction:	RMPF:STOP
	Reply:	RA 0000 RMPF:STOP
(c)	Comment:	hold a flow ramp
	Instruction:	RMPF:HOLD
	Reply:	RA 0000 RMPF:HOLD
(d)	Comment:	resume a flow ramp
	Instruction:	RMPF:CONT
	Reply:	RA 0000 RMPF:CONT

Name

PROF

Turn off pump and sync if pressure is below limit

Synopsis

Instruction: `PROF <pressure>, <time>`

Reply: `<rc> PROF <pressure>, <time>`

Description

The PROF instruction turns of the flow and sends a sync as soon as the actual pressure drops below the <pressure> parameter. If the pressure limit is not reached within <time> the error EE 2053 is generated. To Turn on flow again the instruction FLON must be used.

Parameters

<pressure>	0 to 400.00 bar, warning ! pressure 0 may not be seen due to a misajustment of the pressure sensor!
<time>	0 .. 99999.00 [minutes]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Turn off flow and wait till the pressure drops below 15 bar, error after 1 minute

Instruction: `PROF 15,1`

Reply: `RA 0000 PROF 15,1`

Name

VOFF

Turn off pump after delivering a given volume and sync if pressure is below 5 bar

Synopsis

Instruction: `VOFF <volume>, <time>`

Reply: `<rc> VOFF <volume>, <time>`

Description

The VOFF instruction calculates the volume to deliver as <volume> - 1ul per bar. The pump is turn ed off as soon as the calculated volume is <= 0. The sync is sent as soon as the actual pressure drops below 5 bar. If the pressure limit is not reached within <time> the error EE 2053 is generated. To Turn on the flow again, the instruction FLON must be used.

Parameters

<volume>	0 to 100.000 ml
<time>	0 .. 99999.00 [minutes]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Deliver 0.5 ml, Turn off flow and wait till the pressure drops below 5 bar, error after 1 minute

Instruction: `VOFF 0.5,1`

Reply: `RA 0000 VOFF 0.5,1`

Name

FLON
Turn on flow after PROF or VOFF

Synopsis

Instruction: FLON
Reply: <rc> FLON>

Description

The FLON instruction turns on flow again if flow was turned of with either the PROF of VOFF instruction.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Release the 'FLOW off' state
Instruction: FLON
Reply: RA 0000 FLON

20. Column Compartment (G1316A, G1316B)

20.1. Column Compartment Method Parameter

Name

TEMP
Temperature Setpoint

Synopsis

Instruction:	TEMP <temperature>
Reply:	<rc> TEMP <temperature>
Instruction:	TEMP?
Reply:	<rc> TEMP <temperature>
Instruction:	AT:TEMP <time>, <temperature>
Reply:	<rc> AT:TEMP <time>, <temperature>
Instruction:	AT:TEMP? <time>
Reply:	AT:TEMP <time>,<temperature>
Instruction:	ACT:TEMP?
Reply:	<rc> ACT:TEMP <temperature>

Description

The Temperature setpoint is the temperature that the column compartement keeps constant after beein turned on (see control instruction THRM 1). TEMP? queries the setpoint.
AT:TEMP defines a setpoint change during execution of a time table.
ACT:TEMP? qeueries the actual temperature (even if not controlled). A failed sensor reading returns -274.00
If the column compartment is used in 'combined' mode (see instruction SEP), the TEMP setpoint is also used for the right heat exchanger assembly. Otherwise see RTMP.

Parameters

<temperature>	-274.00 for 'floating/not used'; -5.00 .. 80.00(G1316A) resp. 100.00(G1316B) all in 'C
<time>	0 .. 99999.00 [min]

Default Values

-274.00

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	change setpoint to 12'C
	Instruction:	TEMP 12
	Reply:	RA 0000 TEMP 12.00
(b)	Comment:	Query temperature setpoint
	Instruction:	TEMP?
	Reply:	RA 0000 TEMP 12.00
(c)	Comment:	Query for actual temperature
	Instruction:	ACT:TEMP?
	Reply:	RA 0000 ACT:TEMP 22.43
(d)	Comment:	Change setpoint to 33 'C after 1.5 minutes of timetable execution
	Instruction:	AT:TEMP 1.5, 33
	Reply:	RA 0000 AT:TEMP 1.50,33.00

Name

RTMP

Temperature Setpoint for the Right Heat Exchanger

Synopsis

Instruction:	`RTMP <temperature>`
Reply:	`<rc> TEMP <temperature>`
Instruction:	`RTMP?`
Reply:	`<rc> RTMP <temperature>`
Instruction:	`AT:RTMP <time>, <temperature>`
Reply:	`<rc> AT:RTMP <time>, <temperature>`
Instruction:	`AT:RTMP? <time>`
Reply:	`AT:RTMP <time>,<temperature>`
Instruction:	`ACT:RTMP?`
Reply:	`<rc> ACT:RTMP <temperature>`

Description

The Temperature setpoint is the temperature that the column compartement keeps constant after beein turned on. RTMP? queries the setpoint.

AT:RTMP defines a setpoint change during execution of a time table.

ACT:RTMP? qeueries the actual temperature (even if not controlled). A failed sensor reading returns -274.00

If the column compartment is used in 'combined' mode (see instruction SEP), the TEMP setpoint is also used for the right heat exchanger assembly.

Parameters

<temperature>	-274.00 for 'floating/not used'; -5.00 .. 90.00(G1316A) resp. 100.00(G1316B) all in 'C
`<time>`	0 .. 99999.00 [min]

Default Values

40

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: change setpoint to 12'C
Instruction: `RTMP 12`
Reply: `RA 0000 RTMP 12.00`

(b) Comment: Query temperature setpoint
Instruction: `RTMP?`
Reply: `RA 0000 RTMP 12.00`

(c) Comment: Query for actual temperature
Instruction: `ACT:RTMP?`
Reply: `RA 0000 ACT:RTMP 22.16`

(d) Comment: Change setpoint to 40'C after 2.1 minutes of timetable execution
Instruction: `AT:RTMP 2.1, 40`
Reply: `RA 0000 AT:RTMP 2.10,40.00`

Name

VALV

Switch Column Switching Valve

Synopsis

Instruction:	`VALV <position>`
Reply:	`<rc> VALV <position>`
Instruction:	`VALV?`
Reply:	`<rc> VALV <position>`
Instruction:	`AT:VALV <time>, <position>`
Reply:	`<rc> AT:VALV <time>,<position>`
Instruction:	`ACT:VALV?`
Reply:	`<rc> ACT:VALV <position>`

Description

Switches the Column Switching Valve. This applies for both, 2/6 and 2/10 valves. See also OPT?.. The parameter <position>, if either 0 or 1, defines the position to switch to. Negativ values of <position> are used for special behaviours; see list below. For the actual query the parameter returns the position that was last switched to.

Parameters

position: 0 for port 1->2; 1 for port 1->6 or 1->10 depending on installed valve.
-1: no switch at parameter restoration
-2: toggle at end of run
-3: toggle at end of postrun
-4: toggle at end of analysis
-5: toggle at prepare (see command ATOG, also APG remote prepare pulse outside analysis state)
-6: toggle at begin of run
-7: toggle as Time Table comand

For time tables usage only the values 0, 1 and -7 have got an effect.

Default Values

1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: switch to position 1->6 or 1->10
Instruction: `VALV 1`
Reply: `RA 0000 VALV 1`

(b) Comment: Query actual position
Instruction: `ACT:VALV?`
Reply: `RA 0000 ACT:VALV 1`

(c) Comment: Query setpoint
Instruction: `VALV?`
Reply: `RA 0000 VALV 1`

(d) Comment: Set Timetable Entry: at 2.1 minutes switch to port 1->2
Instruction: `AT:VALV 2.1, 0`
Reply: `RA 0000 AT:VALV 2.1, 0`

Name

LRDD

Sets Temperature Delta for Not Ready Generation for left Heat Exchanger

Synopsis

Instruction:	`LRDD <temperature delta>`
Reply:	`<rc> LRDD <temperature delta>`
Instruction:	`LRDD?`
Reply:	`<rc> LRDD <temperature delta>`

Description

A statechange to not Ready is done as soon as the actual temperature differs from the setpoint for more than specified by LRDD.

Parameters

<temperature delta> 0 .. 100.00 [’]; a value smaller than 0.5 ’ means off.

Default Values

0.8 [’]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set left ready delta
	Instruction:	`LRDD 2`
	Reply:	`RA 0000 LRDD 2.00`
(b)	Comment:	Query left ready delta
	Instruction:	`LRDD?`
	Reply:	`RA 0000 LRDD 2.00`

Name

RRDD

Sets Temperature Delta for Not Ready Message for the Right Heat Exchanger

Synopsis

Instruction:	`RRDD <temperature delta>`
Reply:	`<rc> RRDD <temperature delta>`
Instruction:	`RRDD?`
Reply:	`<rc> RRDD <temperature delta>`

Description

A statechange to not Ready is done as soon as the actual temperature differs from the setpoint for more than specified byRRDD.

Parameters

<temperature delta> 0 .. 100.00 [’]; a value smaller than 0.5 ’ means off.

Default Values

0.8 [’]

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set left ready delta
	Instruction:	`RRDD 2`
	Reply:	`RA 0000 RRDD 2.00`
(b)	Comment:	Query left ready delta
	Instruction:	`RRDD?`
	Reply:	`RA 0000 RRDD 2.00`

Name

SEP
Sets Combined or Seperated Mode

Synopsis

Instruction:	`SEP <combined/seperated>`
Reply:	`<rc> SEP <combined/seperated>`
Instruction:	`SEP?`
Reply:	`<rc> SEP <combined/seperated>`

Description

In Combined mode, both heatexchangers are controlled to the setpoint defined by TEMP. In seperated mode, the setpoint for the left heatexchanger is set by TEMP; for the right heat exchanger by RTMP. This applies also for the 'method turn off: TEMP -274'.

Parameters

<combined/seperated> 0 for combined, 1 for seperated.

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	set seperated mode
	Instruction:	`SEP 1`
	Reply:	`RA 0000 SEP 1`
(b)	Comment:	Query control mode
	Instruction:	`SEP?`
	Reply:	`RA 0000 SEP 1`

Name

LRT

Sets Source for Setpoint to Remote CAN Device

Synopsis

Instruction:	`LRT <local/remote>`
Reply:	`<rc> LRT <local/remote>`
Instruction:	`LRT?`
Reply:	`<rc> LRT <local/remote>`

Description

If set to 'local', the temperature setpoint of the left heat exchanger is defined by the TEMP method parameter. If set to 'remote', the setpoint is queried from the CAN device defined by LRTP. A typical remote source is the optical unit temperature of the detectors

Parameters

<local/remote>	0 for local, 1 for remote.

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	set remote source
	Instruction:	`LRT 1`
	Reply:	`RA 0000 LRT 1`
(b)	Comment:	Query setpoint source
	Instruction:	`LRT?`
	Reply:	`RA 0000 LRT 1`

Name

RRT

Sets Source for Setpoint to Remote CAN Device

Synopsis

Instruction:	`RRT <local/remote>`
Reply:	`<rc> RRT <local/remote>`
Instruction:	`RRT?`
Reply:	`<rc> RRT <local/remote>`

Description

If set to ’local’, the temperature setpoint of the ritht heat exchanger is defined by the TEMP and SEP 0 or by the RTMP method parameter. If set to ’remote’, the setpoint is queried from the CAN device defined by RRTP. A typical remote source is the optical unit temperature of the detectors

Parameters

<local/remote> 0 for local, 1 for remote.

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	set remote source
	Instruction:	`RRT 1`
	Reply:	`RA 0000 RRT 1`
(b)	Comment:	Query setpoint source
	Instruction:	`RRT?`
	Reply:	`RA 0000 RRT 1`

Name

TCOR

Temperature Correction Function

Synopsis

Instruction:	`TCOR <linear>,<offset>`
Reply:	`<rc> TCOR <linear>,<offset>`
Instruction:	`TCOR?`
Reply:	`<rc> TCOR <linear>,<offset>`

Description

This function allowes to recalculate the control temperature T with the formula
T = Setpoint * <linear> + <offset>.

Parameters

linear:	-0.15000 . . +0.15000
offset:	-5.00 .. +5.00

Default Values

linear:	0
offset:	0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set correction to -2K at 10'C and +6K at 50'C
	Instruction:	`TCOR 1.2,-4`
	Reply:	`RA 0000 TCOR 1.2,-4`
(b)	Comment:	Query the correction parameters
	Instruction:	`TCOR?`
	Reply:	`RA 0000 TCOR 1.2,-4`

20.2. ColumnCompartement System Parameter

Name

CNDS

Generate Condensation Info Event instead of Leak Error Event

Synopsis

Instruction:	`CNDS <off/on>`
Reply:	`<rc> CNDS <off/on>`
Instruction:	`CNDS?`
Reply:	`<rc> CNDS <off/on>`

Description

The CNDS instruction provides the possibility to use the leak sensos as condensation sensor. Condesnation is only expected in cases of cooling therfore the function will only work if the setpoint is below the ambient temperature.

Parameters

<off/on>	0 for off (leak: see EE 0064), 1 for on (condensation: see EV 2835)

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	transform leak error event to condensation event
	Instruction:	`CNDS 1`
	Reply:	`RA 0000 CNDS 1`
(b)	Comment:	query setpoint of CNDS
	Instruction:	`CNDS?`
	Reply:	`RA 0000 CNDS 1`

Name

THON

Automatic setpoint activation after power on.

Synopsis

Instruction:	`THON <on/off>`
Reply:	`<rc> THON <on/off>`
Instruction:	`THON?`
Reply:	`<rc> THON <on/off>`

Description

With THON 1 the column compartment activates the actual setpoints as soon as the instrument is power cycled. The actual setpoint is the one that was active before power off.

Parameters

<on/off>	0 for off, 1 for on after power

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set automatic setpoint control after next power cycle
	Instruction:	`THON 1`
	Reply:	`RA 0000 THON 1`
(b)	Comment:	query
	Instruction:	`THON?`
	Reply:	`RA 0000 THON 1`

Name

RCAL

Calibration temperature of the right heat exchanger assembly

Synopsis

Instruction:	`RCAL <calib temp>`
Reply:	`<rc> RCAL <calib temp>`
Instruction:	`RCAL?`
Reply:	`<rc> RCAL <calib temp>`

Description

The calibration temperature is used to calculate a permanent offset to the measured temperature. The delta between the <calib temp> and the actual verification temperature (see RVFY?) gives the offset. The calibration is assumed to be done at exactly 36'C. See also instructions CSTR and CSTP to start and stop calibration mode, and the appropriate statechange events. The Info event EV 2840 indicates that the calibration setpoint (36'C) is reached closely enough to except the calibration temperature. See instruction CAL? for query of the active offsets.

Parameters

<calib temp> 34.00 .. 38.00 'C

Default Values

36

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	initiate calibration calculation to offset for 36.22 'C
	Instruction:	`RCAL 36.22`
	Reply:	`RA 0000 RCAL 36.22`
(b)	Comment:	query last used calibration temperature
	Instruction:	`RCAL?`
	Reply:	`RA 0000 RCAL 36.22`

Name

LCAL

Calibration temperature of the right heat exchanger assembly

Synopsis

Instruction:	`LCAL <calib temp>`
Reply:	`<rc> LCAL <calib temp>`
Instruction:	`LCAL?`
Reply:	`<rc> LCAL <calib temp>`

Description

The calibration temperature is used to calculate a permanent offset to the measured temperature. The delta between the <calib temp> and the actual verification temperature (see RVFY?) gives the offset. The calibration is assumed to be done at exactly 36'C. See also instructions CSTR and CSTP to start and stop calibration mode, and the apprpriate statechange events. The Info event EV 2840 indicates that the calibration setpoint (36'C) is reached closely enough to except the calibration temperature. See instruction CAL? for query of the active offsets.

Parameters

<calib temp> 34.00 .. 38.00 'C

Default Values

36

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	initiate calibration calculation to offset for 36.22 'C
	Instruction:	`LCAL 36.22`
	Reply:	`RA 0000 LCAL 36.22`
(b)	Comment:	query last used calibration temperature
	Instruction:	`LCAL?`
	Reply:	`RA 0000 LCAL 36.22`

Name

FCMP
Flow Compensation

Synopsis

Instruction:	`FCMP <value>`
Reply:	`<rc> FCMP <value>`
Instruction:	`FCMP?`
Reply:	`<rc> FCMP <value>`

Description

With the FCMP instruction the flow compensation can be turned off, set to 'auto' or set to a flow value. The default is off. If set to 'auto', there has to be clustered the pump with delivers the main-flow (see Configuration-parameters CFLW <main-pump>(page 417) and PCFL?(page 417)). This pump will then notify the column compartement about every flow change.
In the range of 0.05 ml/min to 0.8 ml/min, the temperature of the heat exchangers is calculated from the temperature delta of setpoint and ambient, and from the flow delta of 0.8 ml/min - flow home value. This flow compensation of the temperature is not seen at the temperature setpoint.

Parameters

<value>	-2 for 'auto'; -1 for off; 0.05 .. 0.8ml/min

Default Values

-1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set flow compensation to auto
	Instruction:	`FCMP -2`
	Reply:	`RA 0000 FCMP -2`
(b)	Comment:	query the flow compensation value
	Instruction:	`FCMP?`
	Reply:	`RA 0000 FCMP -2`

Name

CFLW

Cluster Main-pump for Flow Compensation

PCFL?

query possible pumps for clustering

Synopsis

Instruction:	`CFLW <main-pump>`
Reply:	`<rc> CFLW <main-pump>`
Instruction:	`CFLW?`
Reply:	`<rc> CFLW <main-pump>`
Instruction:	`PCFW?`
Reply:	`<rc> PCFL <list of pumps>`

Description

With the CFLW instruction the pump which delivers the main-flow is clusterd. After clustering the pump will notify the column compartment of every flow change.
PCFL? is used to query for possible pumps for clustering for Flow CCompensation.

Parameters

<main-pump>	pump which delivers the main-flow if empty, the cluster is released.
<list of pumps>	list of pumps (semi-colon separated) which can be clustered.

Default Values

na

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	cluster main-pump
	Instruction:	`CFLW "G1312A:DE13121312"`
	Reply:	`RA 0000 CFLW "G1312A:DE13121312"`
(b)	Comment:	query possible pumps
	Instruction:	`PCFL?`
	Reply:	`RA 0000 PCFL "G1310A:DE13101310;G1312A:DE13121312"`

20.3. Column Compartement Control Commands

Name

THRM
Turn on Temperature Control

Synopsis

Instruction: `THRM <on/off>`
Reply: `<rc> THRM <on/off>`
Instruction: `ACT:THRM?`
Reply: `<rc> ACT:THRM <on/off>`

Description

If the parameter is 1,the Column Compartement starts to control the temperature to the setpoint given by the instruction TEMP (and RTMP in seperation mode).
If the parameter is 0, temperature control is turn off.

Parameters

`<on/off>` 0 for 'turn off', 1 for 'turn on'

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: turn temperature control off
Instruction: `THRM 0`
Reply: `RA 0000 THRM 0`

Name

ATOG

Switches Valve if appropriate.

Synopsis

Instruction: ATOG

Reply: <rc> ATOG

Description

Switches valve if the method parameter VALV is set to "switch at prepare". This command is used within the prepare phase of the main cosylist -2.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:

Instruction:

Reply:

Name

PRTP?

Returns Possible Remote Temp Partners

Synopsis

Instruction: `PRTP?`

Reply: `<rc> PRTP <partners>`

Description

All actually CAN wired Modules with the Remote Source feature are list in 'partners'.

Parameters

`<partners>` empty if no partner found otherise, all the found partners seperated by colon, each partner identified by type:serialnumber

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query possible partners

Instruction: `PRTP?`

Reply: `RA 0000 PRTP "G1315C:DE12345678,G1315C:DE12345679"`

Name

LRTP
Set Left Remote Temp Partner

Synopsis

Instruction:	`LRTP <partner>`
Reply:	`<rc> LRTP <partner>`
Instruction:	`LRTP?`
Reply:	`<rc> LRTP <partner>`

Description

Defines the device identified by <partner> as the left heat exchanger's remote temperature source..

Parameters

`<partner>`	empty for no partner, otherise, a CAN device identified by type:serialnumber

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set left remote partner
	Instruction:	`LRTP "G1315C:DE12345678"`
	Reply:	`RA 0000 LRTP "G1315C:DE12345678"`
(b)	Comment:	query
	Instruction:	`LRTP?`
	Reply:	`RA 0000 LRTP "G1315C:DE12345678"`

Name

RRTP
Set Right Remote Temp Partner

Synopsis

Instruction:	`RRTP <partner>`
Reply:	`<rc> RRTP <partner>`
Instruction:	`RRTP?`
Reply:	`<rc> RRTP <partner>`

Description

Defines the device identified by <partner> as the ritht heat exchanger's remote temperature source.

Parameters

`<partner>`	empty for no partner, otherise, a CAN device identified by type:serialnumber

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set ritht remote partner
	Instruction:	`RRTP "G1315C:DE12345678"`
	Reply:	`RA 0000 RRTP "G1315C:DE12345678"`
(b)	Comment:	query
	Instruction:	`RRTP?`
	Reply:	`RA 0000 RRTP "G1315C:DE12345678"`

20.4. Column Compartement Actuals

Name

ACT:TEMP?
Actual temperature of left heat exchanger

Synopsis

Instruction: `ACT:TEMP?`
Reply: `<rc> ACT:TEMP? <temperature>`

Description

The ACT:TEMP? instruction queries for the actual temperatrue of the left heat exchanger. The sensor is located inside the exchanger block close to the capillary outlet. .

Parameters

<temperature>	-274, -50.00 .. 140.00 ’C -274 is for missing or undecipherable sensor values

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query left temperature
Instruction: `ACT:TEMP?`
Reply: `RA 0000 ACT:TEMP? 16.34`

Name

ACT:RTMP?
Actual temperature of right heat exchanger

Synopsis

Instruction: `ACT:RTMP?`
Reply: `<rc> ACT:RTMP <temperature>`

Description

The ACT:RTMP? instruction queries for the actual temperatrue of the right heat exchanger. The sensor is located inside the exchanger block close to the capillary outlet. .

Parameters

<temperature>	-274, -50.00 .. 140.00 ’C -274 is for missing or undecipherable sensor values

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query left temperature
Instruction: `ACT:RTMP?`
Reply: `RA 0000 ACT:RTMP 16.34`

Name

ACT:THRM?
query for on/off

Synopsis

Instruction: `ACT:THRM?`
Reply: `<rc> ACT:THRM <on/off>`

Description

The ACT:THRM? instruction queries for the actual state of the instrument. In case of on error a state change to off is done. In case of THON 1 a state change to on is done after power cycling.

Parameters

<on/off> 0 for off, 1 for on

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query instrument state
Instruction: `ACT:THRM?`
Reply: `RA 0000 ACT:THRM 1`

Name

ACT:VALV?

Actual position of column switching valve

Synopsis

Instruction: `ACT:VALV?`

Reply: `<rc> ACT:VALV <position>`

Description

The ACT:VALV? instruction queries for the actual position of the column switching valve.

Parameters

<position>	0,1,2 0 for position 1->6 1 for position 1->2 2 for option not installed

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query valve position

Instruction: `ACT:VALV?`

Reply: `RA 0000 ACT:VALV 1`

Name

ACT:LSCO?

Actual reading of left sensor at column side

Synopsis

Instruction:	`ACT:LSCO?`
Reply:	`<rc> ACT:LSCO <temperature>`

Description

The ACT:LSCO? instruction queries for the actual temperature of the left heat exchanger's column sensor. The reply containes the last reading of the sensor.

Parameters

<temperature>	-274, -50.00 .. 140.00 'C -274 for missing readings

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query temperature

Instruction: `ACT:LSCO?`

Reply: `RA 0000 ACT:LSCO 21.42`

Name

`ACT:RSCO?`

Actual reading of right sensor at column side

Synopsis

Instruction: `ACT:RSCO?`

Reply: `<rc> ACT:RSCO <temperature>`

Description

The ACT:RSCO? instruction queries for the actual temperature of the right heat exchanger's column sensor. The reply containes the last reading of the sensor.

Parameters

<temperature>	-274, -50.00 .. 140.00 'C -274 for missing readings

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query temperature
Instruction: `ACT:RSCO?`
Reply: `RA 0000 ACT:RSCO 20.48`

Name

ACT:LSHS?
Actual reading of left sensor at heat sink side

Synopsis

Instruction: ACT:LSHS?
Reply: <rc> ACT:LSHS <temperature>

Description

The ACT:LSHS? instruction queries for the actual temperature of the left heat exchanger's heat sink sensor. The reply containes the last reading of the sensor.

Parameters

<temperature> -274, -50.00 .. 140.00 'C
-274 for missing readings

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query temperature
Instruction: ACT:LSHS?
Reply: RA 0000 ACT:LSHS 21.42

Name

ACT:RSHS?
Actual reading of right sensor at heat sink side

Synopsis

Instruction: `ACT:RSHS?`
Reply: `<rc> ACT:RSHS <temperature>`

Description

The ACT:RSHS? instruction queries for the actual temperature of the right heat exchanger’s heat sink sensor. The reply containes the last reading of the sensor.

Parameters

<temperature>	-274, -50.00 .. 140.00 ’C -274 for missing readings

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query temperature
Instruction: `ACT:RSHS?`
Reply: `RA 0000 ACT:RSHS 21.42`

Name

ACT:SAMB?
Actual reading of left sensor at ambient correction position

Synopsis

Instruction: `ACT:SAMB?`
Reply: <rc> ACT:SAMB <temperature>

Description

The ACT:SAMB? instruction queries for the actual temperature of the left heat exchanger's ambient correction sensor. The reply containes the last reading of the sensor.

Parameters

<temperature> -274, -50.00 .. 140.00 'C
-274 for missing readings

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query temperature
Instruction: `ACT:SAMB?`
Reply: `RA 0000 ACT:SAMB 24.32`

Name

`ACT:SEXT?`

Actual reading of auxilliary external sensor

Synopsis

Instruction: `ACT:SEXT?`

Reply: `<rc> ACT:SEXT <temperature>`

Description

The ACT:SEXT? instruction queries for the actual temperature of the auxillary external sensor. The reply containes the last reading of the sensor.

Parameters

<temperature> -274, -50.00 .. 140.00 ’C
-274 for missing readings

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query temperature
Instruction: `ACT:SEXT?`
Reply: `RA 0000 ACT:SEXT 12.32`

Name

`ACT:WLTR?`

Actual timeout of wait function for left temperature ready

Synopsis

Instruction: `ACT:WLTR?`

Reply: `<rc> ACT:WLTR <time>`

Description

The ACT:WLTR? instruction queries for the actual remaining time till the wait function for the left heat exchanger's ready state will generate an error. The time is counted back to 0.

Parameters

<time> 0 .. 99999.00 minutes

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query time out value

Instruction: `ACT:WLTR?`

Reply: `RA 0000 ACT:WLTR 1.40`

Name

ACT:WRTR?

Actual timeout value of wait function for ready of right temperature

Synopsis

Instruction: `ACT:WRTR?`

Reply: `<rc> ACT:WRTR <time>`

Description

The ACT:WRTR? instruction queries for the actual remaining time till the wait function for the right heat exchanger's ready state will generate an error. The time is counted back to 0.

Parameters

<time> 0 .. 99999.00 minutes

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query time out value
Instruction: `ACT:WRTR?`
Reply: `RA 0000 ACT:WRTR 1.40`

Name

ACT:WLTP?

Actual timeout value of wait function for left temperature setpoint

Synopsis

Instruction: ACT:WLTP?

Reply: <rc> ACT:WLTP <time>

Description

The ACT:WLTP? instruction queries for the actual remaining time till the wait function for the left heat exchanger's temperature setpoint will generate an error. The time is counted back to 0.

Parameters

<time> 0 .. 99999.00 minutes

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query time out value

Instruction: ACT:WLTP?

Reply: RA 0000 ACT:WLTP 1.40

Name

ACT:WRTP?

Actual timeout value of wait function for right temperature setpoint

Synopsis

Instruction: ACT:WRTP?

Reply: <rc> ACT:WRTP <time>

Description

The ACT:WRTP? instruction queries for the actual remaining time till the wait function for the right heat exchanger's temperature setpoint will generate an error. The time is counted back to 0.

Parameters

<time> 0 .. 99999.00 minutes

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query time out value
Instruction: ACT:WRTP?
Reply: RA 0000 ACT:WRTP 1.40

Name

`ACT:LFAN?`

Actual rotations per minute of the left fan

Synopsis

Instruction: `ACT:LFAN?`

Reply: `<rc> ACT:LFAN <rot>`

Description

The ACT:LFAN? instruction queries for the actual rotations per minute of the left fan.

Parameters

<rot>	0 .. 200 rotations per minute

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query rotations per minute

Instruction: `ACT:LFAN?`

Reply: `RA 0000 ACT:LFAN 80`

Name

ACT:RFAN?

Actual rotations per minute of the right fan

Synopsis

Instruction: `ACT:RFAN?`

Reply: `<rc> ACT:RFAN <rot>`

Description

The ACT:RFAN? instruction queries for the actual rotations per minute of the right fan.

Parameters

<rot> 0 .. 200 rotations per minute

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query rotations per minute

Instruction: `ACT:RFAN?`

Reply: `RA 0000 ACT:RFAN 80`

Name

ACT:SETL?

Actual temperature setpoint of left heat exchanger assembly

Synopsis

Instruction: `ACT:SETL?`

Reply: `<rc> ACT:SETL <temperature>`

Description

The ACT:SETL? instruction queries for the actual setpoint of the left heat exchanger.

Parameters

<temperature> -274, -50 .. 140

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query temperature setpoint

Instruction: `ACT:SETL?`

Reply: `RA 0000 ACT:SETL 14.00`

Name

ACT:SETR?

Actual temperature setpoint of right heat exchanger assembly

Synopsis

Instruction: `ACT:SETR?`

Reply: <rc> ACT:SETR <temperature>

Description

The ACT:SETR? instruction queries for the actual setpoint of the right heat exchanger.

Parameters

<temperature> -274, -50 .. 140

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query temperature setpoint

Instruction: `ACT:SETR?`

Reply: `RA 0000 ACT:SETR 14.00`

Name

ACT:LVFY?

Actual verification temperature of left heat exchanger assembly

Synopsis

Instruction: `ACT:LVFY?`

Reply: `<rc> ACT:LVFY <temperature>`

Description

The ACT:LVFY? instruction queries for the actual verification temperature of the left heat exchanger. This temperature differs from the sensor reading by an offset, that allows to calculate the temperature seen on an external temperature messuring device.

Parameters

<temperature> -274, -50 .. 140

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query temperature setpoint

Instruction: `ACT:LVFY?`

Reply: `RA 0000 ACT:LVFY 28.25`

Name

ACT:RVFY?

Actual verification temperature of left heat exchanger assembly

Synopsis

Instruction: ACT:RVFY?

Reply: <rc> ACT:RVFY <temperature>

Description

The ACT:RVFY? instruction queries for the actual verification temperature of the right heat exchanger. This temperature differs from the sensor reading by an offset, that allows to calculate the temperature seen on an external temperature messuring device.

Parameters

<temperature> -274, -50 .. 140

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query temperature setpoint
Instruction: ACT:RVFY?
Reply: RA 0000 ACT:RVFY 37.11

Name

ACT:ADC?
Actual ADC reading

Synopsis

Instruction: `ACT:ADC?<channel>`
Reply: `<rc> ACT:ADC <channel>,<ADC-value>`

Description

The ACT:ADC? instruction queries for the last ADC reading of the given multiplexer channel.

Parameters

<channel>	1 .. 11, the number correspond to the pins of the ADC 1 for left cooling voltage 2 for left heating voltage 3 for right cooling voltage 4 for right heating voltage 5 for left current 6 for right current 7 for NTC (on board) 8 for PTC (leak sensor) 9 for voltage of left fan 10 for 36Volt supply 11 for 24Volt supply
<ADC-value>	0 .. 1023

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query temperature setpoint
Instruction: `ACT:ADC?`
Reply: `RA 0000 ACT:ADC 37.11`

20.5. Column Compartment Rawdata File Description

This description is a supplement to: Rawdata File on page 253.

The Agilent 1200 COLUMN COMPARTMENT generates up to 2 signals. The rawdata file can store these signals as records RA and RB. To select, which signal should be stored, see instruction RAWS on page 258. For selection of the data representation (decimal, hexadecimal or decimal) see instruction RAWF on page 259.

signal id	description	unit
RA	temperature of left heat exchanger	1/100 grdCelsius
RB	temperature of right heat exchanger	1/100 grdCelsius

20.6. Column Compartment Diagnose Buffers

Name

AD24 (diag)
24V history

Characteristics

max entries	10
max characters	20
storage	volatile
availability	alwaysalways

Contents

time, mean value

time,	seconds since 1970
mean values,	ADC measurements added up since the last entry, and devided by the number of readings since

Description

The multiplexer channel is connected the the 24V from the power supply.

Name

AD36 (diag)
36V history

Characteristics

max entries	10
max characters	20
storage	volatile
availability	always

Contents

time, mean value

time,	seconds since 1970
mean values,	ADC measurements added up since the last entry, and devided by the number of readings since

Description

The multiplexer channel is connected the the 36V from the power supply.

Name

ADFN (diag)
Fan Voltage

Characteristics

max entries	10
max characters	20
storage	volatile
availability	always

Contents

time, mean value

time,	seconds since 1970
mean values,	ADC measurements added up since the last entry, and devided by the number of readings since

Description

The multiplexer channel is connected the the fan voltage of the left column compartement side.

Name

ADRP (diag)
Peltier current (right side)

Characteristics

max entries	10
max characters	20
storage	volatile
availability	always

Contents

time, mean value

time,	seconds since 1970
mean values,	ADC measurements added up since the last entry, and devided by the number of readings since

Description

The multiplexer channel is connected the current measuremetn logic of the right column compartement side.

Name

ADLP (diag)
Peltier Current Left Side

Characteristics

max entries	10
max characters	20
storage	volatile
availability	always

Contents

time, mean value

time,	seconds since 1970
mean values,	ADC measurements added up since the last entry, and devided by the number of readings since

Description

The multiplexer channel is connected the current measurement logic of the left column compartement side.

Name

ARPV (diag)
Peltier Voltage (right side)

Characteristics

max entries	10
max characters	20
storage	volatile
availability	always

Contents

time, mean value

time,	seconds since 1970
mean values,	ADC measurements added up since the last entry, and devided by the number of readings since

Description

The multiplexer channel is connected the voltage circuit of the right column compartement side.

Name

ALPV (diag)

Peltier Voltage (left side)

Characteristics

max entries	10
max characters	20
storage	volatile
availability	always

Contents

time, mean value

time,	seconds since 1970
mean values,	ADC measurements added up since the last entry, and devided by the number of readings since

Description

The multiplexer channel is connected the voltage circuit of the left column compartement side.

Name

LRDY (diag)

Ready History (left side)

Characteristics

max entries	10
max characters	20
storage	volatile
availability	always

Contents

time, high/low, tempdiff

time,	seconds since 1970
high/low,	-1 for below setpoint , 1 for above setpoint
tempdiff	abs of (actual temp - setpoint)

Description

Readyness is checked once per second, a diag entry is done each time the actual temp differs from the setpoint for more than the 'ready delta setpoint' .

Name

RRDY (diag)
Ready History (rightt side)

Characteristics

max entries	10
max characters	20
storage	volatile
availability	always

Contents

time, high/low, tempdiff

time,	seconds since 1970
high/low,	-1 for below setpoint , 1 for above setpoint
tempdiff	abs of (actual temp - setpoint)

Description

Readyness is checked once per second, a diag entry is done each time the actual temp differs from the setpoint for more than the 'ready delta setpoint' .

Name

RPLH (diag)
Temp History Hours (right side)

Characteristics

max entries	20
max characters	20
storage	non volatile
availability	always

Contents

time, mean value

time,	seconds since 1970
mean values,	ADC measurements added up since the last entry, and devided by the number of readings since

Description

The multiplexer channel is connected the the fan voltage of the left column compartement side.

Name

RPLS (diag)

Fan Voltage

Characteristics

max entries	10
max characters	20
storage	volatile
availability	always

Contents

time, mean value

time,	seconds since 1970
mean values,	ADC measurements added up since the last entry, and devided by the number of readings since

Description

The multiplexer channel is connected the the fan voltage of the left column compartement side.

20.7. Column Compartement Diagnose Counters

Name

VLVR (diag)
CSV switching repetition
Increments by 1 with each additional retry to reach the valves end position.

storage	volatile
Event	none
History	none

Name

VLVS (diag)
CSV switches
Increments by 1 with each start of a switching

storage	none erasible
Event	EF 2834
History	10

Name

VLVF (diag)
CSV failures
Increments by 1 with each failed switching.

storage	battery buffered
Event	none
History	5

Name

LPOW (diag)
Increments by 1 every 10 Ws consumed by the left peltier.

storage	none erasible
Event	EF 2852
History	2

Name

RPOW (diag)
Increments by 1 every 10 Ws consumed by the right peltier

storage	none erasible
Event	EF 2853
History	2

Name

LCWR (diag)

Increments by one with each write to the left column tag.

storage	battery bufferd
Event	none
History	none

Name

LCRD (diag)

Increments by one with each read from the left column tag.

storage	none volatile
Event	EF 2078
History	5

Name

RCWR (diag)

Increments by one with each write to the right column tag.

storage	battery buffered
Event	none
History	none

Name

RCRD (diag)

Increments by one with each read from the right column tag.

storage	battery buffered
Event	none
History	none

Name

LTON (diag)

Increments by one each second the left peltier is controlled.

storage	none erasible
Event	2854
History	2

Name

RTON (diag)

Increments by one each second the right peltier is controlled.

storage	none erasible
Event	2855
History	2

20.8. Column Compartment Commands to build Applications

Name

WLTP
Wait till left temperature setpoint is reached than send COSY sync

Synopsis

Instruction: `WLTP <timeout>`
Reply: `<rc> WLTP <timeout>`
Instruction: `WLTP?`
Reply: `<rc> WLTP <timeout>`

Description

The WLTP instruction provides the possibility to synchronize any other function of a CAN wired module of the Agilent 1200 series to the reaching of the setpoint value. See also chapter COSY lists. If the timeout value is counted back before the setpoint is reached the error event EE 2811 is generated.

Parameters

<timeout> -1, 0.00 .. 99999 minutes
-1 for using the time defined by TOUT

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: add cosy function to wait for left temperature setpoint
Instruction: `COSY:ISEQ n, ’WLTP -1’, 567`
Reply: `RA 0000 COSY:ISEQ n, ’WLTP -1’, 567`

Name

WRTP

Wait till right temperature setpoint is reached than send COSY sync

Synopsis

Instruction:	`WRTP <timeout>`
Reply:	`<rc> WRTP <timeout>`
Instruction:	`WRTP?`
Reply:	`<rc> WRTP <timeout>`

Description

The WRTP instruction provides the possibility to synchronize any other function of a CAN wired module of the Agilent 1200 series to the reaching of the setpoint value. See also chapter COSY lists. If the timeout value is counted back before the setpoint is reached the error event EE 2812 is generated.

Parameters

<timeout>	-1, 0.00 .. 99999 minutes -1 for using the time defined by TOUT

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: add cosy function to wait for right temperature setpoint

Instruction: `COSY:ISEQ n, ’WRTP -1’, 568`

Reply: `RA 0000 COSY:ISEQ n, ’WRTP -1’, 568`

Name

WLTR

Wait till left temperature is within Ready Delta than send COSY sync

Synopsis

Instruction:	`WLTR <timeout>`
Reply:	`<rc> WLTR <timeout>`
Instruction:	`WLTR?`
Reply:	`<rc> WLTR <timeout>`

Description

The WLTR instruction provides the possibility to synchronize any other function of a CAN wired module of the Agilent 1200 series to the reaching of the Ready Delta (See Method Parameter LRDD). See also chapter COSY lists. If the timeout value is counted back before the setpoint is reached the error event EE 2811 is generated.

Parameters

<timeout>	-1, 0.00 .. 99999 minutes -1 for using the time defined by TOUT

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: add cosy function to wait for left temperature ready

Instruction: `COSY:ISEQ n, ’WLTR -1’, 568`

Reply: `RA 0000 COSY:ISEQ n, ’WLTR -1’, 568`

Name

WRTR

Wait till right temperature is within Ready Delta than send COSY sync

Synopsis

Instruction:	`WRTR <timeout>`
Reply:	`<rc> WRTR <timeout>`
Instruction:	`WRTR?`
Reply:	`<rc> WRTR <timeout>`

Description

The WRTR instruction provides the possibility to synchronize any other function of a CAN wired module of the Agilent 1200 series to the reaching of the Ready Delta (See Method Parameter RRDD). See also chapter COSY lists. If the timeout value is counted back before the setpoint is reached the error event EE 2812 is generated.

Parameters

<timeout>	-1, 0.00 .. 99999 minutes -1 for using the time defined by TOUT

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: add cosy function to wait for right temperature ready

Instruction: `COSY:ISEQ n, ’WRTR -1’, 568`

Reply: `RA 0000 COSY:ISEQ n, ’WRTR -1’, 568`

20.9. Instructions To Access The Column Identification Tag

Name

LCOL:/RCOL:
Column Identification

Synopsis

Instruction:	`LCOL:<keyword> <parameter>`
Reply:	`<rc> LCOL:<keyword> <parameter>`
Instruction:	`RCOL:<keyword> <parameter>`
Reply:	`<rc> RCOL:<keyword> <parameter>`

Description

The Column Compartment comprises two independent Column Identification Modules, one for the left heat exchanger assembly, and one for the right heat exchanger assembly.
The following table summarizes the instructions to set and list the information stored in the column identification tag. The instruction prefix <col> is either LCOL for the left , or RCOL for the right Identification module.
The instruction <col>:INJ leads to redefine (overwrite) the current number of injections. The next start of a run will increment this number by one. If no column switching valve is installed, both, the left AND the right number of injection will be incremented. If a column switching valve is present, only one number of injections will be incremented. If the left column is selected by the valve during the start of the run (see instruction VALVE 1), the left number of injection will be incremented. If the right column is selected, the number of injections of the right tag will be incremented.
If the tag is currently written to or is read or is not present, the replies to the instructions below are marked with with the message RA 2807.

The following Events indicate state transitions or information about the tags:

ES 2832	No left tag found or left tag lost
EV 2831	Left tag found, (started reading data)
ES 2844	Left tag data is valid
EV 2846	Verify of left tag failed (write to the tag has failed)
EV 2848	Number of injections incremented, left tag
EV 2838	Contents of left tag has been modified
ES 2843	No right tag
EV 2842	Right tag found, (started reading data)
ES 2845	Right tag data is valid
EV 2847	Verify of right tag failed(write to tag has failed)
EV 2849	Number of injections incremented, right tag
EV 2839	Contents of right tag has been modified

Table 2:

Instruction	Param/Range	Description	Example
<col>:PHAS <string>	16 characters	Stationary phase	ODS Hypersil
<col>:PTSZ <size> <col>:PTSZ?	0 .. 25.1	Particle size in micro meter	5.0
<col>:GEOM <length>, <diameter> <col>:GEOM?	length 0 .. 300 diameter 0 .. 9.999	Column Geometry in mm	100, 3.4
<col>:VVOL <volume> <col>:VVOL?	o .. 30.000	Column void volume in ml	10.2
<col>:PROD <product #> <col>:PROD?	12 characters	Product number	DE12345
<col>:SER <serial #> <col>:SER?	8 characters	serial number	SER123
<col>:BAT <batch #> <col>:BAT?	8 characters	batch number	abc123
<col>:MPRS <pressure> <col>:MPRS?	0 .. 400	max pressure in bar	380
<col>:MTMP <temp> <col>:MTMP?	0 .. 100	max temperature in ’C	64
<col>:MPH <ph value> <col>:MPH?	0 .. 14.0	max ph value	10.4
<col:INJ? <col>:INJ <#injections>	0 .. 64000	actual number of injections	722
<col>:CMNT <comment> <col>:CMNT?	16 characters	any comment	my private col

Name

COL:STAT?

Query the presence of a tag

Synopsis

Instruction: `COL:STAT?`

Reply: `<rc> COL:STAT <left status>, <right status>`

Description

Use ACT:RCOL? and ACT:LCOL? instead!

The hardware of the column compartment was changed, removing the bases for this instruction. It is now simulated for backward compatibility.

Parameters

<status byte> - present if '(status byte & 0x02) == 0'

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: identification tag is present
Instruction: `COL:STAT?`
Reply: `RA 0000 COL:STAT 0x10, 0x20`

(b) Comment: identification tag is not present
Instruction: `COL:STAT?`
Reply: `RA 0000 COL:STAT 0x12, 0x22`

Name

`ACT:LCOL?`

Query the presence of a tag (left side)

Synopsis

Instruction: `ACT:LCOL?`

Reply: `<rc> ACT:LCOL <found>`

Description

This query returns if a valid tag is installed at the left side heat exchanger assembly.

Parameters

<found> 0 for no tag, 1 for valid tag found

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: identification tag is present
Instruction: `ACT:LCOL?`
Reply: `RA 0000 ACT:LCOL 1`

(b) Comment: identification tag is not present
Instruction: `ACT:LCOL?`
Reply: `RA 0000 ACT:LCOL 0`

Name

ACT:RCOL?
Query the presence of a tag (right side)

Synopsis

Instruction: ACT:RCOL?
Reply: <rc> ACT:RCOL <found>

Description

This query returns if a valid tag is installed at the right side heat exchanger assembly.

Parameters

<found> 0 for no tag, 1 for valid tag found

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: identification tag is present
Instruction: ACT:RCOL?
Reply: RA 0000 ACT:RCOL 1

(b) Comment: identification tag is not present
Instruction: ACT:RCOL?
Reply: RA 0000 ACT:RCOL 0

Name

COL:RSET

Reset Tag Reading Hardware

Synopsis

Instruction: COL:RSET

Reply: <rc> COL:RSET

Description

Reset the microcontroller that reads the identification tag .

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:

Instruction: COL:RSET

Reply: RA 0000 COL:RSET

Name

COL:INIT
ReInit tag

Synopsis

Instruction: `COL:INIT <left/right>`
Reply: `<rc> COL:INIT <left/right>`

Description

Init a brand new tag. (Clearly: this is done automatically, but only once). This causes to write first and last acessible blocks to block 1.

Parameters

<left/right> - 0 for left tag, 1 for right tag

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: (re)init left tag
Instruction: `COL:INIT 0`
Reply: `RA 0000 COL:INIT 0`

21. Standard Autosampler/Injector (G1329A) High-Flow Autosampler (G2260A)

21.1. Autosampler Method Parameter

Name

DSPD
Set speed for drawing sample

Synopsis

Instruction:	DSPD <spdval>
Reply:	<rc> DSPD <spdval>
Instruction:	DSPD?
Reply:	<rc> DSPD <spdval>

Description

This parameter determines how fast the metering device moves to draw up sample.

Parameter

<spdval>	speed value ∈ [10, 1000] µl/min for 100 µl syringe speed value ∈ [90, 1000] µl/min for 900 µl syringe speed value ∈ [62, 10000] µl/min for G2260A

Default Value

200 µl/min
3000 µl/min for G2260A

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 3007 invalid draw/eject speed

Examples

(a)	Comment:	set draw speed to 150 µl/min
	Instruction:	DSPD 150
	Reply:	RA 0000 DSPD 150
(b)	Comment:	set draw speed to 1500 µl/min
	Instruction:	DSPD 1500
	Reply:	RE 3007 DSPD 1500

Name

ESPD

Set speed for ejecting sample

Synopsis

Instruction: `ESPD <spdval>`
Reply: `<rc> ESPD <spdval>`
Instruction: `ESPD?`
Reply: `<rc> ESPD <spdval>`

Description

This parameter determines how fast the metering device moves to eject sample.

Parameter

`<spdval>` speed value ∈ [10, 1000] µl/min for 100 µl syringe
speed value ∈ [90, 1000] µl/min for 900 µl syringe
speed value ∈ [62, 10000] µl/min for G2260A

Default Value

200 µl/min
3000 µl/min for G2260A

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 3007 invalid draw/eject speed

Examples

(a) Comment: set eject speed to 150 µl/min
Instruction: `ESPD 150`
Reply: `RA 0000 ESPD 150`

(b) Comment: set eject speed to 1500 µl/min
Instruction: `ESPD 1500`
Reply: `RE 3007 ESPD 1500`

Name

IPOS

Needle draw/eject position offset

Synopsis

Instruction:	`IPOS <counts>`
Reply:	`<rc> IPOS <counts>`
Instruction:	`IPOS?`
Reply:	`<rc> IPOS <counts>`

Description

Defines the offset of the needle from the default draw position in a vial.
0 means default draw position, negative values move needle closer to the bottom of the vial.

Parameter

`<counts>`	offset value $\in$ [-2.5, 35] mm

Default Value

0.0 mm

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0502` out of range

Examples

(a)	Comment:	set the needle offset position to -1.0 mm
	Instruction:	`IPOS -1`
	Reply:	`RA 0000 IPOS -1`
(b)	Comment:	request the currently set offset position
	Instruction:	`IPOS?`
	Reply:	`RA 0000 IPOS -1`

Name

IPRG

Name of the injector program

Synopsis

Instruction:	`IPRG <InjProg>`
Reply:	`<rc> IPRG <InjProg>`
Instruction:	`IPRG?`
Reply:	`<rc> IPRG <InjProg>`

Description

Sets the name of the injector program to execute at analysis start.
The name for the injector program is given as an integer.

Parameter

`<value>`	value ∈ [0, INT_MAX] user defined injector program -4 standard injection

Default Value

-4

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0901` no such COSY-List

Examples

(a)	Comment:	set user defined injector program
	Instruction:	`IPRG 33`
	Reply:	`RA 0000 IPRG 33`
(b)	Comment:	request the currently set injector program
	Instruction:	`IPRG?`
	Reply:	`RA 0000 IPRG 33`

Name

IVOL

Set value for injection volume

Synopsis

Instruction:	`IVOL <volume>`
Reply:	`<rc> IVOL <volume>`
Instruction:	`IVOL?`
Reply:	`<rc> IVOL <volume>`

Description

Set value for injection volume in units µl. The maximum injection volume depends on the size of syringe and seat capillary. See MPRM on page 478.

Parameters

`<volume>`	injection volume up to 8000 µl with 1 decimal point G2260A: up to 5000 µl

Default Value

5.0 µl

100 µl for G2260A

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 3008 invalid injection volume

Examples

(a)	Comment:	set injection volume to 5 µl
	Instruction:	`IVOL 5`
	Reply:	`RA 0000 IVOL 5`
(b)	Comment:	set injection volume to 2000 µl with standard seat capillary
	Instruction:	`IVOL 2000`
	Reply:	`RE 3008 IVOL 2000`

Name

`OVLP`

Set "Overlap Injection" mode

Synopsis

Instruction:	`OVLP <Mode>,<TimeOffset>`
Reply:	`<rc> OVLP <Mode>,<TimeOffset>`
Instruction:	`OVLP?`
Reply:	`<rc> OVLP <Mode>,<TimeOffset>`

Description

Set the overlap injection mode.

Parameters

`<Mode>`	0 Off 1 Prefetch next sample vial (for standard injections) 2 Overlap injection cycle
`<TimeOffset>`	In prefetch mode this is the time to wait after start pulse before prefetching next sample vial (time in minutes with 2 decimal points). The optimal value for this parameter is given by the formula: TimeOffset = Stoptime + Posttime - time to deliver sample In overlap mode **TimeOffset** can be set such that switching the valve to “By-pass” has a minimum influence to the analytical results of the current run (ideally between two peaks).

Default Values

Mode 0

TimeOffset 0.0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	turn overlap injection mode on
	Instruction:	`OVLP 2,0.2`
	Reply:	`RA 0000 OVLP 2,0.2`

Name

WSVI

Set wash vial number

Synopsis

Instruction:	`WSVI <vialnum>`
Reply:	`<rc> WSVI <vialnum>`
Instruction:	`WSVI?`
Reply:	`<rc> WSVI <vialnum>`

Description

Set the number of the vial to use for needle washing.

Parameter

`<vialnum>`	0	for no wash vial
	1 .. 100	for 100 vial tray
	1 .. 15	for the 15 vial halftray left
	101 .. 115	for the 15 vial halftray right
	1 .. 40	for the 40 vial halftray left
	101 .. 140	for the 40 vial halftray right
	201 .. 220	for external vial positions

Default Value

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

`RE 3002` invalid vial number

Examples

(a)	Comment:	set vial to use for needle washing
	Instruction:	`WSVI 5`
	Reply:	`RA 0000 WSVI 5`
(b)	Comment:	set invalid vial number
	Instruction:	`WSVI 150`
	Reply:	`RE 3002 WSVI 150`

21.2. Autosampler System Parameter

Name

MIVI
Set missing vial configuration

Synopsis

Instruction:	`MIVI <value>`
Reply:	`<rc> MIVI <value>`
Instruction:	`MIVI?`
Reply:	`<rc> MIVI <value>`

Description

Defines how the Autosampler behaves in a missing vial situation. Ignore missing vial is only possible with standard injections. Missing vial within an injector program generates an error every time.

Parameter

`<value>`	0 generates an error and stops analysis 1 ignores the missing vial and continues with a run of 6 seconds

Default Value

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment:
Instruction: `MIVI 0`
Reply: `RA 0000 MIVI 0`

(b) Comment: ask for current setting
Instruction: `MIVI?`
Reply: `RA 0000 MIVI 0`

Name

MPRM

Set parameter for metering device

Synopsis

Instruction:	`MPRM <Syringe>, <SeatCapillary>`
Reply:	`<rc> MPRM <Syringe>, <SeatCapillary>`
Instruction:	`MPRM?`
Reply:	`<rc> MPRM <Syringe>, <SeatCapillary>`

Description

Set size of syringe and seat capillary.

Parameter

`<Syringe>`	100 100 µl syringe with 200ul loop capillary
	900 900 µl syringe with 1800ul loop capillary
	900 900 µl syringe for G2260A
`<SeatCapillary>`	2.3 2.3 µl standard seat capillary
	1.2 1.2 µl low dispersion
	20 20 µl standard for G2260A
	-> multiple draw with 100 µl syringe:
	400 seat capillary of about 800 µl size
	max injection volume: 400 µl + 100 µl syringe = 500 µl
	1400 seat capillary of about 2800 µl size
	max injection volume: 1400 µl + 100 µl syringe = 1500 µl
	-> multiple draw with 900 µl syringe:
	1400 seat capillary of about 2800 µl size
	max injection volume: 900 µl + 900 µl syringe = 1800 µl
	8000 only with user defined seat capillary or pre-column

Default Values

`Syringe`	100
`SeatCapillary`	2.3
`Syringe`	900 for G2260A
`SeatCapillary`	20 for G2260A

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0502` out of range

Examples

(a)	Comment:	set standard configuration
	Instruction:	`MPRM 100,2.3`
	Reply:	`RA 0000 MPRM 100,2.3`
(b)	Comment:	set a configuration that allows multiple draw up to 500 µl
	Instruction:	`MPRM 100,400.0`
	Reply:	`RA 0000 MPRM 100,400.0`

Name

MDWT

Set wait time for multiple draw

Synopsis

Instruction:	`MDWT <Seconds>`
Reply:	`<rc> MDWT <Seconds>`
Instruction:	`MDWT?`
Reply:	`<rc> MDWT <Seconds>`

Description

Set wait time for pressure balance. After ejecting sample into the seat capillary the system waits the specified time before moving needle up for next sample drawing.

Parameter

`<Seconds>`	wait time $\in$ [0, 255] seconds

Default Values

Seconds: 1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

`RE 0502` out of range

Examples

(a)	Comment:	set wait time for multiple draw
	Instruction:	`MDWT 4`
	Reply:	`RA 0000 MDWT 4`
(b)	Comment:	get current setting of wait time for multiple draw
	Instruction:	`MDWT?`
	Reply:	`RA 0000 MDWT 4`

Name

GHOM
Gripper home after injection.

Synopsis

Instruction:	GHOM <Mode>
Reply:	<rc> GHOM <Mode>
Instruction:	GHOM?
Reply:	<rc> GHOM <Mode>

Description

Move gripper to home position after injection.
This feature has no effect if optimization is on (i.e. overlap injection or prefetch sample vial).

Parameter

<Mode>	0 leave gripper above last vial position (default) 1 move gripper to home position after injection

Default Values

Mode: 0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0502 out of range

Examples

(a)	Comment:	Gripper home after injection
	Instruction:	GHOM 1
	Reply:	RA 0000 GHOM 1

Name

TRAY:SET

Set position for a vial.

Synopsis

Instruction: `TRAY:SET <vial>,<x>,<y>,<z>,<d>`
Reply: `<rc> TRAY:SET <vial>,<x>,<y>,<z>,<d>`

Instruction: `TRAY:SET? <vial>`
Reply: `<rc> TRAY:SET? <vial>,<x>,<y>,<z>,<d>`

Description

Set a new vial position.

Parameter

<vial>	vial location ∈ [1, 220] valid vial numbers are: 1 .. 100 for 100 vial tray 1 .. 15 for 15 vial tray at left side 101 .. 115 for 15 vial tray at right side 1 .. 40 for 40 vial tray at left side 101 .. 140 for 40 vial tray at right side 201 .. 220 for external vial positions
<x>	absolute x-axis position of the vial location ∈ [-89.8, 180.0] mm
<y>	absolute y-axis position of the vial location ∈ [-89.8 , 89.8] mm
<z>	absolute z-axis position of the trays ∈ [0, 47.0] mm
<d>	vial diameter ∈ {12, 22}

Default Values

None

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 3005 given coordinates cannot be transformed to a vial position

Examples

(a) Comment: set a new vial position for vial 201
Instruction: `TRAY:SET 201,-80,-20,47,12`
Reply: `RA 0000 TRAY:SET 201,-80.000000,-20.000000,47.000000,12.000`

(b) Comment: get vial position for vial 201
Instruction: `TRAY:SET? 201`
Reply: `RA 0000 TRAY:SET 201,-80.000000,-20.000000,47.000000,12.000`

Name

`TRAY:TWST`

Reposition vial locations.

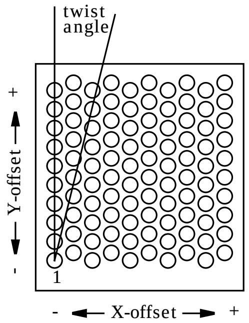
Synopsis

Instruction: `TRAY:TWST <xshift>,<yshift>,<theta>,<toffset>`
Reply: `<rc> TRAY:TWST <xshift>,<yshift>,<theta>,<toffset>`

Instruction: `TRAY:TWST?`
Reply: `<rc> TRAY:TWST <xshift>,<yshift>,<theta>,<toffset>`

Description

This command changes all vial positions of the sample rack. Calling the instruction with all parameter 0 resets the shifting and twisting to factory defaults.



Parameter

`<xshift>`	relative x-axis offset in millimeters ∈ [-0.6, 4.2] a positive number shifts the vial position to the right
`<yshift>`	relative y-axis offset in millimeters ∈ [-5.0, 1.6] a positive number shifts the vial position to the back.
`<theta>`	relative rotation of the vial positions in degrees ∈ [-0.4, 3.5]. The rotation is about the center position of vial number 1. Positive numbers rotate clockwise.
`<toffset>`	offset to theta initialization, a positive number is a clockwise offset (in degree)

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: twist all vial positions for 0.5 degrees
Instruction: `TRAY:TWST 0, 0, 0.5`
Reply: `RA 0000 TRAY:TWST 0, 0, 2.5`

(b) Comment: shift all vial positions for 1 mm in X-direction
Instruction: `TRAY:TWST 1,0,0`
Reply: `RA 0000 TRAY:TWST 1,0,0`

(c) Comment: get current shift values
Instruction: `TRAY:TWST?`
Reply: `RA 0000 TRAY:TWST 2.000000,0.000000,0.000000,0.000000`

Name

VIAL

Sample vial number.

Synopsis

Instruction:	VIAL <vnum>
Reply:	<rc> VIAL <vnum>
Instruction:	VIAL <vnum>,<NextVial>
Reply:	<rc> VIAL <vnum>,<NextVial>
Instruction:	VIAL?
Reply:	<rc> VIAL <vnum>
Instruction:	ACT:VIAL?
Reply:	<rc> ACT:VIAL <vnum>

Description

This command sets the actual sample vial number.

Parameter

<vnum> valid vial numbers are:

1 .. 100	for 100 vial tray
1 .. 15	for 15 vial tray at left side
101 .. 115	for 15 vial tray at right side
1 .. 40	for 40 vial tray at left side
101 .. 140	for 40 vial tray at right side
201 .. 220	for external vial positions
0	for blank run

<NextVial> lookahead for next sample vial, used to overlap the injection cycle (see OVLP on page 475)

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 3002 invalid vial number

Examples

(a) Comment: set sample vial #13
Instruction: `VIAL 13`
Reply: `RA 0000 VIAL 13`

(b) Comment: get current sample vial number
Instruction: `VIAL?`
Reply: `RA 0000 VIAL 13`

(c) Comment: get actual sample vial during analysis
Instruction: `ACT:VIAL?`
Reply: `ACT:VIAL 13`

Name

PINC

Change vial offset.

Synopsis

Instruction:	`PINC <delta>`
Reply:	`<rc> PINC <delta>`
Instruction:	`ACT:PINC?`
Reply:	`<rc> ACT:PINC <offset>`

Description

This command increment the vial offset by the value of delta. The used sample vial number is the sum of the actual set vial (see command VIAL) and the vial offset. If a new value is set with the command VIAL or the module is switched on the vial offset is set automatically to zero.

Parameter

<delta>	value, which is added to the vial offset
<offset>	the actual vial offset

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 3002 invalid vial number

Examples

Name

PRST

Reset vial offset.

Synopsis

Instruction: PRST

Reply: <rc> PRST

Description

This command reset the vial offset.

Parameter

<delta>

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

Name

LGHT

Tray illumination.

Synopsis

Instruction: LGHT <on/off>

Reply: <rc> LGHT <on/off>

Instruction: LGHT?

Reply: <rc> LGHT <on/off/uninit>

Description

Turns the sample illumination on or off. If the sampler returns 2 (uninitialized) on query the controller should set the value to 1 (enabled). This is to ensure that the illumination is disabled by default with controllers that do not support this command and enabled by default if the controller knows this command.

Parameter

<on/off> 0=off: switch off the tray illumination
1=on: switch on the tray illumination
2=uninitialized (default)

Default Values

Mode: 2

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0502 out of range

Examples

(a) Comment: switch on the tray illumination
Instruction: LGHT 1
Reply: RA 0000 LGHT 1

21.3. Thermostatted Sampler System Parameters

Name

`TEMP`

Set sample temperature.

Synopsis

Instruction:	`TEMP <deg>`
Reply:	`<rc> TEMP <deg>`
Instruction:	`TEMP?`
Reply:	`<rc> TEMP <deg>`

Description

This command gives a new temperature set-point to the thermostat module. The temperature range 4°C to 40°C is specified for ambient temperature up to 25°C.

Parameter

<deg>	temperature in °C with no decimal place adjustable range: 4°C - 40°C in 1° increments a value of -274 turns the thermostat module off

Default Value

20

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set sample temperature to 12°C
	Instruction:	`TEMP 12`
	Reply:	`RA 0000 TEMP 12`
(b)	Comment:	get actual temperature setting
	Instruction:	`TEMP?`
	Reply:	`RA 0000 TEMP 12`

Name

APON

Automatically power on.

Synopsis

Instruction:	`APON <OnOff>`
Reply:	`<rc> APON <OnOff>`
Instruction:	`APON?`
Reply:	`<rc> APON <OnOff>`

Description

This command is used to have the thermostat module automatically turned on at instrument (autosampler) power on.

For a time programmed power on for the thermostat module see ATON on page 195.

Parameter

OnOff	0	thermostat off at power on
	1	thermostat on at power on

Default Value

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	have the thermostat turned on at power on
	Instruction:	`APON 1`
	Reply:	`RA 0000 APON 1`

Name

NRMD
Set "Not Ready"-Mode.

Synopsis

Instruction:	`NRMD <param>`
Reply:	`<rc> NRMD <param>`
Instruction:	`NRMD?`
Reply:	`<rc> NRMD <param>`

Description

This command is used to set the "Not Ready"-Mode for the sample thermostat. The mode determines if the instrument shows "Not Ready" while cooling or heating. Normally the instrument ignores the states cooling or heating and allows the start of an analysis in any case. If temperature should not be ignored the instrument sets "Not Ready" and waits until temperature reaches the current set-point ± 1°. If the thermostat module is turned off (see command TEMP) the instrument does not set the "NotReady" states.

Parameter

param	0	ignore temperature
	1	set "Not Ready"-flag if temperature is not at the set-point

Default Value

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	turn on "Not Ready"-Mode
	Instruction:	`NRMD 1`
	Reply:	`RA 0000 NRMD 1`

21.4. ASutosampler Injector Commands

Name

CGOF

Terminates change gripper procedure.

Synopsis

Instruction: CGOF

Reply: <rc> CGOF

Description

This command moves the gripper arm from service position to home position. To initiate change gripper procedure see CGON on page 491.

Parameter

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: terminate change gripper procedure

Instruction: CGOF

Reply: RA 0000 CGOF

Name

CGON

Initiates change gripper procedure.

Synopsis

Instruction: CGON

Reply: <rc> CGON

Description

This command moves the gripper arm into service position. This is the position where the arm can be removed. See also CGOF on page 490.

Parameter

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: initiate change gripper procedure

Instruction: CGON

Reply: RA 0000 CGON

Name

CNOF

Terminates change needle procedure.

Synopsis

Instruction: CNOF

Reply: <rc> CNOF

Description

This command terminates the change needle procedure by moving the needle into seat. See also CNON on page 493.

Parameter

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: terminate change needle procedure

Instruction: CNOF

Reply: RA 0000 CNOF

Name

CNON

Initiates change needle procedure.

Synopsis

Instruction: CNON

Reply: <rc> CNON

Description

The change needle procedure first moves needle up and opens the safety flap. Then, with safety flap kept open the needle moves down and stops at about 15 mm above seat.

See also NDLE:DOWN on page 503, NDLE:UP on page 504, CNOF on page 492.

Parameter

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: initiate change needle procedure

Instruction: `CNON`

Reply: `RA 0000 CNON`

Name

CPOF

Terminates change piston procedure.

Synopsis

Instruction: CPOF

Reply: <rc> CPOF

Description

This command moves the piston to its home position. See also CPON on page 495

Parameter

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: terminate change piston procedure

Instruction: CPOF

Reply: RA 0000 CPOF

Name

CPON

Initiates change piston procedure.

Synopsis

Instruction: CPON

Reply: <rc> CPON

Description

The change piston procedure moves the plunger to the position where its spring has minimum load. See also CPOF on page 494.

Parameter

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: initiate change piston procedure
Instruction: CPON
Reply: RA 0000 CPON

Name

DRAW

Draw sample into syringe capillary.

Synopsis

Instruction:	`DRAW <vol>,<speed>,<source>`
Reply:	`<rc> DRAW <vol>,<speed>,<source>`
Instruction:	`DRAW <vol>,<speed>,<source>,<offset>`
Reply:	`<rc> DRAW <vol>,<speed>,<source>,<offset>`
Instruction:	`DRAW <vol>,<speed>,<source>,<vial>,<offset>`
Reply:	`<rc> DRAW <vol>,<speed>,<source>,<vial>,<offset>`

Description

Draws the given volume of sample from the specified source into syringe capillary.

Parameter

<vol>	-0.2,-0.1,0,0.1 ... 900 [µl] -0.2 means the maximum, e.g. remainder to 100 µl or 900 µl -0.1 means injection volume of the method (see IVOL on page 474)
<speed>	8 maximum speed (1000 [µl/min]) 9 speed defined in the method (see DSPD on page 470) 10 .. 1000 [µl/min] for 100 µl syringe 90 .. 1000 [µl/min] for 900 µl syringe
<source>	0 means from sample vial (see VIAL on page 483) 1 means from vial defined as sample vial plus value of <vial> 2 means from vial defined by parameter <vial> 3 means from seat (needle in seat while drawing) 4 means from air (needle is up while drawing)
<offset>	-2.6,-2.5 .. 35 [mm] draw position above default draw position -2.6 means the offset defined by the method (see IPOS on page 472) 0 is the default offset
<vial>	depending on the value of <source> parameter <vial> is an offset to sample vial (including negative numbers) or an absolute vial number valid vial numbers are: 1 .. 100 for 100 vial tray 1 .. 15 for 15 vial tray at left side 101 .. 115 for 15 vial tray at right side 1 .. 40 for 40 vial tray at left side 101 .. 140 for 40 vial tray at right side 201 .. 220 for external vial positions

Default Values

See parameter VIAL on page 483.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 3002` invalid vial number

Examples

(a) Comment: draw sample by using `vol`, `speed`, `source` and `offset` as set for the method (see IVOL on page 474, DSPD on page 470, VIAL on page 483 and IPOS on page 472)

Instruction: `DRAW -0.1,9,0,-2.6`

Reply: `RA 0000 DRAW -0.1,9,0,-2.6`

(b)	Comment:	draw 5 µl from sample vial with default draw speed
	Instruction:	`DRAW 5.0,9,0`
	Reply:	`RA 0000 DRAW 5.0,9,0`
(c)	Comment:	draw 5 µl from (sample vial + 10)
	Instruction:	`DRAW 5.0,9,1,10,0.0`
	Reply:	`RA 0000 DRAW 5.0,9,1,10,0.0`
(d)	Comment:	draw 5 µl from vial 12 with a draw speed of 250 µl/min
	Instruction:	`DRAW 5.0,250,2,12,0.0`
	Reply:	`RA 0000 DRAW 5.0,250,2,12,0.0`

Name

`EJCT`

Eject sample out of syringe capillary.

Synopsis

Instruction:	`EJCT <vol>,<speed>,<source>`
Reply:	`<rc> EJCT <vol>,<speed>,<source>`
Instruction:	`EJCT <vol>,<speed>,<source>,<offset>`
Reply:	`<rc> EJCT <vol>,<speed>,<source>,<offset>`
Instruction:	`EJCT <vol>,<speed>,<source>,<vial>,<offset>`
Reply:	`<rc> EJCT <vol>,<speed>,<source>,<vial>,<offset>`

Description

Ejects the given volume of sample out of the syringe capillary into the specified source.

Parameter

<vol>	-0.2,-0.1,0,0.1 ... 900 [µl] -0.2 means all that is in the syringe -0.1 means injection volume of the method (see IVOL on page 474)
<speed>	8 maximum speed (1000 [µl/min]) 9 speed defined in the method (see ESPD on page 471) 10 .. 1000 [µl/min] for 100 µl syringe 90 .. 1000 [µl/min] for 900 µl syringe
<source>	0 means into sample vial (see VIAL on page 483) 1 means into vial defined as sample vial plus value of <vial> 2 means into vial defined by parameter <vial> 3 means into seat (needle in seat while ejecting) 4 means into air (needle is up while ejecting)
<offset>	-2.6,-2.5 .. 35 [mm] eject position above default eject position -2.6 means the offset defined by the method (see IPOS on page 472) 0 is the default offset
<vial>	depending on the value of <source> parameter <vial> is an offset to sample vial (including negative numbers) or an absolute vial number valid vial numbers are: 1 .. 100 for 100 vial tray 1 .. 15 for 15 vial tray at left side 101 .. 115 for 15 vial tray at right side 1 .. 40 for 40 vial tray at left side 101 .. 140 for 40 vial tray at right side 201 .. 220 for external vial positions

Default Values

See parameter VIAL on page 483.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 3002` invalid vial number

Examples

(a) Comment: eject sample by using `vol`, `speed`, `source` and `offset` as set for the method (see IVOL on page 474, DSPD on page 470, VIAL on page 483 and IPOS on page 472)

Instruction: `EJCT -0.1,9,0,-2.6`
Reply: `RA 0000 EJCT -0.1,9,0,-2.6`

(b)	Comment:	eject 5 µl from sample vial with default eject speed
	Instruction:	`EJCT 5.0,9,0`
	Reply:	`RA 0000 EJCT 5.0,9,0`
(c)	Comment:	eject 5 µl into (sample vial + 10)
	Instruction:	`EJCT 5.0,9,1,10,0.0`
	Reply:	`RA 0000 EJCT 5.0,9,1,10,0.0`
(d)	Comment:	eject 5 µl into vial 12 with an eject speed of 250 µl/min
	Instruction:	`EJCT 5.0,250,2,12,0.0`
	Reply:	`RA 0000 EJCT 5.0,250,2,12,0.0`

Name

INJT

Complete an injection.

Synopsis

Instruction: `INJT <sync>`

Reply: `<rc> INJT <sync>`

Description

The instruction is used for injector programs to complete an injection. This includes removing sample vial from seat, moving needle into seat, switching valve to main pass and generating a start pulse for other modules.

Note: Within the same analysis the start pulse is send only one time by the command INJT, if a overlap injection mode (see command OVLP) is chosen.

Parameter

<sync> a number used to synchronize COSY-Lists

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: at end of injector program

Instruction: `INJT 302`

Reply: `RA 0000 INJT 302`

Name

MOVV

Move a vial to a new location

Synopsis

Instruction: `MOVV <from>,<to>`

Reply: `<rc> MOVV <from>,<to>`

Description

Moves a vial given by vial position <from> to the location of position <to>.

Parameter

<from>	source vial number	
<to>	destination vial number	
	valid source and destination vial numbers are:	
0	for the actual sample vial	
	1 .. 100	for 100 vial tray
	1 .. 15	for 15 vial tray at left side
	101 .. 115	for 15 vial tray at right side
	1 .. 40	for 40 vial tray at left side
	101 .. 140	for 40 vial tray at right side
	201 .. 220	for external vial positions

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

`RE 3002` invalid vial number

`RE 3003` undefined vial position

Examples

(a) Comment: move vial from location 4 to 5

Instruction: `MOVV 4,5`

Reply: `RA 0000 MOVV 4,5`

(b) Comment: move actual sample vial to 140

Instruction: `MOVV 0,140`

Reply: `RA 0000 MOVV 0,140`

Name

NDLE
Moving the needle.

Synopsis

Instruction:	`NDLE <func>,<offset>,<source>`
Reply:	`<rc> NDLE <func>,<offset>,<source>`
Instruction:	`NDLE <func>,<offset>,<source>,<vial>`
Reply:	`<rc> NDLE <func>,<offset>,<source>,<vial>`

Description

Instruction to move the needle.

Parameter

<func>	0 move to uppermost position 1 move to seat 2 position needle into a vial
<offset>	-2.6,-2.5 .. 35.0 [mm] position above default draw/eject position -2.6 means the offset defined by the method (see IPOS on page 472) 0 is the default offset
<source>	0 for sample vial (see VIAL on page 483) 1 for sample vial plus value of <vial> 2 for vial defined by parameter <vial>
<vial>	depending on the value of <source> parameter <vial> is an offset to sample vial (including negative numbers) or an absolute vial number valid vial numbers are: 1 .. 100 for 100 vial tray 1 .. 15 for 15 vial tray at left side 101 .. 115 for 15 vial tray at right side 1 .. 40 for 40 vial tray at left side 101 .. 140 for 40 vial tray at right side 201 .. 220 for external vial positions

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted]
`RE 3002` invalid vial number

Examples

(a) Comment: move needle up
Instruction: `NDLE 0,0,0`
Reply: `RA 0000 NDLE 0,0,0`

(b) Comment: move needle into seat
Instruction: `NDLE 1,0,0`
Reply: `RA 0000 NDLE 1,0,0`

(c) Comment: move needle into vial 102
Instruction: `NDLE 2,0,2,102`
Reply: `RA 0000 NDLE 2,0,2,102`

Name

NDLE:DOWN

Move needle down a given distance.

Synopsis

Instruction:	`NDLE:DOWN <mm>`
Reply:	`<rc> NDLE:DOWN <mm>`

Description

Moves needle down a given distance in mm. The needle cannot be moved beyond the lower needle sensor!

Parameter

<mm>	distance to move

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 3016 instruction NDLE:DOWN <mm> while needle is in lowest position

Examples

(a)	Comment:	move needle down 5 mm
	Instruction:	`NDLE:DOWN 5`
	Reply:	`RA 0000 NDLE:DOWN 5`
(b)	Comment:	move needle down 5 mm while needle is in seat
	Instruction:	`NDLE:DOWN 5`
	Reply:	`RE 3016 NDLE:DOWN 5`

Name

NDLE:UP

Move needle up a given distance.

Synopsis

Instruction:	`NDLE:UP <mm>`
Reply:	`<rc> NDLE:UP <mm>`

Description

Moves needle up a given distance in mm. The needle cannot be moved beyond the upper needle sensor!

Parameter

<mm>	distance to move

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 3015 instruction NDLE:UP <mm> while needle is in upper position

Examples

(a)	Comment:	move needle up 5 mm
	Instruction:	`NDLE:UP 5`
	Reply:	`RA 0000 NDLE:UP 5`
(b)	Comment:	move needle up 5 mm while needle is in upper most position
	Instruction:	`NDLE:UP 5`
	Reply:	`RE 3015 NDLE:UP 5`

Name

OINJ
Control "Overlap Injection"

Synopsis

Instruction: `OINJ <ControlSync>,<OnOffSync>`
Reply: `<rc> OINJ <ControlSync>,<OnOffSync>`

Description

Command to control overlap injection (see OVLP on page 475).

Parameters

ControlSync	synchronization event used to control overlap injection
OnOffSync	information if overlap mode is on or off

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get synchronization event 80005 if overlap injection is to do
Instruction: `OINJ 80005, 80004`
Reply: `RA 0000 OINJ 80005, 80004`

(b) Comment: get synchronization event 80004 if overlap injection is on
Instruction: `OINJ 0, 80004`
Reply: `RA 0000 OINJ 0, 80004`

Name

RESU

Resume injection.

Synopsis

Instruction: RESU

Reply: <rc> RESU

Description

The instruction continues a suspended injection or a suspended injector program. See also SUSP on page 509

Parameter

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RA 3014 instruction RESU or SUSP while no injection

Examples

(a)	Comment:	resume current injection
	Instruction:	RESU
	Reply:	RA 0000 RESU
(b)	Comment:	try to resume an injection when no injection was started
	Instruction:	RESU
	Reply:	RA 3014 RESU

Name

SMIX
Mix sample with syringe.

Synopsis

Instruction:	`SMIX <vol>,<speed>,<repeat>,<source>`
Reply:	`<rc> SMIX <vol>,<speed>,<repeat>,<source>`

Description

The instruction moves the syringe back and forth to mix drawn sample in the capillary.

Parameter

<vol>	-0.2,-0.1,0,0.1 ... 900 [µl] -0.2 means all that is in the syringe -0.1 means injection volume of the method (see IVOL on page 474)
<speed>	8 maximum speed (1000 [µl/min]) 9 speed defined in the method (see ESPD on page 471) 10 .. 1000 [µl/min] for 100 µl syringe 90 .. 1000 [µl/min] for 900 µl syringe
<repeat>	1 .. 99 repetition count
<source>	3 means in seat capillary (needle in seat while mixing) 4 means in air (needle is up while mixing)

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	mix 5 µl two times at default speed with needle in seat
	Instruction:	`SMIX 5,9,2,3`
	Reply:	`RA 0000 SMIX 5,9,2,3`

Name

SRNG

Move Syringe plunger.

Synopsis

Instruction:	`SRNG <func>,<vol>,<speed>`
Reply:	`<rc> SRNG <func>,<vol>,<speed>`
Instruction:	`SRNG <func>,<vol>,<speed>,<repetition>`
Reply:	`<rc> SRNG <func>,<vol>,<speed>,<repetition>`

Description

This instruction allows to directly move the syringe plunger.

Parameter

<func>	0 for draw (move back) 1 for eject (move forth) 2 for mix 3 for home position 4 for service position
<vol>	-0.2,-0.1,0,0.1 ... 900 [µl] -0.2 means the maximum, e.g. remainder to 100 µl or 900 µl -0.1 means injection volume of the method (see IVOL on page 474)
<speed>	8 maximum speed (1000 [µl/min]) 9 speed defined in the method (see DSPD on page 470) 10 .. 1000 [µl/min] for 100 µl syringe 90 .. 1000 [µl/min] for 900 µl syringe
<repetition>	1 .. 99

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: move syringe to its home position

Instruction: `SRNG 3,0,9`

Reply: `RA 0000 SRNG 3,0,9`

Name

SUSP

Suspend injection.

Synopsis

Instruction: `SUSP`

Reply: `<rc> SUSP`

Description

The instruction stops execution of the current injection or injector program. A suspended injection or injector program can be resumed (see RESU on page 506) or aborted (see ABRT on page 181).

Parameter

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

RA 3014 instruction RESU or SUSP while no injection

Examples

(a) Comment: suspend current injection

Instruction: `SUSP`

Reply: `RA 0000 SUSP`

(b) Comment: try to suspend an injection when no injection was started

Instruction: `SUSP`

Reply: `RA 3014 SUSP`

Name

VLVE

Switch solvent valve.

Synopsis

Instruction:	`VLVE <func>,<source>,<time>`
Reply:	`<rc> VLVE <func>,<source>,<time>`
Instruction:	`VLVE <func>,<source>,<vial>,<time>`
Reply:	`<rc> VLVE <func>,<source>,<vial>,<time>`

Description

Instruction to switch the solvent valve.

Parameter

<func>	0 switch valve to mainpass
	1 switch valve to mainpass with start pulse
	2 switch valve to bypass
	3 switch valve to mainpasss for a given time while needle is in position as defined by parameter <source>
<source>	0 sample vial
	1 for sample vial plus value of <vial>
	2 for vial defined by parameter <vial>
	3 needle in seat
	4 needle in air
<time>	1 .. 99 seconds
<vial>	depending on the value of <source> parameter <vial> is an offset to sample vial (including negative numbers) or an absolute vial number
	valid vial numbers are:
	1 .. 100 for 100 vial tray
	1 .. 15 for 15 vial tray at left side
	101 .. 115 for 15 vial tray at right side
	1 .. 40 for 40 vial tray at left side
	101 .. 140 for 40 vial tray at right side
	201 .. 220 for external vial positions

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: switch valve to mainpass for 12 seconds while needle in vial 5

Instruction: `VLVE 3,2,5,12`

Reply: `RA 0000 VLVE 3,2,5,12`

(b) Comment: switch valve to mainpass with start pulse

Instruction: `VLVE 1,0,1`

Reply: `RA 0000 VLVE 1,0,1`

(c) Comment: switch valve to bypass

Instruction: `VLVE 2,0,1`

Reply: `RA 0000 VLVE 2,0,1`

Name

WASH
Wash needle.

Synopsis

Instruction: `WASH <vial>,<cycle>,<stroke>`
Reply: `<rc> WASH <vial>,<cycle>,<stroke>`

Description

Instruction to wash the injection needle. Before the needle dips into the wash vial the syringe draws an air gap of 0.2 µl to protect the sample in the needle from the wash solvent.

Parameter

<vial>	0 use vial defined by the method (see WSVI on page 476) for vial numbers > 0: 1 .. 100 for 100 vial tray 1 .. 15 for 15 vial tray at left side 101 .. 115 for 15 vial tray at right side 1 .. 40 for 40 vial tray at left side 101 .. 140 for 40 vial tray at right side 201 .. 220 for external vial positions
<cycle>	number of cycles the needle dips into the vial. Range: 1 .. 5
<stroke>	distance to move the needle up and down. Range 1 .. 40 mm

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: wash needle in the vial defined by WSVI
Instruction: `WASH 0,1,25`
Reply: `RA 0000 WASH 0,1,25`

Name

LGHT

Switch on/off the sample chamber illumination.

Synopsis

Instruction: LGHT <off/on/uninit>

Reply: <rc> LGHT <off/on/uninit>

Description

Turns the sample illumination on or off. If the sampler returns 2 (uninitialized) on query the controller should set the value to 1 (enabled). This is to ensure that the illumination is disabled by default with controllers that do not support this command and enabled by default if the controller knows this command.

Parameter

<off/on/uninit>	0	off
	1	on
	2	uninitialized (off) - default

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: switch on the illumination of the sample chamber

Instruction: LGHT 1

Reply: RA 0000 LGHT 1

21.5. Autosampler Actuals

Name

`ACT:IVOL?`
Actual volume while drawing/ejecting.

Synopsis

Instruction: `ACT:IVOL?`
Reply: `<rc> ACT:IVOL <ivol>`

Description

Displays the actual volume in the syringe while drawing/ejecting in units µl.

Parameter

<ivol> actual volume in[µl]

Default Values

None

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get the actual draw volume
Instruction: `ACT:IVOL?`
Reply: `RA 0000 ACT:IVOL 2.5`

Name

`ACT:VIAL?`
Actual sample vial number.

Synopsis

Instruction: `ACT:VIAL?`
Reply: `<rc> ACT:VIAL <vnum>`

Description

Displays the actual sample vial number.

Parameter

<vnum>	actual vial number

Default Values

1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get the actual vial number
Instruction: `ACT:VIAL?`
Reply: `RA 0000 ACT:VIAL 1`

Name

ACT:DRPO?

Actual draw position.

Synopsis

Instruction: `ACT:DRPO?`

Reply: `<rc> ACT:DRPO <vnum>`

Description

Displays the actual draw position number. If the command DRAW is used and source is equal to the sample vial, <vnum> is set to sample vial (see VIAL) plus offset (see PINC).

The value is set to zero with switching valve to mainpass (see INJT, VLVE) and moving syringe to home position (see SRNG).

Parameter

<vnum> actual draw vial number

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: get the actual draw position (VIAL=2, PINC=3)

Instruction: `ACT:DRPO?`

Reply: `RA 0000 ACT:DRPO 5`

Name

ACT:INPO?

Actual injection position.

Synopsis

Instruction: `ACT:INPO?`

Reply: `<rc> ACT:INPO <vnum>`

Description

Displays the draw position of the sample, which has been injected to the system. With switching valve to mainpass (see INJT, VLVE) the value <vnum> is set to the value of ACT:DRPO?, if the draw position is valid.

INJT is executed

- at end of "standard injection"
- at end of "injection with needle wash"
- at end of "user-defined injector program" if there is no INJT-command in the program
- at INJT-command in a "user-defined injector program".

It's cleared if a blank-run is performed.

Parameter

<vnum> actual injection position

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

Name

ACT:DRVO?

Actual draw volume.

Synopsis

Instruction: ACT:DRVO?

Reply: <rc> ACT:DRVO <vol>

Description

Displays the volume, which has been drawn from the position ACT:DRPO. If the command DRAW is used and source is equal to the sample vial, <vnum> is set to sample vial (see VIAL) plus offset (see PINC).

The value is set to zero with switching valve to mainpass (see INJT, VLVE) and moving syringe to home position (see SRNG).

Parameter

<vol> actual draw volumen

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

Name

ACT:INVO?

Actual injection volumen.

Synopsis

Instruction: ACT:INVO?

Reply: <rc> ACT:INVO <vol>

Description

Displays the draw volume of the sample, which has been injected to the system. With switching valve to mainpass (see INJT, VLVE) the value <vnum> is set to the value of ACT:DRPO?, if the draw position is valid.

INJT is executed

- at end of "standard injection"
- at end of "injection with needle wash"
- at end of "user-defined injector program" if there is no INJT-command in the program
- at INJT-command in a "user-defined injector program".

It's cleared if a blank-run is performed.

Parameter

<vol> actual injection volume

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

21.6. Autosampler Sample Transport Commands

Name

`TRAY:CORR:RSET`
Reset gripper alignment.

Synopsis

Instruction: `TRAY:CORR:RSET`
Reply: `<rc> TRAY:CORR:RSET`

Description

Reset gripper alignment to initial values (factory set defaults). For gripper alignment see instructions TRAY:CORR:SET on page 520 and TRAY:CORR:USE on page 521.

Parameter

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: reset gripper alignment
Instruction: `TRAY:CORR:RSET`
Reply: `RA 0000 TRAY:CORR:RSET`

Name

TRAY:CORR:SET

Correct vial position.

Synopsis

Instruction: `TRAY:CORR:SET <vnum>`

Reply: `<rc> TRAY:CORR:SET <vnum>`

Description

Take current position of specified vial as one of the two teaching points for tray correction. To apply the teaching to all vials on the tray see TRAY:CORR:USE on page 521.

Parameter

<vnum> vial position to correct

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 3002 invalid vial number

Examples

(a) Comment: correct vial position

Instruction: `TRAY:CORR:SET <vnum>`

Reply: `RA 0000 TRAY:CORR:SET <vnum>`

Name

TRAY:CORR:USE

Correct all vial positions on the tray.

Synopsis

Instruction:	TRAY:CORR:USE <vnum>
Reply:	<rc> TRAY:CORR:USE <vnum>
Instruction:	TRAY:CORR:USE <vnum1>,<vnum2>
Reply:	<rc> TRAY:CORR:USE <vnum1>,<vnum2>

Description

Correct all vial positions on the tray according to the teached positions of <vnum1> and <vnum2>; both vials must belong to the same column or to the same row. How to teach positions see TRAY:CORR:SET on page 520. The correction includes twisting of all vial positions if the X-positions (Y-positions) of the two vials are different; all positions are twisted with vial number 1 as center (positive angle means clockwise).

If only one parameter is given, the instruction defines a new position for the specified vial according to the current gripper arm location.

Parameter

<vnum1>	first vial to use for the correction
<vnum2>	second vial to use for the correction

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 3002 invalid vial number

Examples

(a)	Comment:	correct all vial positions
	Instruction:	TRAY:CORR:USE <vnum1>,<vnum2>
	Reply:	RA 0000 TRAY:CORR:USE <vnum1>,<vnum2>
(b)	Comment:	correct a single vial position
	Instruction:	TRAY:CORR:USE <vnum>
	Reply:	RA 0000 TRAY:CORR:USE <vnum>

Name

TRAY:PARK

Move gripper arm to park position.

Synopsis

Instruction: `TRAY:PARK`

Reply: `<rc> TRAY:PARK`

Description

Move gripper arm to its park position.

Parameter

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: move gripper arm to its park position

Instruction: `TRAY:PARK`

Reply: `RA 0000 TRAY:PARK`

Name

TRAY:TYPE?

Get current tray type.

Synopsis

Instruction: `TRAY:TYPE?`

Reply: `<rc> TRAY:TYPE <left>,<right>`

Description

Get current tray type.

Tray types available: type 1: 100 x 2 ml
type 2: 40 x 2 ml
type 3: 15 x 8 ml
the following is the list of all possible tray combinations:

RA 0000 TRAY:TYPE 0 0	no tray installed
RA 0000 TRAY:TYPE 1 0	100 x 1.8 ml
RA 0000 TRAY:TYPE 2 2	40 x 1.8 ml left, 40 x 1.8 ml right
RA 0000 TRAY:TYPE 2 0	40 x 1.8 ml left
RA 0000 TRAY:TYPE 0 2	40 x 1.8 ml right
RA 0000 TRAY:TYPE 3 3	15 x 6 ml left, 15 x 6 ml right
RA 0000 TRAY:TYPE 3 2	15 x 6 ml left, 40 x 1.8 ml right
RA 0000 TRAY:TYPE 2 3	40 x 1.8 ml left, 15 x 6 ml right
RA 0000 TRAY:TYPE 0 3	15 x 6 ml right
RA 0000 TRAY:TYPE 3 0	15 x 6 ml left

Parameter

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get current tray type
Instruction: `TRAY:TYPE?`
Reply: `RA 0000 TRAY:TYPE 1,0`

21.7. Autosampler Service Instructions

Name

MONA:DOOR?
Get door sensor reading.

Synopsis

Instruction:	MONA:DOOR?
Reply:	<rc> MONA:DOOR <sensor>

Description

Read out the door sensor.

Parameter

<sensor>	0 if front door open 1 if front door closed

Default Values

None

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	read out the door sensor while door open
	Instruction:	MONA:DOOR?
	Reply:	RA 0000 MONA:DOOR 0

Name

`MONA:NMV?`

Get the status for needle, metering and valve.

Synopsis

Instruction: `MONA:NMV?`

Reply: `<rc> MONA:NMV <needle>,<metering>,<valve>`

Description

This command asks the instrument for the state of needle, metering and valve.

Parameter

<needle>	1 NDL_ON_TOP
	2 NDL_IN_SEAT
	3 NDL_IN_VIAL
	4 NDL_IN_MIXPOS
	5 NDL_ELSEWHERE
<metering>	plunger position in units mm
<valve>	0 MAINPASS
	1 MAINPASSPULSE
	2 BYPASS

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get the status for needle, metering and valve

Instruction: `MONA:NMV?`

Reply: `RA 0000 MONA:NMV 1,0,0.0`

Name

MONA:SENS?

Get the reading of a sensor.

Synopsis

Instruction:	`MONA:SENS? <sensor>`
Reply:	`<rc> MONA:SENS <sensor>,<value>`
Instruction:	`MONA:SENS? <sensor>`
Reply:	`<rc> MONA:SENS <L1>,<L2>,<L3>,<R1>,<R2>,<R3>`

Description

This command gets the reading of the specified sensor.

Parameter

<sensor>	0 X-slide
	1 Z-arm
	2 Theta
	3 Gripper
	4 upper needle sensor
	5 lower needle sensor
	6 metering (<value> = 0 means sensor sees LED signal)
	7 valve mainpass (<value> = 0: contact closed)
	8 valve bypass (<value> = 0: contact closed)
	9 tray encoding sensors (L1,L2,L3,R1,R2,R3)
	1, 1, 1, 0, 0, 0 -> 100 x 1.8 ml vial tray
	0, 1, 1, 1, 1, 1 -> 40 x 1.8 ml vial tray, left side
	1, 1, 1, 0, 1, 1 -> 40 x 1.8 ml vial tray, right side
	1, 0, 1, 1, 1, 1 -> 15 x 1.8 ml vial tray, left side
	1, 1, 1, 1, 0, 1 -> 15 x 1.8 ml vial tray, right side
	0, 0, 0, 1, 1, 1 -> 100 x 1.8 ml cooled vial tray
<value>	for reflective sensors: 0 = reflection, 1 = no reflection
<L1>,<L2>,<L3>	the left three LEDs
<R1>,<R2>,<R3>	the right three LEDs

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	read the upper needle sensor while needle is down
	Instruction:	`MONA:SENS? 4`
	Reply:	`RA 0000 MONA:SENS 4,1`
(b)	Comment:	read the tray encoding sensors with a 100 vial tray
	Instruction:	`MONA:SENS? 9`
	Reply:	`RA 0000 MONA:SENS 1,1,1,0,0,0`

Name

MONA:TRIG

Trigger to event controller.

Synopsis

Instruction: `MONA:TRIG <trigger>`

Reply: `<rc> MONA:TRIG <trigger>`

Description

This command sends a trigger to the event controller.

Parameter

`<trigger>` trigger to send

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: sends trigger to event controller
Instruction: `MONA:TRIG 5000`
Reply: `RA 0000 MONA:TRIG 5000`

Name

`MONA:TRQ`

Torque measurement.

Synopsis

Instruction:	`MONA:TRQ <motor>,<onoff>`
Reply:	`<rc> MONA:TRQ <motor>,<onoff>`
Instruction:	`MONA:TRQ?`
Reply:	`<rc> MONA:TRQ <Min>,<Max>,<Average>`

Description

This command is used to measure the torque for one of the four transport motors.

Parameter

`<motor>`	0 X-axis, 1 Z-axis, 2 Theta, 3 Gripper
`<onoff>`	0 torque measurement off
	1 torque measurement on (including reset for the average)
`<Min>`	the minimum value of the torque measurement
`<Max>`	the maximum value of the torque measurement
`<Average>`	the average value of the torque measurement

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	start torque measurement for Theta
	Instruction:	`MONA:TRQ 2,1`
	Reply:	`RA 0000 MONA:TRQ 2,1`
(b)	Comment:	get the torque measurement
	Instruction:	`MONA:TRQ?`
	Reply:	`RA 0000 MONA:TRQ -8,40,30`

Name

TRAY:ENC?

Get encoder counts.

Synopsis

Instruction:	`TRAY:ENC? <motor>`
Reply:	`<rc> TRAY:ENC <motor>,<CurEncdr>,<ActPos>`

Description

Read the encoder counts of the specified motor.

Parameter

`<motor>`	0 for X-axis 1 for Z-axis 2 for Theta 3 for Gripper
`<CurEncdr>`	the raw encoder reading
`<ActPos>`	the value used as actual position

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	get encoder counts of Theta motor
	Instruction:	`TRAY:ENC? 2`
	Reply:	`RA 0000 TRAY:ENC 2,123,303`

Name

TRAY:ERR?

Get axis error.

Synopsis

Instruction: `TRAY:ERR?`

Reply:
```
<rc> TRAY:ERR <xcnts>,<xcurr>, <zcnts>,<zcurr>,
                   <tcnts>,<tcurr>, <gcnts>,<gcurr>
```

Description

Get position error and current torque for all axis.

Parameter

<xcnts>	position error for X-axis
<xcurr>	current setting for X-axis [0 .. 255]
<zcnts>	position error for Z-axis
<zcurr>	current setting for Z-axis [0 .. 255]
<tcnts>	position error for Theta
<tcurr>	current setting for Theta [0 .. 255]
<gcnts>	position error for Gripper
<gcurr>	current setting for Gripper [0 .. 255]

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get position error and current setting for all axis

Instruction: `TRAY:ERR?`

Reply: `RA 0000 TRAY:ERR 0,0,0,0,0,0,0,0`

Name

TRAY:RSET

Reset sample trans.

Synopsis

Instruction: TRAY:RSET

Reply: <rc> TRAY:RSET

Description

Resets states and flags for the sample transport mechanism.

Parameter

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: reset sample trans

Instruction: TRAY:RSET

Reply: RA 0000 TRAY:RSET

21.8. Thermostatted Sampler Control Commands

Name

THRM:PWR

Turn on/off temperature control.

Synopsis

Instruction:	THRM:PWR <OnOff>
Reply:	<rc> THRM:PWR <OnOff>
Instruction:	THRM:PWR?
Reply:	<rc> THRM:PWR <OnOff>

Description

This command is used to turn on/off power for the sample thermostat module.

Parameter

OnOff	0	power off
	1	power on

Default Value

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	turn on power
	Instruction:	THRM:PWR 1
	Reply:	RA 0000 THRM:PWR 1
(b)	Comment:	get the current power setting (power is on)
	Instruction:	THRM:PWR?
	Reply:	RA 0000 THRM:PWR 1

Name

`THRM:SER`

Serial number for ALS thermostat.

Synopsis

Instruction:	`THRM:SER <SerialNumber>`
Reply:	`<rc> THRM:SER <SerialNumber>`
Instruction:	`THRM:SER?`
Reply:	`<rc> THRM:SER <SerialNumber>`

Description

This command is used to read or set the serial number for the ALS Thermostat.

Parameter

<SerialNumber> A string of 10 characters "DEywwxxxxx"

DE	produced in Germany (US for USA)
y	the year after 1990
ww	the week code
xxxxx	the running serial number

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: read the serial number

Instruction: `THRM:SER?`

Reply: `RA 0000 THRM:SER DE71400020`

Name

THRM:TYPE

Product number for ALS thermostat.

Synopsis

Instruction:	`THRM:TYPE <ProductNo>`
Reply:	`<rc> THRM:TYPE <ProductNo>`
Instruction:	`THRM:TYPE?`
Reply:	`<rc> THRM:TYPE <ProductNo>`

Description

This command is used to read or set the product number for the ALS Thermostat.

Parameter

<ProductNo> A string of 6 characters "G1330x"

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: read the product number
Instruction: `THRM:TYPE?`
Reply: `RA 0000 THRM:TYPE G1330A`

Name

THRM:REV

Hardware revision for ALS thermostat.

Synopsis

Instruction:	`THRM:REV <Revision>`
Reply:	`<rc> THRM:REV <Revision>`
Instruction:	`THRM:REV?`
Reply:	`<rc> THRM:REV <Revision>`

Description

This command is used to read or set the hardware revision for the ALS Thermostat.

Parameter

<Version> A string of 1 character A..Z

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: read the hardware version

Instruction: `THRM:REV?`

Reply: `RA 0000 THRM:REV A`

Name

THRM:SENS?

Get the reading of a sensor.

Synopsis

Instruction:	`THRM:SENS? <sensor>`
Reply:	`<rc> THRM:SENS <sensor>,<value>`

Description

This command gets the reading of either a temperature sensor or of a fan speed sensor. Depending on the sensor specified the command returns a temperature in °C or a speed in rpm.

Parameter

<sensor>	1 sensor for temperature controlled air blown to sample vials 2 peltier sensor heat exchange left 3 peltier sensor heat-sink left 4 peltier sensor heat exchange right 5 peltier sensor heat-sink right 6 auxiliary temperature sensor (connector J18 on main board) 7 heat-sink fan 1 left (1500 rpm .. 2640 rpm) 8 heat-sink fan 2 left (1500 rpm .. 2640 rpm) 9 heat-sink fan 3 right (1500 rpm .. 2640 rpm) 10 heat-sink fan 4 right (1500 rpm .. 2640 rpm) 11 heat exchange fan for temperature controlled air (2490 rpm .. 5190 rpm) 12 ambient temperature measured on main board 13 sensor for condensation (not available now, for future use)
<value>	temperature in °C or speed in rpm

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	read the temperature controlled air sensor
	Instruction:	`THRM:SENS? 1`
	Reply:	`RA 0000 THRM:SENS 1,12.0`
(b)	Comment:	get the speed of the heat exchange fan
	Instruction:	`THRM:SENS? 11`
	Reply:	`RA 0000 THRM:SENS 11,3960.00`

Name

`THRM:SIG1`

Map sensor reading to an actual signal.

Synopsis

Instruction:	`THRM:SIG1 <sensor>`
Reply:	`<rc> THRM:SIG1 <sensor>`
Instruction:	`THRM:SIG1?`
Reply:	`<rc> THRM:SIG1 <sensor>`

Description

This command maps the reading of a temperature sensor or of a fan speed sensor to the actual signal 1 (see ACT:SIG1? on page 542). For parameter sensor see THRM:SENS? on page 536.

Parameter

<sensor>	the sensor to set as actual for ACT:SIG1?

Default Value

2

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: map the temperature controlled air sensor to signal 1

Instruction: `THRM:SIG1 1`

Reply: `RA 0000 THRM:SENS 1`

Name

THRM:SIG2

Map sensor reading to an actual signal.

Synopsis

Instruction:	THRM:SIG2 <sensor>
Reply:	<rc> THRM:SIG2 <sensor>
Instruction:	THRM:SIG2?
Reply:	<rc> THRM:SIG2 <sensor>

Description

This command maps the reading of a temperature sensor or of a fan speed sensor to the actual signal 2 (see ACT:SIG2? on page 543). For parameter sensor see THRM:SENS? on page 536.

Parameter

<sensor>	the sensor to set as actual for ACT:SIG2?

Default Value

4

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: map the temperature controlled air sensor to signal 2

Instruction: THRM:SIG1 1

Reply: RA 0000 THRM:SENS 1

Name

THRM:STAT?

Read status register.

Synopsis

Instruction: `THRM:STAT? <Select>`

Reply: `<rc> THRM:STAT <Select>,<Value>`

Description

This command gets the reading of the status registers.

Parameter

<Select>	<Value>	
1	0	over current for peltier drive 1
	1	current ok
2	0	over current for peltier drive 2
	1	current ok
3	0	over current for peltier drive 3
	1	current ok
4	0	line voltage selector 230V
	1	line voltage selector 115V
5	0	fuse F1 for +36V raw defect or missing
	1	fuse F1 ok
6	0	fuse F2 for +36V raw defect or missing
	1	fuse F2 ok
7	0	TCA-board overtemp >80°
	1	TCA-board temperature ok
8		raw voltage measurement (36V -> 3.6V) the value displayed is the voltage multiplied by 10 (36 = 3.6V)
9		get the current power of the peltiers in watt RA 0000 THRM:STAT 9,<PowerAB>,<PowerC>

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: read the line voltage selection

Instruction: `THRM:STAT? 4`

Reply: `RA 0000 THRM:STAT 4,0`

21.9. Thermostatted Sampler Actuals

Name

ACT:TEMP?

Actual temperature.

Synopsis

Instruction: `ACT:TEMP?`

Reply: `<rc> ACT:TEMP <deg>`

Description

Displays the actual temperature of the thermostatted air blown to the sample tray.

Parameter

<deg> actual temperature in °C with 2 digital places

Default Values

None

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: get the actual temperature

Instruction: `ACT:TEMP?`

Reply: `RA 0000 ACT:TEMP 12.50`

Name

`ACT:THRM?`
Actual power setting.

Synopsis

Instruction: `ACT:THRM?`
Reply: `<rc> ACT:THRM <on/off>`

Description

Displays the actual power setting.

Parameter

<on/off>	1 for power on 0 for power off

Default Values

None

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get the actual power setting
Instruction: `ACT:THRM?`
Reply: `RA 0000 ACT:THRM 1`

Name

ACT:SIG1?
Actual sensor reading.

Synopsis

Instruction: ACT:SIG1?
Reply: <rc> ACT:SIG1 <value>

Description

Displays the actual value of the assigned sensor. See THRM:SIG1 on page 537.

Parameter

<value> actual temperature in °C for temperature sensors or speed in rpm for fan sensors

Default Value

Sensor 2

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: get the actual value of signal 1
Instruction: ACT:SIG1?
Reply: RA 0000 ACT:SIG1 12.50

Name

`ACT:SIG2?`

Actual sensor reading.

Synopsis

Instruction: `ACT:SIG2?`

Reply: `<rc> ACT:SIG2 <value>`

Description

Displays the actual value of the assigned sensor. See THRM:SIG2 on page 538.

Parameter

<value> actual temperature in °C for temperature sensors or speed in rpm for fan sensors

Default Values

Sensor 4

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get the actual value of signal 2

Instruction: `ACT:SIG1?`

Reply: `RA 0000 ACT:SIG1 12.50`

21.10. Thermostatted Sampler Rawdata File Description

This description is a supplement to: Rawdata File on page 253.
The Agilent 1200 ALS Thermostat generates up to 12 temperature signals which can be stored in a rawdata file. The rawdata file stores these signals as records RA to RL. Signal RA is the actual temperature of the thermostatted air blown to the sample vials. Signal RB is the temperature of the peltier sensor heat exchange left, and so forth. To select, which signal to store, see instruction RAWS on page 258.

For the instruction to start storing rawdata see RAWD:STRT on page 254. For selection of the data representation (binary, hexadecimal or decimal) see instruction RAWF on page 259.

RAWS	Description	Units
1	Temperature controlled air sensor.	1/100°C
2	Peltier sensor heat exchange left.	1/100°C
4	Peltier sensor heat-sink left	1/100°C
8	Peltier sensor heat exchange right	1/100°C
16	Peltier sensor heat-sink right	1/100°C
32	Auxiliary temperature sensor on main board	1/100°C
64	Heat-sink fan 1left	rpm
128	Heat-sink fan 2 left	rpm
256	Heat-sink fan 3 right	rpm
512	Heat-sink fan 4 right	rpm
1024	Heat exchange fan for temperature controlled air.	rpm
2048	Ambient temperature measured on main board.	1/100°C

21.11. Autosampler Diagnose Counters

Name

NCNT (diag)
Needle movement counter.

Characteristics

storage type: non erasable
history size : 5
limit event : EF 4038 Needle movement cycles exceeded.

Description

Increments by one with each succeessful "needle into seat" movement. The erasable part of the counter is set to 0 with each coldstart.

Name

VCNT (diag)
Valve switching counter.

Characteristics

storage type: non erasable
history size : 5
limit event : EF 4039 Valve switching cycles exceeded.

Description

Increments by one with each succeessful "valve to mainpass" switching. The erasable part of the counter is set to 0 with each coldstart.

21.12. Thermostatted Sampler Diagnostic

21.12.1. Counters

Name

`POOF (diag)`

Peltier On/Off counter.

storage type: non erasable
history size: 1
limit event: EF 4115 Peltier On/Off cycles exceeded.

Increments by one with each turning on of the peltier elements. The erasable part of the counter is set to 0 with each cold-start.

Name

`PONT (diag)`

Peltier on-time counter.

storage type: non erasable
history size: 1
limit event: EF 4116 Peltier on-time exceeded.

Counts the time in seconds during the peltier elements are on.

Name

`PLDF (diag)`

Peltier load factor.

storage type: non erasable
history size: 1
limit event: EF 4117 Peltier load factor.

Holds a load factor for the peltiers (set-point delta > 10°C).

Name

`PENY (diag)`

Total energy consumption.

storage type: non erasable
history size: 1
limit event: EF 4118 Peltier total energy consumption.

Stores the total energy consumption in Wh.

21.12.2. Buffers

Name

SNS1 (diag)

Diagnostic buffer for sensor 1.

Characteristics

max entries	60
max characters	6
storage	battery buffered
availability	always
entry	once per minute

Contents

Temperature

Temperature	the temperature reading of the sensor

Description

The buffer holds the temperature reading of sensor #1 at heat exchange for sample vials.

Name

SNS2 (diag)

Diagnostic buffer for sensor 2.

Characteristics

max entries	60
max characters	6
storage	battery buffered
availability	always
entry	once per minute

Contents

Temperature

Temperature	the temperature reading of the sensor

Description

The buffer holds the temperature reading of sensor #2 at heat-sink side.

Name

SNS3 (diag)

Diagnostic buffer for sensor 3.

Characteristics

max entries	60
max characters	6
storage	battery buffered
availability	always
entry	once per minute

Contents

Temperature

Temperature	the temperature reading of the sensor

Description

The buffer holds the temperature reading of sensor #3 at heat-sink side.

Name

SNS4 (diag)

Diagnostic buffer for sensor 4.

Characteristics

max entries	60
max characters	6
storage	battery buffered
availability	always
entry	once per minute

Contents

Temperature

Temperature	the temperature reading of the sensor

Description

The buffer holds the temperature reading of sensor #4 at heat exchange for sample vials.

Name

SNS5 (diag)

Diagnostic buffer for sensor 5.

Characteristics

max entries	60
max characters	6
storage	battery buffered
availability	always
entry	once per minute

Contents

Temperature

Temperature	the temperature reading of the sensor

Description

This sensor measures the actual temperature of the thermostatted air blown to the sample tray. The course of this temperature can also be stored in a rawdata file (see Thermostatted Sampler Rawdata File Description on page 544).

Name

COND (diag)

Diagnostic buffer for condensation sensor.

Characteristics

max entries	48
max characters	6
storage	battery buffered
availability	always
entry	once per 30 minutes

Contents

Temperature

Temperature	the temperature reading of the sensor

Description

This buffer stores the information if there was condensation during the last 24 hours.

22. Detector Instructions (Common)

22.1. Method Parameter

22.1.1. Analog Output Setting

Name

AZE1|AZE2
set zero offset of analog output

Synopsis

Instruction:	`AZE1 <offset%>`
Reply:	`<rc> AZE1 <offset%>`
Instruction:	`AZE2 <offset%>`
Reply:	`<rc> AZE2 <offset%>`
Instruction:	`AZE1?`
Reply:	`<rc> AZE1 <offset%>`
Instruction:	`AZE2?`
Reply:	`<rc> AZE2 <offset%>`

Description

Sets the zero offset voltage for analog output 1 (2) to a percentage of <offset%> of the full scale voltage. For information on the full scale voltage see ASC1 (ASC2) instruction.
The VWD only offers one analog output. Only AZE1 instruction is used.

Parameters

`<offset%>` 1 .. 99 [%]

Default Values

5

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0502 out of range

Examples

(a) Comment: set zero offset voltage for analog output 1to 50% of full scale
Instruction: `AZE1 50`
Reply: `RA 0000 AZE1 50`

Name

AAT1|AAT2
select analog output attenuation

Synopsis

Instruction:	`AAT1 <att_id>`
Reply:	`<rc> AAT1 <att_id>`
Instruction:	`AAT2 <att_id>`
Reply:	`<rc> AAT2 <att_id>`
Instruction:	`AAT1?`
Reply:	`<rc> AAT1 <att_id>`
Instruction:	`AAT2?`
Reply:	`<rc> AAT2 <att_id>`

Description

The DAD offer two analog outputs: signal A at analog output 1 and signal B at analog output 2.
The attenuation can be changed in power of 2 between 2AU and 2^{-10}AU.
The full range output is 2^{att_id-10}AU.
The VWD only offers one analog output. Only AAT1 instruction is used.

Parameters

`<att_id>`	0..11 full range output is 2^{att_id-10}AU. 11: 2AU output, 10: 1AU output, 9:0.5AU output .. 0: $\frac{1}{1024}$ AU output

Default Values

10 (1 AU full range output)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0502` out of range

Examples

(a)	Comment:	set 2 AU output at analog output 1:
	Instruction:	`AAT1 11`
	Reply:	`RA 0000 AAT1 11`
(b)	Comment:	set 2^{-10} AU (~1mAU) output at analog output 1:
	Instruction:	`AAT1 0`
	Reply:	`RA 0000 AAT1 0`
(c)	Comment:	query att selection for output 1
	Instruction:	`AAT1?`
	Reply:	`RA 0000 AAT1 0`

22.1.2. Prepare / Balance Behaviour (not for G1321A Fluorescence Detector)

Name

`PRMO`
prepare mode

Synopsis

Instruction:	`PRMO <prpmode>,<headroom>,<smpgain>,[<refgain>]`
Reply:	`<rc> PRMO <prpmode>,<headroom>,<smpgain>,[<refgain>]`
Instruction:	`PRMO?`
Reply:	`<rc> PRMO <prpmode>,<headroom>,<smpgain>,[<refgain>]`

Att: `<refgain>` is only used at VWD, `<prpmode>` and `<smpgain>` are ignored by DAD-C

Description

The setting of this instruction gives the possibility of different behaviour of the detector when a prepare (=balance) occured. A balance is started with the PREP command or automatic with the remote lines.

There are three different types of working: <prpmode> =

automatic this is the normal behaviour like before, but the <headroom> value is used to make the detector more flexible for negative absorbance or gradient application to get no underflow if more light is on the detector.

set_fix_gain if fix gain is used the <smpgain> value is set to the frontend

no_gain_change if this mode is used, the frontend don't change the gain if a PREP occured.

with <headroom> the minimum space between current light-level and a underflow value is given in mAU, this value is only used in *automatic* mode.

<headroom> can be given in a [AU] in factor f [1] or in percent of lamp intensity increase p [%]

there are following relationships: $\log(f) = a$ $\log(1+p/(100\%)) = a$ $f = 1+p/(100\%)$

examples: f=10 <-> p=900% <-> a=1AU f=2 <-> p=100% <-> a=301mAU

f=1 <-> p=0% <-> a=0AU f=1.1 <-> p=10% <-> a=41mAU

in this instruction <headroom> must be given in mAU <-> a

with <smpgain> the direct gain value is given for the nect PREP-cycle if the mode is *set_fix_gain*

Parameters

`<prpmode>`	0: *automatic* (use <headroom> value) 1: *set_fix_gain* (use <smpgain> value 2: *no_gain_change* (don't change current gain setting)
`<headroom>`	1 .. 4000 mAU (example: with 200mAU the detector may have a underflow below -200mAU.)
`<smp/refgain>`	0..5

Default Values

`<prpmode>`	0: *automatic*
`<headroom>`	100 mAU
`<smp/refgain>`	0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0502` out of range

Examples

(a) Comment: Set automatic balancing with more headroom than default
Instruction: `PRMO 0,500,1`
Reply: `RA 0000 PRMO 0,500,1`

Name

PRAU
automatic prepare

Synopsis

Instruction:	`PRAU <prep>, <strt>, <post>`
Reply:	`<rc> PRAU <prep>, <strt>, <post>`
Instruction:	`PRAU?`
Reply:	`<rc> PRAU <prep>, <strt>, <post>`

Description

This parameter allows automatic balancing of the detector in prerun/start or postrun phase either on remote activities or on conditional prepare commands (PRPC / PPSC) .

Parameters

<prep>	0 : no prepare/balance when prep remote line occured or PRPC command is sent
	1 : prepare/balance when prep remote line occured or PRPC command is sent
<strt>	0 : no prepare/balance when start occured
	1 : prepare/balance when start occured
<post>	0 : no prepare/balance after postrun finished or PPSC command sent
	1 : prepare/balance after postrun finished or PPSC command sent

Default Values

<prep>	1
<strt>	0
<post>	0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0502 out of range

Examples

(a) Comment: set automatic prepare/balance at end of postrun
Instruction: `PRAU 0,0,1`
Reply: `RA 0000 PRAU 0,0,1`

22.2. System Parameter

Name

ASC1|ASC2
select analog output voltage range

Synopsis

Instruction:	`ASC1 <range_id>`
Reply:	`<rc> ASC1 <range_id>`
Instruction:	`ASC2 <range_id>`
Reply:	`<rc> ASC2 <range_id>`
Instruction:	`ASC1?`
Reply:	`<rc> ASC1 <range_id>`
Instruction:	`ASC2?`
Reply:	`<rc> ASC2 <range_id>`

Description

The Agilent 1200 Series MWD (G1365A/B) and DAD (G1315A/B) offer two analog outputs: signal A at analog output 1 and signal B at analog output 2. The voltage range can be switched between 0.1V and 1V full scale. This setpoint is not affected when changing the detector method.
The Agilent 1200 Series VWD (G1314A/B/C) only offers one analog output. Only ASC1 instruction is used.

Parameters

`<range_id>`	0 or 1 0 means: 0.1 V full scale 1 means: 1 V full scale

Default Values

1 (1 V full scale)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0502` out of range

Examples

(a)	Comment:	select 0.1 V for output 1 (signals A) (or channel1 for Agilent 1200 Series VWD (G1314A/B/C))
	Instruction:	`ASC1 0`
	Reply:	`RA 0000 ASC1 0`
(b)	Comment:	query range selection for output 1
	Instruction:	`ASC1?`
	Reply:	`RA 0000 ASC1 1`

Name

D2ON

configure automatic turn on of deuterium lamp at startup

Synopsis

Instruction:	`D2ON <on/off>`
Reply:	`<rc> D2ON <on/off>`
Instruction:	`D2ON?`
Reply:	`<rc> D2ON <on/off>`

Description

This parameter determines whether the Deuterium lamp will be switched on during module start-up or not. The D2ON setting is not affected when changing the method. The D2ON? query returns the actual D2-lamp configuration.

Parameters

<on/off>	1:	D2-lamp is switched on at startup
	0:	D2-lamp is not switched on at startup

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

`RE 0502` out of range

Examples

(a)	Comment:	configure D2-lamp to be switched on at startup
	Instruction:	`D2ON 1`
	Reply:	`RA 0000 D2ON 1`
(b)	Comment:	query actual D2-lamp configuration
	Instruction:	`D2ON?`
	Reply:	`RA 0000 D2ON 1`

22.3. Control Commands

Name

LAMP
switch deuterium lamp on/off

Synopsis

Instruction:	`LAMP <on/off>`
Reply:	`<rc> LAMP <on/off>`
Instruction:	`ACT:LAMP?`
Reply:	`<rc> ACT:LAMP <lampstate>`

Description

The LAMP command switches the Deuterium lamp on/ off. The query ACT:LAMP? returns the actual lampstate. A successful igniton of the D2-lamp generates the following events:
ES 7400 UV lamp ignition (page 900)
ES 7404 UV lamp on (page 900)
ES 7406 UV lamp warm up (only VWD) (page 900)
Following events may occur during ignition phase in case of an error:
EE 7450 UV lamp: no current (page 900)
EE 7451 UV lamp: no voltage (page 900)
EE 7453 No heater current (page 900)
If ignitions fails following events are generated:
EE 7452 Ignition failed (page 900)
ES 7403 UV lamp off (page 900)
If the ignition of the D2-lamp failed (no lamp current or voltage sensed) the module automatically tries to ignit the lamp for a second time after a delay of 30 seconds and generates the event
EV 7401 UV lamp: retry ignition (page 900)

Parameters

`<on/off>`	0 means lamp off
	1 means lamp on
`<lampstate>`	0 means D2-lamp is off
	1 means D2-lamp is on
	2 means D2-lamp ignition
	3 means D2-lamp warmup (only VWD)

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0803 cover open violation
RE 1050 Lamp on not allowed while device in permanent error (DAD-C only)
RE 1050 Lamp on not allowed while device in permanent error (DAD-C only)

Examples

(a)	Comment:	switch D2-lamp on
	Instruction:	`LAMP 1`
	Reply:	`RA 0000 LAMP 1`
(b)	Comment:	query actual lamp status
	Instruction:	`ACT:LAMP?`
	Reply:	`ACT:LAMP 1`

Name

PREP

Balance the detector

Synopsis

Instruction:	`PREP`
Reply:	`<rc> PREP`
Instruction:	`AT:PREP <time>,0`
Reply:	`<rc> AT:PREP <time>,0`
Instruction:	`AT:PREP? <time>`
Reply:	`<rc> AT:PREP <time>,0`
Instruction:	`AT:PREP <time>   (DAD/MWD C only)`
Reply:	`<rc> AT:PREP <time>`
Instruction:	`AT:PREP? <time>  (DAD/MWD C only)`
Reply:	`<rc> AT:PREP <time>`

Description

PREP executes a balance at the detector. Gain is new calculated and set. Rawdata output is set to 0 AU and analog output is set to offset% -level defined with the ANA1 instruction.

The DAD/MWD-A/B performs the following:
- open slit (if not open) using current slitwidth
- set peakwidth (PKWD) to 0 (no filtering)
- calculate and set gain table (using one raw intensity scan: 50 ms)
- close slit
- measure dark scan (bunch of n scans * 50 ms ; n = 2^(OldPKWD+1) - 1)
- calculate and set darktable
- open slit using current slitwidth
- restore peakwidth
- measure scan
- calculate background scan and set rawdata output to 0 AU

The DAD/MWD-C performs the following:
- move slit to set position
- choose slowest possible readout program for desired PKWD
- while headroom is smaller than desired headroom: choose next faster readout program
- close slit
- measure dark spectrum
- open slit to set position
- measure background spectrum

A successful balance will generate the events listed below:

ES 7200 Detector: PREPARE (page 903)

ES 7090 Detector : IDLE (page 903)

see also PRMO and PRAU for working mode and automation of PREP command

Parameters

`<time>`	0 .. 99999.00 runtime in minutes, counted upward from 0 at start of run.

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6001 prep while startup
RE 6002 prep while reset
RE 6003 Prepare not allowed during test (only DAD-SL (G1315C))
RE 6004 prep while prep
RE 6005 prep while scan
RE 6006 prep while calibration test
RE 6007 prep while holmium check
RE 6011 prep while leak
RE 6012 prep while shutdown
RE 6013 Prepare not allowed while abort
RE 6021 prep while ignition
RE 6022 prep while lampoff
RE 6023 prep while lamp warmup
RE 6030 prep not allowed
RE 1021 Prepare while DSP not ok (only DAD/MWD (G1315A; G1315B; G1365A; G1365B))

Examples

(a) Comment: execute balance
Instruction: `PREP`
Reply: `RA 0000 PREP`

Name

PRPC

Conditional balance at prerun

PPSC

Conditional balance after postrun

Synopsis

Instruction:	PRPC
Reply:	<rc> PRPC
Instruction:	PPSC
Reply:	<rc> PPSC

Description

Executes a balance due to the setting of the automatic prepare mode parameter (PRAU).
PRPC balances the detector if "automatic balance at prerun" is set (PRAU 1,x,x)
PPSC balances the detector if "automatic balance after postrun" is set (PRAU x,x,1).
Both instructions are only valid as part of a COSY-list

Parameters

None

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:

Instruction:	PRPC
Reply:	RA 0000 PRPC

Name

SIMU:DFLT
loads default test chromatogram
SIMU:DNLC "<data> ..."
adds data to test chromatogram
SIMU:RSET
reset test chromatogram to OFF
ACT:SIMU?
query test chromatogram status

Synopsis

Instruction:	SIMU:DFLT
Reply:	<rc> SIMU:DFLT
Instruction:	SIMU:DNLC "<datapoint><datapoint>..."
Reply:	<rc> SIMU:DNLC "<datapoint>..."
Instruction:	SIMU:RSET
Reply:	<rc> SIMU:RSET
Instruction:	ACT:SIMU?
Reply:	<rc> ACT:SIMU <status>

Description

For test purposes the Agilent 1200 Series MWD (G1365A/B) and DAD (G1315A/B) has the capability to send a test chromatogram (both, digital data and analog output data); it can be eighter a built-in set of data or a data set previously downloaded into the module.
The module needs to be prepared by loading the built-in test data (SIMU:DFLT) or by downloading a test chromatogram (SIMU:DNLC "..."). Whenever the module is prepared for a test run, the event
EV 7210 Prepared for send simulated Chromatogramm (page 900)
is sent.
Only the one run following this preparation will sent the requested data. The data header of the rawdata file is "RD SIM ..." instead of "RD RUN ...", and the event
EV 7211 Start sending a simulated Chromatogramm (page 900)
is sent (additionally) at start of this test run. During a test run, no spectra are delivered.
Whenever a test run ends or the test is reset (SIMU:RSET), the event
EV 7212 End sending a simulated Chromatogramm (page 900)
is sent.
If a run lasts longer than the stored test data points can provide for, the test data are read once more from the very beginning.
The test data are sent at the data rate of the actual detector method. The test data are not filtered and therefore are not affected by changing the peakwidth setting (PKWD). Changing the peakwidth setting will only influence the time axis of the chromatogram data.

Parameters

<datatpoint>	8 character HEX-ASCII, representing a 4 byte signed integer in MOTOROLA FORMAT (1 count is 2^-21 AU).
<status>	0 means: test chromatogram switched OFF >0 means: number of data points of the actual test chromatogram.

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0772 not enough memory (page 918)

Examples

(a)	Comment:	reset test chromatogram to OFF
	Instruction:	`SIMU:RSET`
	Reply:	`RA 0000 SIMU:RSET`
(b)	Comment:	load default test chromatogram
	Instruction:	`SIMU:DFLT`
	Reply:	`RA 0000 SIMU:DFLT`
(c)	Comment:	query actual test chromatogram status
	Instruction:	`ACT:SIMU?`
	Reply:	`RA 0000 ACT:SIMU 675`
(d)	Comment:	add 2 data points to actual test chromatogram: 1) 0000A0000H = 655360 * 2^-21AU = 312.5 mAU and 2) FFFFFF00H = -256 * 2^-21AU = 0.1216 mAU
	Instruction:	`SIMU:DNLC "000a0000ffffff00"`
	Reply:	`RA 0000 ACT:SIMU 677`

22.4. Detector Common Spectra Monitor Commands (DAD, FLD, SPM)

Following monitor instructions only apply to Agilent 1200 Series DAD (G1315A/B), FLD (G1321A) and SPM (G8453A).

Name

`SPMO:STRT`
start spectra monitor
`SPMO:STOP`
stop spectra monitor
`SPMO:PERI <period>`
set spectra monitor period
`SPMO:PERI?`
query spectra monitor period

Synopsis

Instruction:	`SPMO:STRT`
Reply:	`<rc> SPMO:STRT`
Instruction:	`SPMO:STOP`
Reply:	`<rc> SPMO:STOP`
Instruction:	`SPMO:PERI <period>`
Reply:	`<rc> SPMO:PERI <period>`
Instruction:	`SPMO:PERI?`
Reply:	`<rc> SPMO:PERI <period>`

Description

The spectra monitor periodically offers spectra at the spectra monitor CU (CU#6). The data format of the spectra is the same as for the rawdata file spectra, except for the record header:

SM BIN, 217;<int1><int2>...

where '217' is the number of integers following.

The spectra monitor is controlled by the start and stop instructions SPMO:STRT and SPMO:STOP. The period instruction SPMO:PERI controls, how frequently spectra are offered at the monitor CU. If the last spectrum has not been read out of the spectra CU when the next spectrum is ready, this new spectrum is discarded.

Parameters

<period>	1 ... 999 (seconds) 0 means: as fast as possible

Default Values

default value for period is 2 (seconds)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set spectra monitor period to 1 second
	Instruction:	`SPMO:PERI 1`
	Reply:	`RA 0000 SPMO:PERI 1`
(b)	Comment:	start the spectra monitor
	Instruction:	`SPMO:STRT`
	Reply:	`RA 0000 SPMO:STRT`

23. Diode-Array Detector (G1315B) MultiWavelength Detector (G1365B)

23.1. Agilent 1200 Series DAD / MWD Method Parameter

23.1.1. Main Method Parameter

Name

PKWD

Sets peakwidth of smallest peak expected.

Synopsis

Instruction: PKWD <peakwidthnum>
Reply: <rc> PKWD <peakwidthnum>
Instruction: PKWD?
Reply: <rc> PKWD <peakwidthnum>

Description

The PKWD setpoint is the peakwidth at half height of the smallest peak expected. The setpoint controls the signal and spectra filtering and the digital datarate.It is also used to preset the peakdetector at start of run. After changing the PKWD parameter, the detector balances automatically.

Parameters

<peakwidthnum> 0 .. 7

PKWD	peakwidth at half height	response time [10 .. 90 %]	signal data rate	signal data rate	spectral data rate	spectral data rate every 2nd
0	< 0.01min	< 0.1 sec	20 Hz	50 ms	20 Hz	10 Hz
1	> 0.01 min	0.2 sec	20 Hz	50 ms	20 Hz	10 Hz
2	> 0.02 min	0.5 sec	10 Hz	100 ms	10 Hz	5 Hz
3	> 0.05 min	1sec	5 Hz	200 ms	5 Hz	2.5 Hz
4	> 0.1 min	2 sec	2.5 Hz	400 ms	2.5 Hz	1.25 Hz
5	> 0.2 min	4 sec	1.25 Hz	800 ms	1.25 Hz	0.62 Hz
6	> 0.4 min	8 sec	0.62 Hz	1.6 sec	0.62 Hz	0.32 Hz
7	> 0.85 min	16 sec	0.31 Hz	3.2 sec	0.31 Hz	0.16 Hz

Default Values

4

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0502 out of range

Examples

(a) Comment: set peakwidth parameter
Instruction: PKWD 5
Reply: RA 0000 PKWD 5

Name

MEMO
spectra acquisition mode

Synopsis

Instruction:	MEMO <mode>
Reply:	<rc> MEMO <mode>
Instruction:	MEMO?
Reply:	<rc> MEMO <mode>
Instruction:	AT:MEMO <time>,<mode>
Reply:	<rc> AT:MEMO <time>,<mode>
Instruction:	AT:MEMO? <time>
Reply:	<rc> AT:MEMO <time>,<mode>
Instruction:	ACT:MEMO?
Reply:	ACT:MEMO <mode>

Description

Controls the spectra acquisition during the analytical run. If set to 'all' and 'all in peak', the number of spectra stored per time depends on the peakwidth setting (PKWD). If set to a value other than 'none' or 'all', the spectra acquisition also depends on the peakdetector settings (THRS and AT:PD-WD).

MEMO is a method parameter; changing its value affects both, the method parameter setpoint and the actual value. During a run, the spectra acquisition mode may change due to time-programmed events. The query ACT:MEMO? always returns the effective acquisition mode whereas the MEMO? returns the method setpoint. At the end of the run, the method parameter again becomes active, independent of the latest time-programmed settings.

Parameters

<time>	0 .. 99999.00	
	runtime in minutes, counted upward from 0 at start of run.	
<mode>	0 means	none (no spectra are stored)
	1 mens	apex (spectra nearest to top of peak is stored)
	2 means	apex and baseline (begin and end of peak)
	3 means	apex and slopes
	4 means	apex, slopes and baseline (begin and end of peak)
	5 means	all in peak (half data rate than all)
	6 means	every 2nd spectrum
	7 means	all

Default Values

0 (no spectra are stored)

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0502 out of range
RE 0302 timetable full
RE 0304 no timetable

Examples

(a)	Comment:	set spectra acquistion mode to 7 (all spectra are stored)
	Instruction:	MEMO 7
	Reply:	RA 0000 MEMO 7
	Reply:	

Name

SLIT
set slitwidth

Synopsis

Instruction:	`SLIT <slitwidth`
Reply:	`<rc> SLIT <slitwidth>`
Instruction:	`SLIT?`
Reply:	`<rc> SLIT <slitwidth>`

Description

Defines the width of the built-in microslit in its open-position.The SLIT method parameter is not time-programmable, the SLIT? query returns the method parameter setpoint.
The instrument performs a balance automatically, whenever the SLIT parameter has been changed.

Parameters

`<slitwidth>`	0 means	1 nm slit
	1 means	2 nm slit
	2 means	4 nm slit
	3 means	8 nm slit
	4 means	16 nm slit

Default Values

2 (4 nm slit)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0502` out of range

Examples

(a)	Comment:	define 16 nm slit
	Instruction:	`SLIT 4`
	Reply:	`RA 0000 SLIT`
(b)	Comment:	query parameter setpoint
	Instruction:	`SLIT?`
	Reply:	`RA 0000 SLIT 4`

23.1.2. Peak Detector

Name

AT:PDPW
peakdetector peakwidth

Synopsis

Instruction:	AT:PDPW <time>,<peakwidthnum>
Reply:	<rc> AT:PDPW <time>,<peakwidthnum>
Instruction:	AT:PDPW? <time>
Reply:	<rc> AT:PDPW <time>,<peakwidthnum>
Instruction:	ACT:PDPW?
Reply:	<rc> ACT:PDPW <peakwidthnum>

Description

The peakdetector controls the automatic acquisition of spectra; it needs to be adjusted to the (expected) peakwidth and to an absorbance threshold .
To allow adjustment of the peakdetector settings during a separation, both, the peakdetector peakwidth and the threshold are timeprogrammable.
At start of an analytical run, the peakdetector peakwidth is preset to the value of PKWD , during a run it can be enlarged by a factor of 2,4 or 8.
The ACT:PDPW? query always returns the effective peakdetector peakwidth during a run.

Parameters

<time> 0 .. 99999.00
runtime in minutes, counted upward from 0 at start of run.

<peakwidthnum> 0 .. 3
0 : same value as PKWD
1 : peakdetector peakwidth is 2-times greater than PKWD
2 : peakdetector peakwidth is 4-times greater than PKWD
3 : peakdetector peakwidth is 8-times greater then PKWD
Note: Peakdetector peakwidth only can be enlarged up to the maximum peakwidth which can be defined by PKWD. If e.g. the value of PKWD is set to 6 any value of the peakdetector peakwidth greater than 1 will have no affect on the peakdetector.

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0302 timetable full
RE 0304 no timetable
RE 0502 out of range

Examples

(a) Comment: Increase (expected) peakwidth for peakdetector by factor 2 at 5.5 minutes.
Instruction: `AT:PDPW 5.5,1`
Reply: `RA 0000 AT:PDPW 5.50,1`

Name

THRS
set peakdetector threshold

Synopsis

Instruction:	`THRS <threshold>`
Reply:	`<rc> THRS <threshold>`
Instruction:	`THRS?`
Reply:	`<rc> THRS <threshold>`
Instruction:	`AT:THRS <time>,<threshold>`
Reply:	`<rc> AT:THRS <time>,<threshold>`
Instruction:	`AT:THRS? <time>`
Reply:	`<rc> AT:THRS <time>,<threshold>`
Instruction:	`ACT:THRS?`
Reply:	`<rc> ACT:THRS <threshold>`

Description

The peakdetector controls the automatic acquisition of spectra; it needs to be adjusted to the (expected) peakwidth (PKWD of the detector and AT:PDPW, page 567) and to an absorbance threshold. To allow adjustment of the peakdetector settings during a separation, both, the peakdetector peakwidth and the threshold are timeprogrammable.

Parameters

`<time>`	0 .. 99999.00 runtime in minutes, counted upward from 0 at start of run.
`<threshold>`	0.001 .. 1000.000 (mAU)

Default Values

10.000 mAU

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0302` timetable full
`RE 0304` no timetable
`RE 0502` out of range

Examples

(a) Comment: set peakdetector threshold to 1 mAU.
Instruction: `THRS 1`
Reply: `RA 0000 THRS 1.000`

23.1.3. Other Method Parameter

Name

SIGn

wavelength definition for signal A,B,C,D or E

Synopsis

Instruction:	SIGn <wl>,<bw>,<ref_wl>,<ref_bw>
Reply:	<rc> SIGn <wl>,<bw>,<ref_wl>,<ref_bw>
Instruction:	SIGn?
Reply:	<rc> SIGn <wl>,<bw>,<ref_wl>,<ref_bw>
Instruction:	AT:SIGn <time>,<wl>,<bw>,<ref_wl>,<ref_bw>
Reply:	<rc> AT:SIGn <time>,<wl>,<bw>,<ref_wl>,<ref_bw>
Instruction:	AT:SIGn? <time>
Reply:	<rc> AT:SIGn <time>,<wl>,<bw>,<ref_wl>,<ref_bw>
Instruction:	ACT:SIGn?
Reply:	<rc> ACT:SIGn <wl>,<bw>,<ref_wl>,<ref_bw>

Description

The Agilent 1200 Series DAD / MWD continuously generates 5 chromatographic signals (signal A, B, C, D and E). These signals are built out of the digitized values from the 1024 photodiodes. The SIG1 paramter setpoints define which photodiode values to use when building the signal A value, SIG2 parameter those of the signal B and so on.
Because more than one diode can be used to build a signal at a certain wavelength, the SIGn definition also allows to specify the number of diodes to be used (1 or more); this is achieved by specifying a center wavelength and a bandwidth.
The signal is defined as the difference between a sample wavelength and a reference wavelength; the reference wavelength can be switched off (zero).
The signal definition can be changed during a run by time-programming.
The signal definition is a method parameter: changing its values affects both, the method paramter setpoints and the actual signal calculations (also during the run).
During a run, the signal calculation may change due to time-programmed events. The query ACT:SIGn? always returns the effective signal calculations, whereas the SIGn? query returns the method parameter settings. At end of run, the method parameter again becomes active, independent of the latest time programmed settings.

Parameters

<wl>	190 .. 950 center wavelength in nm
<bw>	2 .. 400 bandwidth for the (sample) wavelength in nm. (190 <= <wl> - <bw>/2 <= 950)
<ref_wl>	0, 190 .. 950 center of the reference wavelength in nm 0 means: reference switched OFF.
<ref_bw>	0,2 .. 400 bandwidth for the reference wavelength in nm; must be 0, if reference is switched off (190 <= <ref_wl> - <ref_bw> / 2 <= 950)
<time>	0 .. 99999.00 runtime in minutes, counted upward from 0 at start of run

Default Values

SIG1 250, 100, 360,100
SIG2 254, 16, 360,100
SIG3 210, 8, 360,100
SIG4 230, 16, 360,100
SIG5 280, 16, 360,100

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0502 out of range
RE 0304 no timetable
RE 0302 timetable full
RE 1020 Wrong wavelength range

Examples

(a)	Comment:	Set signal A to 320nm, 20nm bandwidth, no reference
	Instruction:	`SIG1 320,20,0,0`
	Reply:	`<rc> SIG1 320,20,0,0`
(b)	Comment:	Query signal A setpoints (method parameter)
	Instruction:	`SIG1?`
	Reply:	`<rc> SIG1 320,20,0,0`
(c)	Comment:	Enter event in timetable: switch signal C to 280,6 nm, referenced agianst 340,20nm at runtime 3.7 min
	Instruction:	`AT:SIG1 3.7,280,6,340,20`
	Reply:	`RA 0000 AT:SIG1 3.7,280,6,340,20`
(d)	Comment:	Query Signal C timetable entry at time 3.7 min
	Instruction:	`AT:SIG1? 3.7`
	Reply:	`RA 0000 AT:SIG1 3.7,280,6,340,20`
(e)	Comment:	Query actual definition of signal E
	Instruction:	`ACT:SIG5?`
	Reply:	`RA 0000 ACT:SIG5 277,8,500,100`

Name

WLRG

set wavelength range

Synopsis

Instruction:	`WLRG <from>,<to>,<step>`
Reply:	`<rc> WLRG <from>,<to>,<step>`
Instruction:	`WLRG?`
Reply:	`<rc> WLRG <from>,<to>,<step>`

Description

Sets the wavelength range and stepwidth for spectra.The stepwidth defines the distance [in nm] between two adjacent data points. The total number of data points per spectrum

((<to> - <from>) / <stepwidth> + 1)

must be less or equal to 1024.

The method parameter WLRG is not timetable programmable, the WLRG? query returns the method parameter setting.

Parameters

<from>	190 .. 950: lower wavelength limit [nm]
<to>	190 .. 950: upper wavelength limit [nm]
<step>	0.01 .. 100.00 : stepwidth [nm]

Default Values

<from> : 190 nm

<to> : 400 nm

<step> : 2 nm

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 0502 out of range

RE 1020 Wrong wavelength range

Examples

(a)	Comment:	sets wavelength range from 300 to 500 nm in steps of 0.5 nm
	Instruction:	`WLRG 300,500,0.5`
	Reply:	`RA 0000 WLRG 300,500,0.5`
(b)	Comment:	query actual wavelength range setiing
	Instruction:	`WLRG?`
	Reply:	`RA 0000 WLRG 300,500,0.5`

Name

UVON <yes/no>
configure, if uv light source is required

Synopsis

Instruction:	UVON <yes/no>
Reply:	<rc> UVON <yes/no>
Instruction:	UVON?
Reply:	<rc> UVON <yes/no>

Description

Determines whether a method requires the D2-lamp or not. The module becomes NOT READY (see also ACT:NRDY?) if the D2-lamp is switched off while the UVON parameter is set to 1. The UVON? query returns the actual setting of this method parameter.

Parameters

<yes/no>	0	method does not require D2-lamp
	1	method requires D2-lamp

Default Values

1

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0502 out of range

Examples

(a)	Comment:	method should require D2-lamp
	Instruction:	UVON 1
	Reply:	RA 0000 UVON 1
(b)	Comment:	Query actual setting
	Instruction:	UVON?
	Reply:	RA 0000 UVON 1

Name

VION <yes/no>

configure, if visible light source is required

Synopsis

Instruction:	VION <yes/no>
Reply:	<rc> VION <yes/no>
Instruction:	VION?
Reply:	<rc> VION <yes/no>

Description

Determines whether a method requires the Tungsten lamp or not. The module becomes NOT READY (see also ACT:NRDY?) if the Tungsten lamp is switched off while the VION parameter is set to 1. The VION? query returns the actual setting of this method parameter.

Parameters

<yes/no>	0	method does not require Tungsten lamp
	1	method requires Tungsten lamp

Default Values

1

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 0502 out of range

Examples

(a)	Comment:	method should require Tungsten lamp
	Instruction:	VION 1
	Reply:	RA 0000 VION 1
(b)	Comment:	Query actual setting
	Instruction:	VION?
	Reply:	RA 0000 VION 1

23.2. Agilent 1200 Series DAD / MWD System Parameter

Name

TGON
Tungsten lamp configuration

Synopsis

Instruction:	`TGON <on_off>`
Reply:	`<rc> TGON <on_off>`
Instruction:	`TGON?`
Reply:	`<rc> TGON <on_off>`

Description

This parameter determines whether the Tungsten lamp will be switched on during instrument startup or not. The TGON setting is not affected when changing the method. The TGON? query returns the actual Tungsten lamp configuration.

Parameters

<on_off>	1	Tungsten lamp is switched on at startup
	0	Tungsten lamp is not switched on at startup

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0502` out of range

Examples

(a)	Comment:	configure Tungsten lamp to be switched on at startup
	Instruction:	`TGON 1`
	Reply:	`RA 0000 TGON 1`
(b)	Comment:	query actual Tungsten lamp configuration
	Instruction:	`TGON?`
	Reply:	`RA 0000 TGON 1`

Name

MWRG

Set monitor wavelength range

Synopsis

Instruction:	`MWRG <from>,<to>,<step>`
Reply:	`<rc> MWRG <from>,<to>,<step>`
Instruction:	`MWRG?`
Reply:	`<rc> MWRG <from>,<to>,<step>`

Description

Sets the wavelength range and stepwidth monitor spectra. the stepwidth defines the distance in nm between two adjacent data points. The total number of data points per spectrum

(<to> - <from>) / (stepwith + 1)

must be less or equal 1024.

The MWRG setting is not affected when changing the method, the MWRG? query returns the actual parameter setting.

Parameters

`<from>`	190 .. 950 lower wavelength limit in nm
<to>	190 .. 950 upper wavelength limit in nm
<stepwidth>	0.01 .. 100.00 stepwidth in nm

Default Values

<from>	190
<to>	400
<stepwidth>	2

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 0502 out of range

RE 1020 Wrong wavelength range

Examples

(a) Comment: set monitor wavelength range from 200 to 300 nm in steps of 2 nm

Instruction: `MWRG 200,300,2`

Reply: `RA 0000 MWRG 200,300,2`

(b) Comment: query current monitor wavelength settings

Instruction: `MWRG?`

Reply: `RA 0000 MWRG 200,300,2`

23.3. Agilent 1200 DAD / MWD Commands and Queries

Name

VIS
switch tungsten lamp on/off

Synopsis

Instruction:	`VIS <on/off>`
Reply:	`<rc> VIS <on/off>`
Instruction:	`ACT:VIS?`
Reply:	`<rc> ACT:VIS <lampstate>`

Description

Switches the Tungsten lamp on/ off. The query ACT:VIS? returns the actual lampstate.The event

ES 1023 VIS lamp: ON

is generated if the Tungsten lamp was switched on successfully. Otherwise one of the following events may occur:

EE 1018 VIS lamp: no current
EE 1019 VIS lamp: no voltage

Parameters

`<on/off>`	`0 means OFF`
	`1 means ON`
`<lampstate>`	`0 means VIS-lamp is off`
	`1 means VIS-lamp is on`

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0803 cover open violation

Examples

(a)	Comment:	switch tungsten lamp on
	Instruction:	`VIS 1`
	Reply:	`RA 0000 VIS 1`
(b)	Comment:	query actual lamp status
	Instruction:	`ACT:VIS?`
	Reply:	`RA 0000 ACT:VIS 1`

Name

`ACT:OUT?`

Query for actual signal values

Synopsis

Instruction: `ACT:OUT?`

Reply: `<rc> ACT:OUT <sigA>,<sigB>,<sigC>,<sigD>,<sigE>`

Description

Returns the actual absorbance values of all five signals.

Parameters

`sigA`	-4000.000 .. 4000.000 [mAU]

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: read actual signal absorbance of signal A .. signal E

Instruction: `ACT:OUT?`

Reply: `RA 0000 ACT:OUT 8.365,4.283,252.923,3.218,4.228`

Name

ACT:DAD?
Actual detector status

Synopsis

Instruction: `ACT:DAD?`
Reply: `<rc> ACT:DAD <generic>,<uvlamp>,<vislamp>`

Description

The actual query returns the detector state, consisting of a generic state (balance,reset,idle), the status of the UV-lamp and the status of the VIS-lamp.

Parameters

<generic>	reflects generic detector state
	0 : idle (ES 7090 Detector : IDLE)
	1 : detector performs a balance (ES 7200 Detector: PREPARE)
	2 : detector is in reset state (ES 1012 Detector : Reset)
	3 : startup
<uvlamp>	reflects UV-lamp state
	0 : UV-lamp off (ES 7403 UV lamp off)
	1 : UV-lamp on (ES 7404 UV lamp on)
	2 : UV-lamp in ignition phase (ES 7400 UV lamp ignition)
	3 : startup
<vislamp>	reflects VIS-lamp state
	0 : VIS-lamp is off (ES 1024 VIS lamp: OFF)
	1 : VIS-lamp is on (ES 1023 VIS lamp: ON)
	2 : Startup

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get actual detector status: idle and both lamps on
Instruction: `ACT:DAD?`
Reply: `RA 0000 ACT:DAD 0,1,1`

23.4. MWD and DAD Specific States

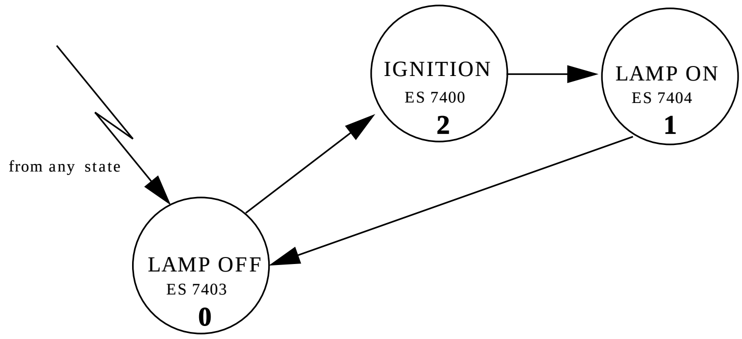
Name

UVLampSTM
Statemachine of UVLamp Behaviour

Description

This statemachines build up the UVlamp states and possible changes.

State Diagram



Name

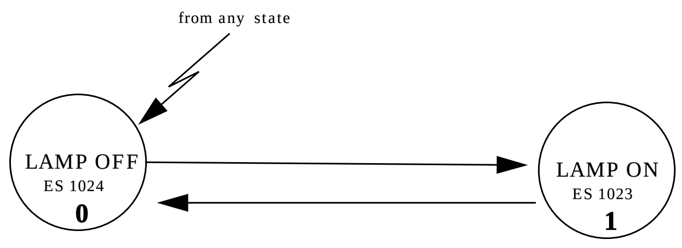
VISLampSTM

Statemachine of VIS-Lamp Behaviour

Description

This statemachines build up the VIS-lamp states and possible changes.

State Diagram



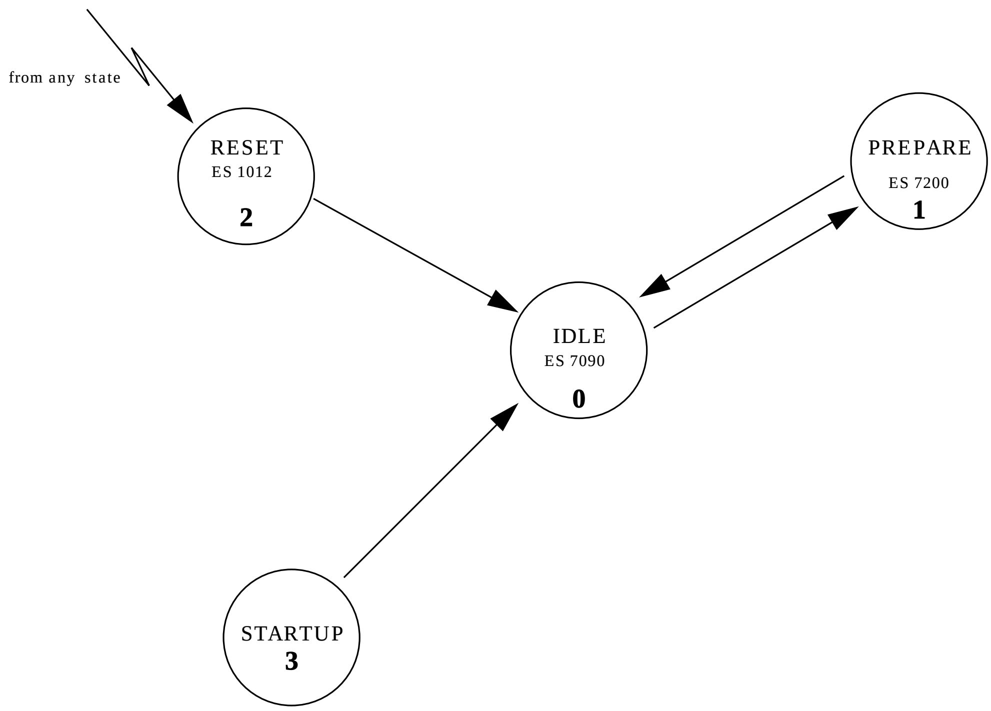
Name

DeviceSTM
Statemachine of general Detector States

Description

This statemachines build up the states and possible changes of the detector state.

State Diagram



23.5. Agilent 1200 Series DAD / MWD Rawdata File Description

This description is a supplement to: Rawdata File on page 253.
The Agilent 1200 Series DAD / MWD rawdata file records can be signal records (up to 5 different signals: RA, RB, RC, RD, RE) or spectra records (SL).

23.5.1. Signal Records

The Agielnt 1200 Series DAD / MWD generates up to 5 signals (see instructions SIG1 ... SIG5). The rawdata file can store these signals as records RA, RB, RC, RD and RE, where RA corresponds to SIG1, and so on. To select, which signal should be stored, see instruction RAWS on page 258. For selection of the data representation (decimal, hexadecimal or decimal) see instruction RAWF on page 259.

The data in the signal records are absorbance data. The absorbance data representation is 2^-21 AU per count (1 count is 2^-21 AU ~= 0.477 uAU).

23.5.2. Spectra Records

Every spectra record contains parameter information and the absorbance data. The absorbance data representation is 2^-21 AU per count (1 count is 2^-21 AU ~= 0.477 uAU).

#	description	unit	example (DEC format)
	header (always 12 bytes fix, independent of the used format)		SL DEC,0135;
	number of parameter values following (currently 21)		21,
1	time of start of analysis in seconds since 1970	s	848075283,
2	time of acquisition relative to start of analysis	ms	7605,
3	time of acquisition relative to start of run	ms	4030,
4	undefined	-	200,
5	center wavelength of first data-point	pm	190000,
6	step between two adjacent data points	pm	2000,
7	undefined	-	273,
8	undefined	-	0,
9	undefined	-	0,
10	undefined	-	0,
11	undefined	-	273,
12	undefined	-	2056,

#	description	unit	example (DEC format)
13	peakdetector status 0-3: unused 4: inflection point upslope 1 5: inflection point upslope 2 6: inflection point donwslope 1 7: inflection point donwslope 2 8: unused 9: begin of peak 10: top of peak 11: top of small peak 12: cance 13: valley 14: end of peak 15: unknown 16: force baseline 80: periodic 81: all in peak) NOTE: Although this rawdata format description also applies for the specta data out of the SPECTRA MONITOR CU, the peakdetector information is available only with the data out of the RAW-DATA CU, otherwise it is 0.		81,
14	current peakwidth (0 .. 7 , see instruction PKWD on page 564)		3,
15	peakdetector peakwidth (0 .. 3 , see instruction PKWD on page 564)		0,
16	position of u-slit used for this spectrum		4,
17	undefined	-	1,
18	undefined	-	0,
19	undefined	-	0,
20	undefined	-	0,
21	number of data points following (N)		356,
1	1. data point	2^-21AU	-2414,
...	...	...	...
N	Nth data point (last)	2^-21AU	6345

23.6. Agilent 1200 DAD / MWD Diagnose Buffers

Name

`ATMP (diag)`

array temperature of the last 24 hours.

Characteristics

max entries	288
max characters	10
storage	volatile ram, wrap-around
availability	always
storage frequency	1 every 5 minutes

Contents

temp

<temp>	array temperature in 1/1000 degC

Description

The buffer is filled continuously with the AD converted array temperature signal. The A/D conversion is part of the array readout cycle and frequency therefore depends on the actual datarate settings. These samples are bunched continuously and the mean value is stored every 5 minutes in the ATMP buffer. If the buffer is full, the oldest entry is overwritten.

Name

DARK (diag)

dark current scan of the latest dark current test

Characteristics

max entries	1024
max characters	10
storage	volatile ram
availability	always

Contents

adc-count

<adc-count>	dark current value in adc-counts for one diode

Description

Before execution of the dark current test (TEST:DARK), the diagnose buffer is cleared. After successful execution of the test, the buffer is filled with a dark current spectrum of exactly 1024 values, each of them representing the value of one diode in adc-counts

Name

FSCN (diag)

spectrum of the latest filter test

Characteristics

max entries	1024
max characters	10
storage	volatile ram
availability	always

Contents

abs_value

<abs_value>	absorbance value [in AU]

Description

The diagnosis buffer contains the result of the latest holmium-oxyde filter test (TEST:FILT) and is cleared each time a new test is started. After successful execution, the buffer contains a holmium-oxyde spectrum from 190 to 950 nm with 1nm stepwidth

Name

FSIG (diag)

trace signal of the latest filter test

Characteristics

max entries	100
max characters	15
storage	volatile ram
availability	always

Contents

signaldatat

<signaldata>	absorbance value [in AU]

Description

This buffer contains the result of the latest holmium-oxyde filter test (TEST:FILT) and is cleared each time a new test is started.During execution of the test, signal data are measured and stored in the buffer while the filter is moved into and out of the light path.

Name

INT (diag)

test intensity scan

Characteristics

max entries	1024
max characters	10
storage	volatile ram
availability	always

Contents

adc-counts

adc-counts	intensity value

Description

The buffer contains a full intensity spectrum from 190 - 950 nm with 1 nm resolution as a result from the latest TEST:INT instruction.Each buffer entry is an intensity value [in adc-counts]

Name

RCAL (diag)

detector wavelength recalibration history.

Characteristics

max entries	40
storage	FLASH
availability	always

Contents

time, alpha_offset, beta_offset

<time>	seconds since 1970, the time when the recalibration was done
<alpha_offset>	offset of D2-alpha line(in pm)
<beta_offset>	offset of D2- beta line (in pm)

Description

Every time the detector wavelength is recalibrate (WCAL instruction) , an entry in buffer WCAL is made. If the buffer is full, the oldest entry is overwritten.

Name

SCAN (diag)

spectrum of the latest test scan

Characteristics

max entries	1024
max characters	15
storage	volatile ram
availability	always

Contents

abs_value

<abs_value>	absorbance value in AU

Description

This buffer contains the result of the latest scan test (TEST:SCAN) and is cleared each time a new test is started.After successful execution of the test, the buffer is filled with the measured absorbance spectrum.

Name

SLIT (diag)

Microslit test buffer

Characteristics

max entries	15
max characters	30
storage	volatile ram
availability	always

Contents

signal1,signal2,signal3

signal 1	intensity value at 300 nm
signal 2	intensity value at 600 nm
signal 3	intensity value at 800 nm

Description

This diagnose buffer contains the results of the latest microslit test initiated with the TEST:SLIT instruction and is cleared each time a new test is started.Each buffer entry is a tripel of intensity values measured at same time at different wavelengths

Name

UVIG (diag)

ignition course of the last ignition cycle of the deuterium lamp

Characteristics

max entries	200
max characters	35
storage	volatile
availability	always
storage frequency	on every ignition cycle: 10Hz until buffer is fullz

Contents

anode_voltage,anode_current,heater_voltage,heater_current

anode_voltage,	anode voltage in µV
anode_current,	anode current in µA
heater_voltage,	heater voltage in µV
heater_current,	heater current in µA

Description

Ignition course with values above of the last ignition cycle of the deuterium lamp.

Name

UV_V (diag)

anode voltage course of the D2 lamp during the last minute

Characteristics

max entries	500
max characters	10
storage	volatile, wrap-around
availability	always
storage frequency	10 Hz

Contents

anode_voltage

anode_voltage,	anode voltage in µV

Description

The buffer is filled continuously with the AD converted signal of the anode voltage at a data rate of 10Hz. If the buffer is full, the oldest entry is overwritten.

Name

UV_1 (diag)

course of the D2 lamp during the last two hours (anode and heater voltage/current)

Characteristics

max entries	120
max characters	35
storage	volatile, wrap-around
availability	always
storage frequency	1 per minute

Contents

anode_voltage,anode_current,heater_voltage,heater_current

anode_voltage,	anode voltage in µV
anode_current,	anode current in µA
heater_voltage,	heater voltage in µV
heater_current,	heater current in µA

Description

The buffer is filled continuously with the AD converted signals (voltage and current of anode and heater) at a data rate of 1 per minute. If the buffer is full, the oldest entry is overwritten.

Name

VISC (diag)

current course of the tungsten lamp during the last minute

Characteristics

max entries	500
max characters	10
storage	volatile, wrap-around
availability	always
storage frequency	10 Hz

Contents

lamp_current

lamp_current,	tunsten lamp current in µA

Description

The buffer is filled continuously with the AD converted signal lamp current) at a data rate of 10Hz. If the buffer is full, the oldest entry is overwritten.

Name

VISO (diag)

course of the latest turn-on cycle of the tungsten lamp

Characteristics

max entries	50
max characters	35
storage	volatile
availability	always
storage frequency	on every turn on: 10Hz until buffer is full

Contents

lamp_voltage, lamp_current

lamp_voltage,	tungsten lamp voltage in µV
lamp_current,	tungsten lamp current in µA

Description

Before switching on the tungsten lamp, the diagnose buffer is cleared. During the switch-on phase of the tungsten lamp the buffer is filled with the AD converted signals (voltage and current) at a data rate of 10 Hz, until the buffer is full.

Name

VIS1 (diag)

course of the tungsten lamp during the last two hours (voltage/current)

Characteristics

max entries	120
max characters	35
storage	volatile, wrap-around
availability	always
storage frequency	1 per minute

Contents

anode_voltage,anode_current,heater_voltage,heater_current

lamp_voltage,	tungsten lamp voltage in µV
lamp_current,	tungsten lamp current in µA

Description

The buffer is filled continuously with the AD converted signals (voltage and current) at a data rate of 1 per minute. If the buffer is full, the oldest entry is overwritten.

Name

WCAL (diag)

Result of the latest wavelength recalibration test

Characteristics

max entries	1
max characters	10
storage	flash
availability	always

Contents

time, act_a, act_b, base_a, base_b

time	seconds in 1970, the time when whe latest test was executed.
act_a	measured actual deviation of D2-alpha line (in picometer).
act_b	measured actual deviation of D2-beta line (in picometer).
base_a	measured absolute deviation of D2-alpha line (in picometer)
base_b	measured absolute deviation of D2-beta line (in picometer).

Description

The diagnose buffer WCAL always contains the result of the latest wavelength recalibration test (see TEST:WCAL instruction). The four numbers are the actual and absolute D2 alpha and beta line positions.

23.7. Agilent 1200 DAD / MWD Diagnose Counters

Name

D_SW (diag)

D2-lamp ignition counter

storage type: non erasable
history size : 20
limit event : EF 7474 UV-lamp ignition cycles exceeded

Increments by one with each succeessful ignition of the D2-lamp. The erasable part of the counter is set to 0 with each coldstart.

Name

V_SW (diag)

Tungsten lamp switch on counter

storage type: non erasable
history size: 20
limit event : EF 7475 VIS- lamp ignition cycles exceeded (only MWD, DAD and SPM)

Increments by one each time the tungsten lamp is switched on.The erasable part of the counter is set to 0 witch each coldstart.

Name

D_TI (diag)

Accumulated D2-lamp burn time

storage type: non-erasable
history size: 20
limit event: EF 7470 UV- lamp life-time exceeded

Holds the total burn time in seconds of the D2-lamp. Increments by one each second the D2-lamp is burning.The erasable part of the D_TI counter is set to zero at instrument coldstart.

Name

V_TI (diag)

Accumulated tungsten lamp burn time

storage type : non-erasable
history size: 20
limit event : EF 7471 VIS- lamp life-time exceeded (only MWD, DAD and SPM)

Holds the total burn time in seconds of the Tungsten lamp.Increments by one with each second the Tungsten lamp is burning.The erasable part of the V_TI counter is set to zero at instrument cold-start.

Name

D_ON (diag)

D2-lamp ontime

storage type : volatile
limit event : EF 7472 UV- lamp on-time exceeded

Provides time in seconds the D2-lamp is burning since switched on.Increments by one with each second the D2-lamp is burning.

Name

V_ON (diag)

D2-lamp ontime

stotage type : volatile
limit event : EF 7473 VIS- lamp on-time exceeded (only MWD, DAD and SPM)

Provides time in seconds the Tungsten lamp is burning since switched on.Increments by one with each second the Tungsten lamp is burning.

Name

USLT (diag)

Slit movement counter

storage type : non-erasable
limit event: EF 1100 u-Slit movement counter limit exceeded (only MWD and DAD)
history size: 5

Increments by one each time the slit is moved.The erasable part of the counter is set to zero at instrument coldstart.

Name

FILT (diag)

Filter movement counter

storage type: non-erasable
history size: 5
limit event: EF 1101 Holmium-oxyde filter movement counter limit exceeded (only MWD and DAD)

Increments by one each time the filter is moved. The erasable part of the counter is set to zero at instrument coldstart.

23.8. Agilent 1200 DAD / MWD and SPM Spectrometer Id & Recalibration Commands

Name

SPID

set spectrophotometer identification data in EEPROM on spectrophotometer

Synopsis

Instruction: `SPID "<id>", "<data>"`
Reply: `<rc> SPID "<id>", "<data>"`
Instruction: `SPID? "<id>"`
Reply: `<rc> SPID "<id>", "<data>"`

Description

This instruction is used to write, read or modify the information on the spectrophotometer EEPROM. This instruction is password protected (see instruction SSPW). The information on the EEPROM is separated in blocks, each identified by a two character (ASCII) <id>. Every block may hold one or more bytes (255 max). To get a list of all block-ids currently stored in the EEPROM, query SPID? "" (<id>: null-string).
To write a new block of informaton to the EEPROM, use e.g SPID "SN", "1234G1399A3456". If the block with the id 'SN' already exists, it is overwrittten.
To read a block of information use e.g. SPID? "SN". SPID "XX","" is returned, if 'XX' does not exist.
Deleting a block is done by writing a null-string to it. (e.g. SPID "SN","")
Clearing the whole EEPROM is done by writing a null-string to block "" (SPID "","")
This instruction needs special capabilities (password, key)
Unless otherwise noted, the information in a block is not interpreted by the device.

The following<id> are or may be interpreted by the device:
Cx:
The information of all blocks, that have an <id> starting with 'C' are used for wavelength calibration. The data in every block always are a sequence of one or more floating point numbers in the range +/-8000000E+/-127, separated by commas. CA holds the alpha and beta line position, any other 'C.' block holds the coefficients of one wavelength section, including the wavelength range. For more information see DSP documentation.

The following <id> are reserved:
SN: spectrophotometer serail number
CA, C0, C1,C2: WL-Scale
DT: Date of manufacturing (mmddyy)
TL: Tooling of manuf. (X1, X2, ...)
OF: Order filter version (V1,V2, ...)
AP: Aperture F/# (F/2, F/4)
SC: Parameter scatter correction (0, nnn)
DC: Date of WL calibration (mmddyy)
SL: slit type (uSlit, 25u)

Not used, but reserved:
WL: wavelength range in pm (from, to, # of diodes used) -1: undefined
SW: slitwidth in pm (for micromechanic slit: slitwidth of pos 0, pos1, ...).
Slitwidth=0: closed, slitwidth=-1: undefined or reference position)
AR: array info (e.g. serial number)
AE: array exposure time

Parameters

<id>	two bytes, ASCII, no control characters null-string has special meaning
<data>	character string (ASCII) null-string has special meaning

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0101 not allowed (not in supervisor state)
RE 0103 wrong parameter (parameter range error)
RE 0104 full (if EEPROM is full)

Examples

(a) Comment: query spectrophotometer serial number
Instruction: `SPID? "SN"`
Reply: `RA 0000 SPID 1,"SN","1234G1234A00567"`

(b) Comment: set spectrophotometer slit data
Instruction: `SPID "SW","01"`
Reply: `RA 0000 SPID "SW","01"`

Name

`SSPW`

Password for spectrometer identification data in EEPROM

Synopsis

Instruction: `SSPW "<password>"`

Reply: `<rc> SSPW`

Description

Permits writing data to the EEPROM (see SPID command), if password is given. Once set, the module stays in this supervisor state until the next power cycle.

Parameters

<password> max. 80 characters in double quotes

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Name

WCAL

Set new wavelength calibration offset

Synopsis

Instruction:	`WCAL <alpha_offset>,<beta_offset>`
Reply:	`<rc> WCAL <alpha_offset>,<beta_offset>`
Instruction:	`WCAL?`
Reply:	`<rc> WCAL <alpha_offset>,<beta_offset>`

Description

WCAL allows to recalibrate the spectrophotometer using the fixed wavelengths of the alpha and beta lines of the deuterium lamp. WCAL specifies the offset to be used relative to the factory based spectrophotometer calibration.

Use TEST:WCAL instruction to measure the actual offset of the alpha and beta lines (relative to the factory calibration).The WCAL? query returns the currently used wavelength calibration offset D2-(alpha andD2- beta line

Parameters

<alpha_offset>	offset of D2-alpha line(in pm)
<beta_offset>	offset of D2- beta line (in pm)

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	query actual wavelength calibration offsets
	Instruction:	`WCAL?`
	Reply:	`RA 0000 WCAL 143,-50`
(b)	Comment:	set new wavelength calibration offset
	Instruction:	`WCAL 143,-50`
	Reply:	`RA 0000 WCAL 143,-50`

23.9. Agilent 1200 DAD / MWD Test Instructions

Name

TEST:FILT
Holmium filter test

Synopsis

Instruction: TEST:FILT
Reply: <rc> TEST:FILT

Description

Performs a test to verify the functionality of the built-in holmium filter and stores results in the diagnose buffer FSIG and FSCN. A signal is sampled with 20Hz while the filter is moved into and out of the light path.The signal data are stored in the diagnose buffer FSIG. Also a test spectrum is measured when the filter is within the light path, which is stored in the diagnose buffer FSCN. Signal A is used with 400 nm sample wavelength and 10 nm bandwidth without reference wavelength.The test spectrum uses the full wavelength range 190 - 950 nm at 1nm resolution. At begin of the test peakwidth is set to 0, the slitwidth is set to 1 nm and a complete balance cycle is done, at the end of the test all parameters are restored and the detector balances automatically.

The test fails if no reference position for the filter could be found, e.g. if both lamps are off, the slit is closed or the light path is blocked. In this case FSIG and FSCN are empty and the error event
EE 1050 Holmium filter test failed
is sent.
The filter test is not allowed during balance or while another instrument test is in progress. Also the filter test is rejected during the startup phase or if a general instrument error (e.g. LEAK) occurred.

Parameter

<none>

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 Test not allowed during startup
RE 6072 Test not allowed while shutdown
RE 6063 Test not allowed while other test is active
RE 6064 Test not allowed during prepare
RE 6071 Test not allowed while leak

Example

(a) Comment: Perform filter test
Instruction: TEST:FILT
Reply: <rc> TEST:FILT
A successful test will generate the events listed below:
ES 0112 test
EV 1164 Holmium-oxyde filter test started
ES 7200 Detector: PREPARE
ES 7090 Detector : IDLE
ES 0113 no test

Name

TEST:SLIT
Microslit test

Synopsis

Instruction: TEST:SLIT
Reply: <rc> TEST:SLIT

Description

Performs a test to verify the functionality of the u-slit.Three intensity signals are measured using all possible slitwidths including the closed position, beginning with 16nm to 8 , 4 , 2 and 1nm, the closed position and again up to 16nm slitwidth.The signals are taken at 300nm, 600 nm and 800 nm and stored in the diagnose buffer SLIT . Before starting the test, a complete calibration cycle(gain, dark current, background) is done with 16 nm slitwidth and a 240Hz array readout program. After the test all parameters are restored and the detector balances automatically.

Parameter

<none>

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 Test not allowed during startup
RE 6072 Test not allowed while shutdown
RE 6063 Test not allowed while other test is active
RE 6064 Test not allowed during prepare
RE 6071 Test not allowed while leak

Example

(a) Comment: Perform slit test
Instruction: TEST:SLIT
Reply: <rc> TEST:SLIT
A successful test will generate the following events:
ES 0112 test
EV 1165 Micro-slit test started
ES 7200 Detector: PREPARE
ES 7090 Detector : IDLE
ES 0113 no test

Name

TEST:DARK
Measure dark current

Synopsis

Instruction: TEST:DARK
Reply: <rc> TEST:DARK

Description

A dark current spectrum is measured and stored in the diagnose buffer DARK. The test uses a 20-Hz array readout program (#9), sets the gain to its maximum, clears the dark table and switches of the calculation of logarithmn. The resulting spectrum contains 1024 values [counts] and is non-linearized, which means that each value represents one diode.
After finishing the test, all parameters are restored and the detector balances automatically.

Parameter

<none>

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 Test not allowed during startup
RE 6072 Test not allowed while shutdown
RE 6063 Test not allowed while other test is active
RE 6064 Test not allowed during prepare
RE 6071 Test not allowed while leak

Example

(a) Comment: Measure dark current
Instruction: TEST:DARK
Reply: <rc> TEST:DARK
A successful test will generate the events listed below:
ES 0112 test
EV 1161 Dark current test started
ES 7200 Detector: PREPARE
ES 7090 Detector : IDLE
ES 0113 no test

Name

TEST:INT
Measure intensity spectrum

Synopsis

Instruction: TEST:INT
Reply: <rc> TEST:INT

Description

Measures an intensity spectrum using 1 nm slitwidth, the 20-Hz array readout program #4 and fixed gain 0. The resulting intensity spectrum is stored in the diagnose buffer "INT".The test also switches off the calculation of logarithm, sets the wavelength range from 190 to 950 nm at 1 nm resolution and sets peakwidth to 0.
After finishing the test, all parameters are restored and the detector balances automatically.

Parameter

<none>

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 Test not allowed during startup
RE 6072 Test not allowed while shutdown
RE 6063 Test not allowed while other test is active
RE 6064 Test not allowed during prepare
RE 6071 Test not allowed while leak

Example

(a) Comment: Measure intensity spectrum
Instruction: TEST:INT
Reply: <rc> TEST:INT
A successful test will generate the events listed below:
ES 0112 test
EV 1160 Intensity test started
ES 7200 Detector: PREPARE
ES 7090 Detector : IDLE
ES 0113 no test

Name

TEST:SCAN
Measure test absorbance spectrum

Synopsis

Instruction: TEST:SCAN
Reply: <rc> TEST:SCAN
Instruction: SCAN
Reply: <rc> SCAN

Description

For test purposes an absorbance spectrum is measured. TEST:SCAN sets the wavelength range to 190 - 950 nm at 1 nm resolution and stores the result in the diagnose buffer SCAN. After finishing the test all parameters are restored
The SCAN command uses the actual wavelength settings and stores the result in the diagnose buffer SCN.

Parameter

<none>

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 Test not allowed during startup
RE 6072 Test not allowed while shutdown
RE 6063 Test not allowed while other test is active
RE 6064 Test not allowed during prepare
RE 6071 Test not allowed while leak

Example

(a) Comment: Measure test absorbace spectrum
Instruction: TEST:SCAN
Reply: <rc> TEST:SCAN
TEST:SCAN will generate the events listed below:
ES 0112 test
EV 1163 Test scan started
ES 0113 no test
and SCAN will generate following events:
ES 0112 test
ES 0113 no test

Name

TEST:DAC

Turns on or switch off the DAC test mode.

Synopsis

Instruction: TEST:DAC <on/off>

Reply: <rc> TEST:DAC <on/off>

Instruction: TEST:DAC?

Reply: <rc> TEST:DAC <on/off>

Description

This function turns the detector in the DAC test mode (testing the analog output) or switch off this mode and returns to normal operation.

This test mode must be terminated by the user.

A fix pattern is send to the analog output. This pattern and a specified noise must be measured at the analog output.

The signal is a rectangle:

Height is 10µV +/- 2µV, Noise must be <= 3µVpp, Absolute value is depending at Zero Offset.

This must be controlled and passed by a external device like a integrator.

During test mode no analysis can be start.

Parameters

<on/off> 0 for off, 1 for on

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 test while startup
RE 6063 test while test
RE 6064 test while prep
RE 6071 test while leak
RE 6072 test while shutdown
RE 6075 not allowed during test
RE 6077 test while abort

Examples

(a) Comment: set the detector into the DAC test mode

Instruction: TEST:DAC 1

Reply: RA 0000 TEST:DAC 1

A successful test will generate the following events:

ES 0112 test
EV 7372 DAC test started
ES 0113 no test

Name

TEST:WCAL

Measures actual D2 line positions

Synopsis

Instruction: TEST:WCAL

Reply: <rc> TEST:WCAL

Description

Measures the actual D2 line positions (D2-alhpa and D2-beta) of the deuterium lamp and determines their deviations (in picometer). The result is put to the diagnose buffer 'WCAL'. Two different informations are given:

1. The deviation of the measured line positions from the actual recalibration. This deviation is called the 'actual' deviation.

2. The deviation of the measured line positions from the factory calibration. This deviation is called the 'absolute' deviation.

To recalibrate the spectrophotometer, use instruction WCAL, which expects the 'absolutee' deviation as its parameters.
TEST:WCAL generates an error event, if determination of the D2 line failed (eg. if the D2 lamp was not switched ON).
The test uses the 1-nm slit, sets peakwidth to 0 and does a calibration with fix gain 0 at the beginning.

Parameters

<act_a>	measured actual deviation of D2-alpha line (in picometer).
<act_b>	measured actual deviation of D2-beta line (in picometer).
<base_a>	measured absolute deviation of D2-alpha line (in picometer)
<base_b>	measured absolute deviation of D2-beta line (in picometer).

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 test while startup
RE 6063 test while test
RE 6064 test while prep
RE 6071 test while leak
RE 6072 test while shutdown
RE 6075 not allowed during test
RE 6077 test while abort

Example

(a) Comment: Returns measured deviations of D2 line wavelength positions
Instruction: TEST:WCAL
Reply: <rc> TEST:WCAL

A successful test will generate the events listed below:

ES 112 test begin
EV 1162 Wavelength calibration test started
ES 113 test end

24. Variable Wavelength Detector (G1314A/B/C)

24.1. VWD Method Parameter

Name

`PKWD`

Sets peakwidth of smallest peak expected

Synopsis

Instruction:	`PKWD <peakwidthnum>`
Reply:	`<rc> PKWD <peakwidthnum>`
Instruction:	`PKWD?`
Reply:	`<rc> PKWD <peakwidthnum>`
Instruction:	`ACT:PKWD?`
Reply:	`<rc> ACT:PKWD <peakwidthnum>`
Instruction:	`AT:PKWD <time>, <peakwidthnum>`
Reply:	`<rc> AT:PKWD <time>, <peakwidthnum>`
Instruction:	`AT:PKWD? <time>`
Reply:	`<rc> AT:PKWD <time>, <peakwidthnum>`

Description

The PKWD setpoint is the peakwidth at half hight of the smallest peak expected.
The setpoint controls the singal filtering.

Parameters

`<peakwidthnum>` 0 .. 7 (G1314A/B), -2..7 (G1314C)

<table>
<tr><th>PKWD</th><th>peakwidth
at half hight</th><th>rise time
[10 .. 90 %]</th><th>signal
data rate</th><th>signal
slicewidth</th></tr>
<tr><td>-2
(G1314C only)</td><td>< 0.00125 min</td><td>< 0.03 sec</td><td rowspan="11">see
DRAT</td><td rowspan="11">see
DRAT</td></tr>
<tr><td>-1
(G1314C only)</td><td>>= 0.00125 min</td><td>0.03 sec</td></tr>
<tr><td>0
(G1314C)</td><td>>= 0.0025 min</td><td>0.06 sec</td></tr>
<tr><td>0
(G1314A/B)</td><td>< 0.005 min</td><td>< 0.12 sec</td></tr>
<tr><td>1</td><td>>= 0.005 min</td><td>0.12 sec</td></tr>
<tr><td>2</td><td>>= 0.010min</td><td>0.25 sec</td></tr>
<tr><td>3</td><td>>= 0.025 min</td><td>0.5 sec</td></tr>
<tr><td>4</td><td>>= 0.05 min</td><td>1 sec</td></tr>
<tr><td>5</td><td>>= 0.1 min</td><td>2 sec</td></tr>
<tr><td>6</td><td>>= 0.2 min</td><td>4 sec</td></tr>
<tr><td>7</td><td>>= 0.4 min</td><td>8 sec</td></tr>
</table>

Default Values

5 (risetime: 2 sec)

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: set peakwidth parameter
Instruction: `PKWD 6`
Reply: `RA 0000 PKWD 6`

Name

DRAT

Set the output data rate of the detector.

Synopsis

Instruction:	`DRAT <data_rate_num>`
Reply:	`<rc> DRAT <data_rate_num>`
Instruction:	`DRAT?`
Reply:	`<rc> DRAT <data_rate_num>`

Description

This instruction set the data output rate of the detector to the workstation
This function often is used together with the PKWD command.
If timetable entry of PKWD is used the datarate should be combined to smallest entry of PKWD.
If no timetable entry is used data_rate_num can be the same value as peakwidthnum
(see PKWD instruction).
The Datarate cannot be changed during RUN mode of the detector.

Parameters

`<data_rate_num>`	-2 for 54.95 Hz datarate (signal slicewidth is 18.2 msec, G1314C only)
	-1 for 27.47 Hz datarate (signal slicewidth is 36.4 msec, G1314C only)
	0 for 13.74 Hz datarate (signal slicewidth is 72.8 msec, G1314A/B/C)
	1 for 13.74 Hz datarate (signal slicewidth is 72.8 msec, G1314A/B/C)
	2 for 13.74 Hz datarate (signal slicewidth is 72.8 msec, G1314A/B/C)
	3 for 13.74 Hz datarate (signal slicewidth is 72.8 msec, G1314A/B/C)
	4 for 6.87 Hz datarate (signal slicewidth is 145.6 msec, G1314A/B/C)
	5 for 3.43 Hz datarate (signal slicewidth is 291.2 msec, G1314A/B/C)
	6 for 1.72 Hz datarate (signal slicewidth is 582.4 msec, G1314A/B/C)
	7 for 0.86 Hz datarate (signal slicewidth is 1164.8 msec, G1314A/B/C)

Default Values

5 (3.43 Hz, 291.2 msec)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0305` not allowed during run

Examples

(a)	Comment:	set data rate to 6.87 Hz
	Instruction:	`DRAT 4`
	Reply:	`RA 0000 DRAT 4`
(b)	Comment:	try to set data rate setting during a run
	Instruction:	`DRAT 3`
	Reply:	`RE 0305 DRAT 3`

Name

DDLY
Switch data delay of output data on or off.

Synopsis

Instruction: `DDLY <on/off>`
Reply: `<rc> DDLAY <on/off>`
Instruction: `DDLY?`
Reply: `<rc> DDLAY <on/off>`

Description

This instruction switch the delay of data output to workstation and to analog output on or off.
The delay of the data output has the advantage that a change or a switch of the responsetime PKWD has no influence at the retention time of the sample. On the other side there was always a delay of the data output and the measurement in the cell. For making stop-flow scans it is better to have a undelayed behaviour of the detector. With this instruction the delay of the internal digital filter can be switched on or off.
The delay switch cannot be changed during RUN mode of the detector.
For lowest peakwidth the difference between delayed data or undelayd data is about 10 to 12 seconds.

Parameters

`<on/off>`	0 for off = data not delayed (realtime output) 1 for on = data delayed (retention time constant)

Default Values

0 (not delayed)

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0305 not allowed during run

Examples

(a) Comment: switch data delay off
Instruction: `DDLY 0`
Reply: `RA 0000 DDLY 0`

Name

SIG1

Select wavelength

Synopsis

Instruction:	`SIG1 <wavelength>`
Reply:	`<rc> SIG1 <wavelength>`
Instruction:	`SIG1?`
Reply:	`<rc> SIG1 <wavelength>`
Instruction:	`ACT:SIG1?`
Reply:	`<rc> ACT:SIG1 <wavelength>`
Instruction:	`AT:SIG1 <time>, <wavelength>`
Reply:	`<rc> AT:SIG1 <time>, <wavelength>`
Instruction:	`AT:SIG1? <time>`
Reply:	`<rc> AT:SIG1 <time>, <wavelength>`

Description

This instruction set the grating of the variable wavelength detector to the desired position.

Parameters

`<wavelength>` 190 .. 600 [nm]

Default Values

254

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set 400 nm
Instruction: `SIG1 400`
Reply: `RA 0000 SIG1 400`

(b) Comment: query actual wavelength setting
Instruction: `SIG1?`
Reply: `RA 0000 SIG1 400`

(at DaVinci sigtype 1,2,3,100,101,102,103,213,214,215 is accessable)

Name

ACH1|ACH2
select the source channel for analog output

Synopsis

Instruction:	`ACH1 <sigtype>`
Reply:	`<rc> ACH1 <sigtype>`
Instruction:	`ACH2 <sigtype>`
Reply:	`<rc> ACH2 <sigtype>`
Instruction:	`ACH1?`
Reply:	`<rc> ACH1 <sigtype>`
Instruction:	`ACH2?`
Reply:	`<rc> ACH2 <sigtype>`

Description

With this instruction a signal described in special section (see page 640) can be routed to the analog output.
The VWD only offers one analog output. Only ACH1 instruction is used.

Parameters

`<sigtype>` 1 .. 999

Default Values

1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 6811` unknown signal type

Examples

(a) Comment: switch signal type 3 to analog output 1
Instruction: `ACH1 3`
Reply: `RA 0000 ACH1 3`

(b) Comment: query the actual channel of the anlog output 1
Instruction: `ACH1?`
Reply: `RA 0000 ACH1 3`

Name

POL

Set polarity of rawdata and analog output

Synopsis

Instruction:	`POL <polarity>`
Reply:	`<rc> POL <polarity>`
Instruction:	`POL?`
Reply:	`<rc> POL <polarity>`

Description

This instruction set the polarity of data handling for the detector. For positive polarity (0=normal = default) the detector output positive data if detecting higher absorbance compared to the value during balance. With negative polarity (1) the detctor output positive data if detecting lower absorbance compared to the value during balance.
This function affects all data generated from SIG1 or SIGn.
This function affects only signal types below or equal 100 defined by CHA1 or CHAn instruction. (only chrommatighraphic data are affected!)

Parameters

`<polarity>` 0 for positive polarity, 1 for negative polarity

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set negative polarity
Instruction: `POL 1`
Reply: `RA 0000 POL 1`

Name

UVON

D2-lamp required

Synopsis

Instruction:	`UVON <yes_no>`
Reply:	`<rc> UVON <yes_no>`
Instruction:	`UVON?`
Reply:	`<rc> UVON <yes_no>`

Description

Determines whether a method requires the D2-lamp or not. The instrument becomes NOT READY (see also ACT:NRDY?) if the D2-lamp is switched off while the UVON parameter is set to 1. The UVON? query returns the actual setting of this method parameter.

Parameters

<yes_no>	0	method does not require D2-lamp
	1	method requires D2-lamp

Default Values

1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	method should require D2-lamp
	Instruction:	`UVON 1`
	Reply:	`RA 0000 UVON 1`
(b)	Comment:	Query actual setting
	Instruction:	`UVON?`
	Reply:	`RA 0000 UVON 1`

Name

SCAN

Scan variables for default scan

Synopsis

Instruction:	`SCAN <scan_type>,<wlfrom>,<wlto>,<wlinc>`
Reply:	`<rc> SCAN <scan_type>,<wlfrom>,<wlto>,<wlinc>`
Instruction:	`SCAN?`
Reply:	`<rc> SCAN <scan_type>,<wlfrom>,<wlto>,<wlinc>`

Description

This variables are used to take a scan without wavelength parameters.
If scan is taken with command: SCAN:TAKE 0 these wavelength settings are used for scan.

Parameters

<scan_type>	0 .. 4
<wlfrom>	190 .. 600 [nm] for scantype 0, 1
<wlto>	190 .. 600 [nm] for scantype 0, 1
<wlinc>	1.. 10 [nm]

Default Values

0, 190, 400, 2

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set interesting wavelength area
	Instruction:	`SCAN 0,350,450,1`
	Reply:	`RA 0000 SCAN 0,350,450,1`

24.2. VWD System Parameter

Name

LTYP

Deuterium lamptype

Synopsis

Instruction:	`LTYP <lamp_type>`
Reply:	`<rc> LTYP <lamp_type>`
Instruction:	`LTYP?`
Reply:	`<rc> LTYP <lamp_type>`

Description

This setting are used for different types of D2-lamps used in the VWD.

Parameters

<lamp_type>	0	lamp: 2140-0590 (heat only during ignition)
	1	lamp: G1314-60100 (continous heating) (default)

Default Values

1

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: set lamp type for Koto lamp

Instruction: `LTYP 1`

Reply: `RA 0000 LTYP 1`

24.3. VWD Control Commands

24.3.1. Calibration Control Commands

Name

`WCAL`
Set the wavelength calibration
`WCAL?`
Showes the last calibration results

Synopsis

Instruction:	`WCAL`
Reply:	`<rc> WCAL`
Instruction:	`WCAL?`
Reply:	`<rc> WCAL <wl_gap_0>,<wl_gap_656>,<wl_gap_comp>`
Instruction:	`WCAL? <cal_type>`
Reply:	`<rc> WCAL <cal_type>,<wl_gap>`
Instruction:	`WCAL:MOT?`
Reply:	`<rc> WCAL:MOT <mot_0>,<mot_656>,<mot_361>,` `<mot_419>,<mot_486>,<mot_537>`
Instruction:	`WCAL:WGAP?`
Reply:	`<rc> WCAL:WGAP <wl_gap_0>,<wl_gap_656>,<wl_gap_361>,` `<wl_gap_419>,<wl_gap_486>,<wl_gap_537>`
Instruction:	`WCAL:INTE?`
Reply:	`<rc> WCAL:INTE <intensity_0>,<intensity_656>`

Description

WCAL 1 itself adjust the values find with the TEST:WCAL algorithm to the optical unit.
After setting of values correct EV 7311,3 is send out.

Parameters

<cal_type>	0 for 0 order calibration 1 for 656nm calibration 2 for wavelength compensation
<wl_gap_...>	-100.0 .. 100.0 wl gap of the last calibration in nm
<mot_...>	0 .. 32000 value showes the resulting motor-steps after calibration
<intensity_...>	0 .. 2^31 intensity = adc-counts of reference diode of the last calibration

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 6310` calibration set without calibration test
`RE 6311` compensation set without compensation test

Examples

(a)	Comment:	set calibration of the result of last TEST:WCAL
	Instruction:	`WCAL`
	Reply:	`RA 0000 WCAL`
(b)	Comment:	show the result of the last calibration
	Instruction:	`WCAL?`
	Reply:	`RA 0000 WCAL 0.5,0.8,0.2`

Name

CALS

Does a specific wavelength calibration

CALS?

Showes the last calibration type and result

Synopsis

```
Instruction:  CALS &lt;caltype&gt;
Reply:        &lt;rc&gt; CALS &lt;caltype&gt;
Instruction:  CALS?
Reply:        &lt;rc&gt; CALS &lt;caltype&gt;,&lt;previous_mot&gt;,&lt;previous_int&gt;,
                      &lt;previous_wl_gap&gt;,&lt;actual_mot&gt;,&lt;actual_int&gt;,
                       &lt;actual_wl_gap&gt;
Instruction:  CALS? &lt;caltype&gt;
Reply:        &lt;rc&gt; CALS &lt;caltype&gt;,&lt;previous_mot&gt;,&lt;previous_int&gt;,
                      &lt;previous_wl_gap&gt;,&lt;actual_mot&gt;,&lt;actual_int&gt;,
                       &lt;actual_wl_gap&gt;
```

Description

CALS sets the result of the wavelength calibration search (started by TEST:CALS the detector. The result of the calibration are shown with the CALS? instruction in the responding values.
The <previous_mot> is the stepper motor result of the last calibration cycle, the <actual_mot> is the result of the current calibration cycle. If both values are the same, exact same wavelength is occured. The given <previous_int> and <actual_int> is the corresponding intensity at the end of the calibration. <previous_wl_gap> and <actual_wl_gap> are the wavelength gap in 0.1 nm.
After restarting calibration again the next <previous_...> is the previous <actual_...>.
All values will be independent stored for 0th order calibration and for 656nm calibration.
CALS? shows the values of the last calibration that executed.
For setting the start condition of the test the motor step value be set by the SWLC instruction.
See also TEST:CALS (page 668).

Parameters

<caltype>	0 for 0th order calibration / 1 for calibration at 656nm
<previous_mot>	0 .. 32000 value showes the resulting motor-steps of the previous calibration
<actual_mot>	0 .. 32000 value showes the resulting motor-steps after calibration
<previous_int>	0 .. 2^31 intensity = adc-counts of reference diode of the previous calibration
<actual_int>	0 .. 2^31 intensity = adc-counts of reference diode after calibration
<previous_wl_gap>	0 .. 100.0 wl gap of the previous calibration in nm
<actual_wl_gap>	0 .. 100.0 wl gap of the last calibration in nm

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6310 calibration set without calibration test

Examples

(a) Comment: show the result of the last 0-order calibration
Instruction: `CALS? 0`
Reply: `RA 0000 CALS 0,21110,134432,0.3,21112,128763,0.5`

Name

`WCMP`

Does a wavelength compensation

`WCMP?`

Showes the last compensation results

Synopsis

Instruction:	`WCMP`
Reply:	`<rc> WCMP`
Instruction:	`WCMP?`
Reply:	`<rc> WCMP <mot_361>,<mot_419>,<mot_486>,<mot_537>,` `<wlgap_361>,<wlgap_419>,<wlgap_486>,<wlgap_537>`

Description

Parameters

<mot_...>	0 .. 32000 motor steps of given wavelengths
<wlgap_...>	0 .. 100.0 wavelength gap before compensation in nm

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

`RE 6311` compensation set without compensation test

Examples

(a) Comment: show the result of the last wavelength compensation

Instruction: `WCMP?`

Reply: `RA 0000 WCMP 1328,1544,1798,1996,0.2,0.3,-0.1,0.4`

Name

`WHOL`

Does a wavelength calibration check with holmium filter

`WHOL?`

Showes the last holmium check results

Synopsis

Instruction:	`WHOL`
Reply:	`<rc> WHOL`
Instruction:	`WHOL?`
Reply:	`<rc> WHOL <ok_notok>,<val1>,<val2>,<val3>`

Description

WHOL starts a lambda check with holmium filter and compares the results with the expected holmium wavelengths. The difference of 3 wavelengths is the output and result of this check.
WHOL starts the holmium check. When the check finishe a event is send. The result can be get with WHOL?. WHOL is used for wavelength validation, not for calibration.
WHOL always start a holmium scan simular to SCAN:TAKE 2,190,800,1.

Parameters

`<ok_notok>`	1 for ok, 0 for not ok
`<val1,2,3>`	-12.0 .. +12.0 [nm] as return values!

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6120 holmium check not allowed while lamp off
RE 6121 holmium check not allowed while startup
RE 6122 holmium check not allowed while reset
RE 6123 holmium check not allowed while test
RE 6124 holmium check not allowed while prepare
RE 6125 holmium check not allowed while scan
RE 6126 holmium check not allowed while calibration
RE 6127 holmium check not allowed while holmium check

Examples

(a)	Comment:	start holmium check
	Instruction:	`WHOL`
	Reply:	`RA 0000 WHOL`
(b)	Comment:	get results from holmium check
	Instruction:	`WHOL?`
	Reply:	`RA 0000 WHOL 0,-0.2,0.1`

Name

`WHOL:ABRT`

Abort the wavelength calibration check with holmium filter

Synopsis

Instruction: `WHOL:ABRT`

Reply: `<rc> WHOL:ABRT`

Description

A wavelength calibration check, started by WHOL, can be aborted.

Parameters

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: abort holmium check

Instruction: `WHOL:ABRT`

Reply: `RA 0000 WHOL:ABRT`

Name

SWLC

Set wavelength calibration start value offset

Synopsis

Instruction: `SWLC <caltype>,<start_mot_offset>`

Reply: `<rc> SWLC <caltype>,<start_mot_offset>`

Instruction: `SWLC? <caltype>`

Reply: `<rc> SWLC <caltype>,<start_mot_offset>`

Description

The VWD wavelength calibration is performed by an automatic test using the 0 order and 656nm line. This test is performed when executing the CALS instruction. The result of this wavelength calibration test is the value <actual_mot>. For the next calibration cycle the algorithm starts at this value.

If the algorithm don't terminate the start value for searching the 0 order or 656nm line can be changed with this instruction. The <start_mot_offset> will be add to the start condition <actual_mot>. (-> the start for the search algorithm is <actual_mot>+<start_mot_offset>)

After normal termination the <start_mot_offset> always will be set to 0.

The SWLC setting is not affected when changing the VWD method.

Parameters

<caltype>	0 for 0th order calibration 1 for calibration at 656nm
<start_mot>	-1024 .. 1024 (motor steps)

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set wavelength start offset value to 10

Instruction: `SWLC 0,10`

Reply: `RA 0000 SWLC 0,10`

Name

CALR

Set VWD to 'readyfor calibration'

CALR?

Query time to be ready for calibration

Synopsis

Instruction:	CALR?
Reply:	<rc> CALR <sec_for_rdy>
Instruction:	CALR
Reply:	<rc> CALR

Description

After lamp is switched on the detector waits 10 minutes for lamp stabilize.
During this time calibration is not allowed CALR? showes the time in seconds to stay ready for calibration. During this time the detector stay not in NOTREADY condition because a analysis can be started. Only calibration is not allowed. With CALR the detector was direct switched to 'ready for calibration'.

Parameters

<sec_for_rdy> 0 .. 600

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: check remaining time to start calibration
Instruction: CALR?
Reply: RA 0000 CALR 324

24.3.2. Spectrum Control Commands

Name

SCAN:TAKE

Take a new spectrum

Synopsis

Instruction:	`SCAN:TAKE <stype>,<wlfrom>,<wlto>,<wlinc>`
Reply:	`<rc> SCAN:TAKE <stype>,<wlfrom>,<wlto>,<wlinc>`
Instruction:	`SCAN:TAKE <stype>`
Reply:	`<rc> SCAN:TAKE <stype>`
Instruction:	`SCAN:TAKE?`
Reply:	`<rc> SCAN:TAKE <stype>,<wlfrom>,<wlto>,<wlinc>`
Instruction:	`SCAN:TAKE? <stype>`
Reply:	`<rc> SCAN:TAKE <stype>,<wlfrom>,<wlto>,<wlinc>`
Instruction:	`AT:SCAN <time>,<stype>,<wlfrom>,<wlto>,<wlinc>`
Reply:	`<rc> AT:SCAN <time>,<stype>,<wlfrom>,<wlto>,<wlinc>`
Instruction:	`AT:SCAN? <time>`
Reply:	`<rc> AT:SCAN <time>,<stype>,<wlfrom>,<wlto>,<wlinc>`

Description

With the instruction a scan is started to get a spectrum. After finishing the spectrum the wavelength is reset to the wavelength before starting the scan. Also a event is send out at end of spectrum acquisition. The spectrum is been send out at rawdata cu. Additional the last spectrum of every type is stored into a diag-cu.

If scan is taken with instruction SCAN:TAKE <stype> only, the wavelength setting is given by the internal default variables that can be modified with SCAV <wlfrom>,<wlto>,<wlinc>.

SCAN:TAKE? returns values of last startes scan, SCAN:TAKE? <stype> returns values of last started scan of the given type.

Parameters

<stype>	0 for normal spectrum = sample spectrum (given in AU =log(intensity) including subtraction of blank spectrum 1 for blank spectrum = reference spectrum (given in AU =log(intensity) 2 for holmium spectrum (given as intensity =no log) 3 for lamp characteristic (given as intensity =no log) with sample diode 4 for lamp characteristic (given as intensity =no log) with reference diode
<wlfrom>	190 .. 600 [nm] for stype 0, 1 / 190..800 [nm] for stype 2, 3
<wlto>	190 .. 600 [nm] for stype 0, 1 / 190..800 [nm] for stype 2, 3
<wlinc>	1.. 10 [nm]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6091 scan not allowed while startup
RE 6092 scan not allowed while reset
RE 6094 scan not allowed while prepare
RE 6095 scan not allowed while scan
RE 6096 scan not allowed while calibration test
RE 6097 scan not allowed while holmium check
RE 6101 scan not allowed while lamp ignition
RE 6102 scan not allowed while lamp off
RE 6502 spectra buffer full
RE 6504 take spectrum not allowed
RE 6506 blank spectrum not available
RE 6507 blank spectrum range mismatch

Examples

(a) Comment: take a spectrum
Instruction: `SCAN:TAKE 0,190,600,1`
Reply: `RA 0000 SCAN:TAKE 0,190,600,1`

Name

SCAN:ABRT

Abort current spectrum taken.

Synopsis

Instruction: SCAN:ABRT

Reply: <rc> SCAN:ABRT

Description

A spectrum scan, that is started by SCAN:TAKE ... can be aborted without storage.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: abort current scan

Instruction: SCAN:ABRT

Reply: RA 0000 SCAN:ABRT

Name

SCAN:MAX

Find a maximum in a spectrum

Synopsis

Instruction: `SCAN:MAX? <scantype>,<wlfrom>,<wlto>`

Reply: `<rc> SCAN:MAX <scantype>,<wlfrom>,<wlto>,<wlmax>`

Description

With the instruction a absolute maximum in a given wavelength area in a spectrum aquired before can be calculated. The query returns the wavelength were a maximum is find.
If more than one same maximum find the lowest wavelength was returned.

Parameters

<scantype>	0 for normal spectrum = sample spectrum (given in AU =log(intensity) including subtraction of blank spectrum 1 for blank spectrum = reference spectrum (given in AU =log(intensity) 2 for holmium spectrum (given as intensity =no log) 3 for lamp characteristic (given as intensity =no log) with sample diode 4 for lamp characteristic (given as intensity =no log) with reference diode
<wlfrom>	190 .. 800 [nm] within the area of the spectrum aquired before
<wlto>	190 .. 800 [nm] within the area of the spectrum aquired before
<wlmax>	190 .. 800 [nm] within the area <wlfrom> and <wlto>

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6501 spectrum does not exist
RE 6505 spectrum min/max search inconsistent

Examples

(a) Comment: find a maximum a in sample spectrum
Instruction: `SCAN:MAX 0,250,290`
Reply: `RA 0000 SCAN:MAX 0,250,290,272`

Name

SCAN:MIN
Find a minimum in a spectrum

Synopsis

Instruction: `SCAN:MIN? <scantype>,<wlfrom>,<wlto>`
Reply: `<rc> SCAN:MIN <scantype>,<wlfrom>,<wlto>,<wlmin>`

Description

With the instruction a absolute minimum in a given wavelength area in a spectrum aquired before can be calculated. The query returns the wavelength were a minimum is find.
If more than one same minimum is find the lowest wavelength was returned.

Parameters

<scantype>	0 for normal spectrum = sample spectrum (given in AU =log(intensity) including subtraction of blank spectrum 1 for blank spectrum = reference spectrum (given in AU =log(intensity) 2 for holmium spectrum (given as intensity =no log) 3 for lamp characteristic (given as intensity =no log) with sample diode 4 for lamp characteristic (given as intensity =no log) with reference diode
<wlfrom>	190 .. 800 [nm] within the area of the spectrum aquired before
<wlto>	190 .. 800 [nm] within the area of the spectrum aquired before
<wlmin>	190 .. 800 [nm] within the area <wlfrom> and <wlto>

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6501 spectrum does not exist
RE 6505 spectrum min/max search inconsistent

Examples

(a) Comment: find a minimum a in sample spectrum
Instruction: `SCAN:MIN 0,220,250`
Reply: `RA 0000 SCAN:MIN 0,220,250,232`

24.4. VWD Actuals

Name

ACT:VAL?

Output a internal or chromatographic or other value

Synopsis

Instruction: `ACT:VAL? <sigtype>`

Reply: `<rc> ACT:VAL <sigtype>,<value>`

Description

This function outputs a internal or also a chromatographic value of the VWD for service or diagnostic use. If <sigtype> is not defined a 0 will be returned.

Parameters

<sigtype>	see sigtype description: VWD Rawdata File Description on page 640
<value>	+/- 2^31 (long: depending at <sigtype>)

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: look at current sigtype 4

Instruction: `ACT:VAL? 4`

Reply: `RA 0000 ACT:VAL 4,200`

Name

ACT:SPEC?

Actual variable wavelength detector status

Synopsis

Instruction: `ACT:SPEC?`

Reply: `<rc> ACT:SPEC <general>,<uvlamp>,<init>,<reset>`

Description

The actual query returns the detector state, consisting of a general state, the status of the UV-lamp, the detector initial state and the detector reset state.

Parameters

<general>	reflects general detector state 0 : idle (ES 7090 Device idle) 1 : detector performs a holmium check (ES 7070 WHOL started) 2 : detector performs a wavelength calibration (ES 7060 TEST:WCAL started) 3 : detector performs a scan (ES 7050 Scan) 4 : detector performs a balance (ES 7200 Prepare) 5 : detector is in reset state (ES 7020 Detector Reset) 6 : startup phase of detector
<uvlamp>	reflects UV-lamp state 0 : UV-lamp off (ES 7403 UV lamp off) 1 : UV-lamp on (ES 7404 UV lamp on) 2 : UV-lamp in warmup phase (ES 7406 UV lamp warm up (only VWD)) 3 : UV-lamp in ignition phase (ES 7400 UV lamp ignition) 4 : startup
<init>	reflects init state of detector 0 : detector not in init phase 1 : detector in init phase (during initialisation)
<reset>	reflects reset state of detector 0 : detector not in reset phase 1 : detector in reset phase (during reset of the detector)

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: get actual detector status: idle and lamp on

Instruction: `ACT:SPEC?`

Reply: `RA 0000 ACT:SPEC 0,1,0,0`

24.5. VWD Specific Statemachines

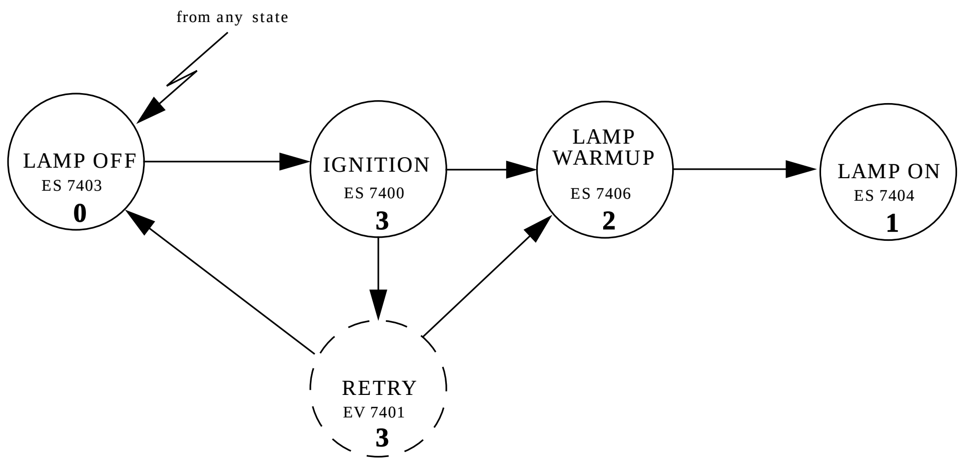
Name

LampSTM
Statemachine of Lamp Behaviour

Description

This statemachines build up the lamp states and possible changes.

State Diagram



Name

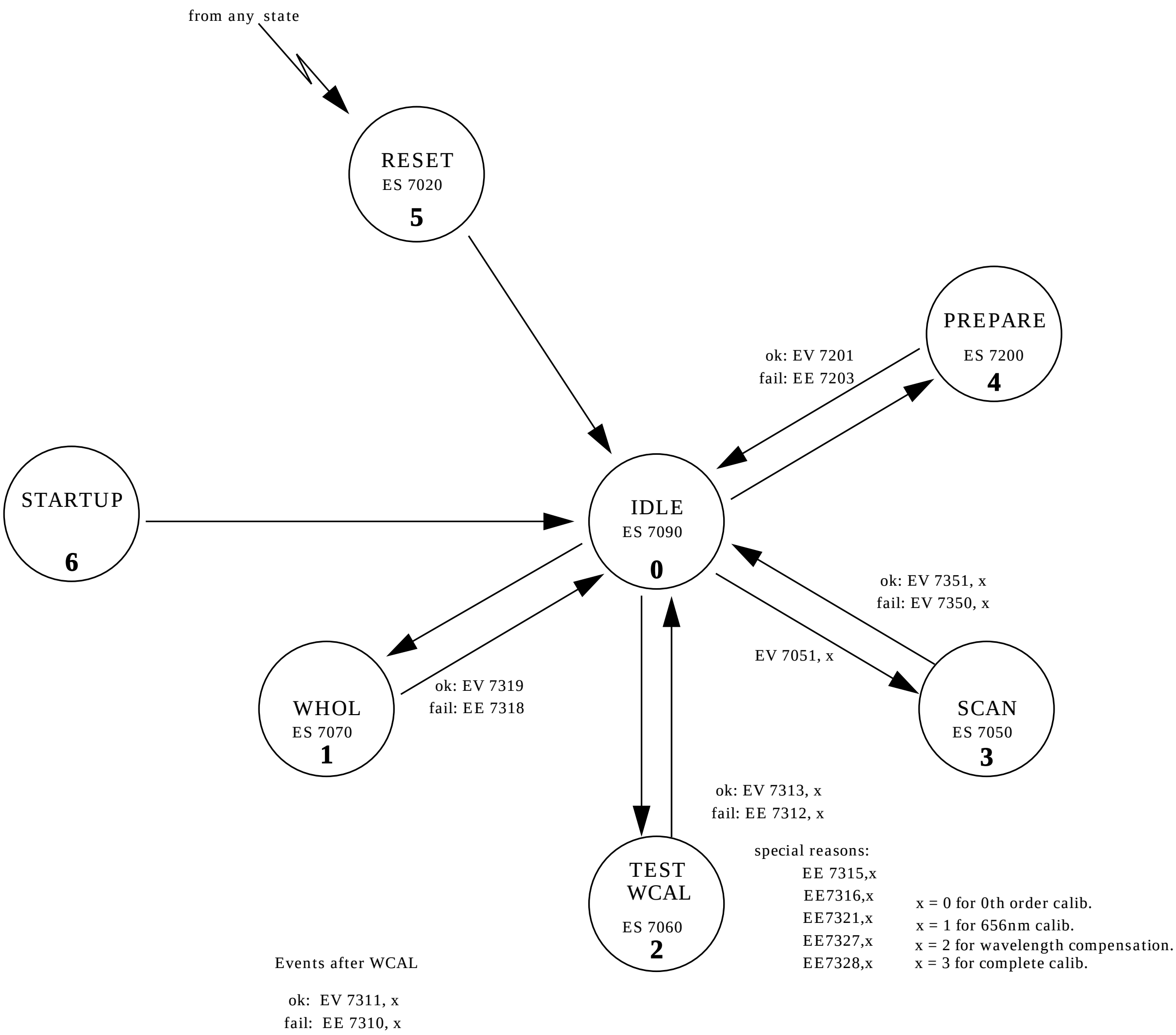
DeviceSTM

Statemachine of general Detector States

Description

This statemachines build up the states and possible changes of the detector state.

State Diagram



24.6. VWD Rawdata File Description

Description

This description is a supplement to: .

The table below describes all signals available in the VWD detector. Signals can be accessed by:
ACT:VAL? <sigtype> command to query a signal
ACH1 <sigtype> to route a signal to the analog output
RAWS <signalset> to get signals by rawdata: RAWS on page 258

Table

signal type	num	RAWS		description	rawdata unit	actual units	analog scale / 1V	file type
raw-data out-put	1	1	A	Absorbance Units AU	2^{-24} AU	AU	1 AU	.ch
	2	2	B	Sample AU without using Reference	2^{-24} AU	AU	1 AU	.ch
	3	4	C	Reference AU without using Sample	2^{-24} AU	AU	1 AU	.ch
	4	8	D	Test Chromatogram output	2^{-24} AU	AU	1 AU	.ch
	5	16	E	Unfiltered and not logarithm sample	1	counts	2^{21} cnt.	.reg
	6	32	F	Unfiltered and not logarithm reference	1	counts	2^{21} cnt.	.reg
	7	64	G	Analog Output Test Function	1	counts		.reg
	8	128	H	Sample diode current	$\frac{0.2098}{2^{gain}}$ pA	nA	0.2098×2^{20} pA (220nA)	.reg
	9	256	I	Reference diode current				.reg
	10	512	J	lamp anode voltage	µV	V	2^{20}µV	.reg
	11	1024	K	lamp anode current	µA	mA	2^{20}µA	.reg
	12	2048	L	lamp heater voltage	µV	V	2^{20}µV	.reg
	13	4096	M	lamp heater current	µA	mA	2^{20}µA	.reg
	14	8192	N	grating motor steps	1	1	2^{15}	.reg
	15	16384	O	filter motor steps	1	1	2^{15}	.reg
	16	32768	P	board temperature	0.1 degC	degC	2^{10}x0.1degC	.reg
	17	65536	Q	PTC current	0.1 mA	mA	2^{12}x0.1mA	.reg
	18	2^17	R	0 (future use)	1	1		.reg
	19	2^18	S	0 (future use)	1	1		.reg
	20	2^19	T	0 (future use)	1	1		.reg
	21	2^20	U	0 (future use)	1	1		.reg
	22	2^21	V	0 (future use)	1	1		.reg
	23	2^22	W	0 (future use)	1	1		.reg
	24	2^23	X	0 (future use)	1	1		.reg
	25	2^24	Y	0 (future use)	1	1		.reg
	26	2^25	Z	0 (future use)	1	1		.reg
no raw-data out-put	100			leak status	enum	enum	8	.reg
	101			grating position sensor (only 0 or 1)	1	1	1	.reg
	102			filter positioning sensor (only 0 or 1)	1	1	1	.reg
	103			Sample Preamplifier Gain	1	1	8	.reg
	104			Reference Preamplifier Gain	1	1	8	.reg
	105			NTC voltage	mV	V	2^{14}mV	.reg
	106			PTC voltage	mV	V	2^{14}mV	.reg
	107			0 (future use)	1			.reg
	108			0 (future use)	1			.reg
	109			0 (future use)	1			.reg
	110			0 (future use)	1			.reg

24.7. VWD Diagnose History Buffers

Name

RCAL (diag)
recalibration of the detector

Characteristics

max entries	30
max characters	60
storage	volatile (flash)
availability	always
storage frequency	on request: one entry

Contents

time, 3 x wl_gap, 2 x steps, 2 x intensity

time,	seconds since 1970, the time when 0-order calibration test is finished
wl_gap[0]	wavelength gap of last zero order calibration
wl_gap[1]	wavelength gap of last 656nm calibration
wl_gap[2]	maximum wavelength gap of last wavelength compensation
mot_steps[0]	motor steps of last zero order calibration
mot_steps[1]	motor steps of last 656nm calibration
intensity[0]	intensity of last zero order calibration
intensity[1]	intensity of last 656nm calibration

Description

Buffer will filled at each execution of WCAL, that now set new calibration values to the hardware.

Name

WCA0 (diag)

result of the wavelength calibration test at 0 order

Characteristics

max entries	50
max characters	35
storage	volatile (nv)
availability	always
storage frequency	on request: one entry

Contents

time, actual_steps,actual_refcounts

time,	seconds since 1970, the time when 0-order calibration test is finished
actual_steps,	the resulting motor steps of the finished calibration test
actual_refcounts	the reference diode counts at actual_steps of the grating motor after the calibration test
actual_wl_gap	the resulting wavelength gap of the calibration test in nm.

Description

A start of a 0-order-calibration test gives the motor steps of the zero order condition and the reference diode counts of the adc. This values are stored into the buffer.
The buffer is only filled if test terminated normal.

Name

WCA6 (diag)

result of the wavelength calibration test at 656 nm

Characteristics

max entries	50
max characters	35
storage	volatile (nv)
availability	always
storage frequency	on request: one entry

Contents

time, actual_steps,actual_refcounts

time,	seconds since 1970, the time when 656nm calibration test is finished
actual_steps,	the resulting motor steps of the finished calibration test
actual_refcounts	the reference diode counts at actual_steps of the grating motor after the calibration test
actual_wl_gap	the resulting wavelength gap of the calibration test in nm.

Description

A start of a 656nm-calibration test gives the motor steps of the zero order condition and the reference diode counts of the adc. This values are stored into the buffer.

The buffer is only filled if test terminated normal.

Name

BALH (diag)

balance/prepare result history

Characteristics

max entries	50
max characters	35
storage	volatile
availability	always
storage frequency	on request: one entry

Contents

time, wavelength,gain,sam_counts,ref_counts

time,	seconds since 1970, the time when the balance occured
wavelength,	the wavelength at which the balance occured
smp_gain,	the resulting sample preamplifier gain
ref_gain,	the resulting reference preamplifier gain
smp_counts,	the adc-counts of the sample diode after balance
ref_counts,	the adc-counts of the reference diode after balance

Description

At end of each balance the values above will be stored to this buffer.

Name

TSTF (diag)
filter motor test results

Characteristics

max entries	30
max characters	25
storage	volatile (NV)
availability	always
storage frequency	on request: one entry

Contents

time, minutes

time,	seconds since 1970, the time when the filter motor test started
motor_steps	resulting motor steps of test, 0 indicates error result

Description

Result of the filter motor test. Start is only on user request.

Name

TSTG (diag)

grating motor test results

Characteristics

max entries	30
max characters	25
storage	volatile (NV)
availability	always
storage frequency	on request: one entry

Contents

time, minutes

time,	seconds since 1970, the time when the grating motor test started
motor_steps	resulting motor steps of test, 0 indicates error result

Description

Result of the grating motor test. Start is only on user request.

24.8. VWD Diagnose Data Buffers

Name

UVIG (diag)
ignition course of the last ignition cycle of the deuterium lamp

Characteristics

max entries	200
max characters	35
storage	non volatile
availability	always
storage frequency	on request: whole buffer with 10Hz

Contents

anode_voltage,anode_current,heater_voltage,heater_current

anode_voltage,	anode voltage in µV
anode_current,	anode current in µA
heater_voltage,	heater voltage in µV
heater_current,	heater current in µA

Description

Ignition course with values above of the last ignition cycle of the deuterium lamp.

Name

UV_1 (diag)

long term values of the deuterium lamp

Characteristics

max entries	120
max characters	35
storage	non volatile
availability	always
storage frequency	always: 1 entry/minute

Contents

anode_voltage,anode_current,heater_voltage,heater_current

anode_voltage,	anode voltage in µV
anode_current,	anode current in µA
heater_voltage,	heater voltage in µV
heater_current,	heater current in µA

Description

Long term values of the deuterium lamp.

Name

`UV_V (diag)`

short term values of the deuterium lamp

Characteristics

max entries	500
max characters	35
storage	non volatile
availability	always
storage frequency	always: 10 Hz

Contents

anode_voltage

anode_voltage,	anode voltage in µV

Description

Short term values (voltage) of the deuterium lamp.

Name

SPE0 (diag)

last spectrum of type 0 = sample spectrum

Characteristics

max entries	611
max characters	35
storage	non volatile
availability	always
storage frequency	on request: whole buffer with 1value/1nm

Contents

abs

abs	absorbance from 190nm to 800nm in steps of 1 nm in AU

Description

Contents the last spectrum acquisition generated by SPEC:TAKE 0,...

Name

SPE1 (diag)
last spectrum of type 1 = blank spectrum

Characteristics

max entries	611
max characters	35
storage	non volatile
availability	always
storage frequency	on request: whole buffer with 1value/1nm

Contents

abs

abs	absorbance from 190nm to 800nm in steps of 1 nm in AU

Description

Contents the last spectrum acquisition generated by SPEC:TAKE 1,...

Name

SPE2 (diag)

last spectrum of type 2 = holmium spectrum

Characteristics

max entries	611
max characters	35
storage	non volatile
availability	always
storage frequency	on request: whole buffer with 1value/1nm

Contents

abs

abs	absorbance from 190nm to 800nm in steps of 1 nm in AU

Description

Contents the last spectrum acquisition generated by SPEC:TAKE 2,...

Name

SPE3 (diag)
last spectrum of type 3 = lamp characteristic with sample diode

Characteristics

max entries	611
max characters	35
storage	non volatile
availability	always
storage frequency	on request: whole buffer with 1value/1nm

Contents

intensity

intensity	intensity from 190nm to 800nm in steps of 1 nm in ADC-counts

Description

Contents the last spectrum acquisition generated by SPEC:TAKE 3,...

Name

SPE4 (diag)

last spectrum of type 4 = lamp characteristic with reference diode

Characteristics

max entries	611
max characters	35
storage	non volatile
availability	always
storage frequency	on request: whole buffer with 1value/1nm

Contents

intensity

intensity	intensity from 190nm to 800nm in steps of 1 nm in ADC-counts

Description

Contents the last spectrum acquisition generated by SPEC:TAKE 4,...

24.9. VWD Diagnose Counters

24.9.1. Lamp

Name

`D_SW (diag)`

Deuterium lamp ignition counter (switch on cycles)

Increments by each ignition of the deuterium lamp. The number is equal to the counts of the 'switch on' cycles of the deuterium lamp. If a retry is used two ignition-cycles are done: the counter increments by 2.

Characteristics

counting frequency	ignition cycles
storage	volatile (flash)
history entry	20
EMF-event	EF 7474 UV-lamp ignition cycles exceeded (page 901)

Name

`D_TI (diag)`

Deuterium lamp on time of this lamp

Gives the time in seconds during this D2-lamp is on.

When a new lamp is installed the counter D_TI which stores the 'lamp-on-time' of the current lamp must be reset. Before reseting the value will be stored together with the current date code into the included buffer. The buffer now shows the dates of lamp changes and the burning-time in minutes of this lamps.

Characteristics

counting frequency	seconds (of complete lamp on time)
storage	volatile (flash)
history entry	20
EMF-event	EF 7470 UV- lamp life-time exceeded (page 901)

Name

`D_ON (diag)`

Deuterium lamp on time this on cycle

Gives the time in seconds during the lamp is on started from last 'switch on' of the lamp

Characteristics

counting frequency	seconds (of lamp on time)
storage	non volatile (ram)
history entry	20
EMF-event	EF 7472 UV- lamp on-time exceeded (page 901)

24.9.2. Others

Name

DECS (diag)

Diode exposure counter for sample diode

Gives a value for the integrated light on this sample diode to calculate lifetime/exposure of the sample diode. Increment is depending at wavelength, because UV-light is more destroying the diode.

Characteristics

counting frequency	
storage	volatile (flash)
history entry	4
EMF-event	EF 7550 DECS Sample Diode Exposure Counter limit exceeded (only VWD) (page 901)

Name

DECR (diag)

Diode exposure counter for reference diode

Gives a value for the integrated light on this reference diode to calculate lifetime/exposure of the refernece diode. Increment is depending at wavelength, because UV-light is more destroying the diode.

Characteristics

counting frequency	
storage	volatile (flash)
history entry	4
EMF-event	EF 7551 DECR Reference Diode Exposure Counter limit exceeded (only VWD) (page 901)

24.10. VWD Service Functions

24.10.1. Stepper Motor Functions

Name

SVC:FILA
Set the filter motor to a absolute desired position

Synopsis

Instruction:	`SVC:FILA <pos_id>`
Reply:	`<rc> SVC:FILA <pos_id>`
Instruction:	`SVC:FILA?`
Reply:	`<rc> SVC:FILA <pos_id>`

Description

Set or query the filter motor to a absolute desired position.

Parameters

<pos_id	0	sensor position
	1	off
	2	on (second order filter)
	3	holmium filter
	4	unused position

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: set holmium filter
Instruction: `SVC:FILA 2`
Reply: `RA 0000 SVC:FILA 2`

Name

SVC:FILR

Set the filter motor to a relative desired position

Synopsis

Instruction: SVC:FILR <pos_rel>

Reply: <rc> SVC:FILR <pos_rel>

Description

Set the filter motor to a relative position counted from the actual position.

SCR:FILR 0 has no effect on the instrument.

Parameters

<pos_rel>	+/- 4

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	set next forward filter motor position
	Instruction:	SVC:FILR 1
	Reply:	RA 0000 SVC:FILR 1
(b)	Comment:	set backward filter motor position
	Instruction:	SVC:FILR -1
	Reply:	RA 0000 SVC:FILR -1

Name

SVC:GRAA
Set the grating motor to a absolute desired position

Synopsis

Instruction:	`SVC:GRAA <steps>`
Reply:	`<rc> SVC:GRAA <steps>`
Instruction:	`SVC:GRAA?`
Reply:	`<rc> SVC:GRAA <steps>`

Description

Set or query the grating motor to a absolute desired position.

Parameters

`<steps>` 0 .. 30000

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set grating motor to 14800 step-position
Instruction: `SVC:GRAA 14800`
Reply: `RA 0000 SVC:GRAA 14800`

Name

SVC:GRAR
Set the grating motor to a relative desired position

Synopsis

Instruction: SVC:GRAR <steps>
Reply: <rc> SVC:GRAR <steps>

Description

Set the grating motor to a relative desired position.
SCR:GRAR 0 has no effect on the instrument.

Parameters

<steps> +/- 30000

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: move grating motor 10 steps forward
Instruction: SVC:GRAA 10
Reply: RA 0000 SVC:GRAA 10

(b) Comment: move grating motor 10 steps backward
Instruction: SVC:GRAA -10
Reply: RA 0000 SVC:GRAA -10

Name

SVC:SHOM
Home position and initialize for stepper motors

Synopsis

Instruction: SVC:SHOM <home_id>
Reply: <rc> SVC:SHOM <home_id>

Description

Does a initialization of the grating motor or the filter motor and search home position or moves to desired zero-order position.

Parameters

<home_id>
0 grating motor move to zero-order position
1 grating motor move to home position
2 filter motor move to home position

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: init the grating stepper motor and search home position
Instruction: SVC:SHOM 1
Reply: RA 0000 SVC:SHOM 1

24.10.2. Frontend Controlling

Name

SVC:PRGS
Set direct the absolute preamplifier gain of the sample adc.

Synopsis

Instruction:	`SVC:PRGS <sample_gain>`
Reply:	`<rc> SVC:PRGS <sample_gain>`
Instruction:	`SVC:PRGS?`
Reply:	`<rc> SVC:PRGS <sample_gain>`

Description

The gain of the preamplifier stage of the sample adc can be direct set or query.

Parameters

<sample_gain>	0 .. 5 normal gain settings	(set & query)
	-1 special gain during calibration	(query only not settable!)
	-2 special gain during scan	(query only not settable!)

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual preamplifier gain of the sample adc
Instruction: `SVC:PRGS?`
Reply: `RA 0000 SVC:PRGS 3`

Name

SVC:PRGR

Set direct the absolute preamplifier gain of the reference adc.

Synopsis

Instruction:	SVC:PRGR <ref_gain>
Reply:	<rc> SVC:PRGR <ref_gain>
Instruction:	SVC:PRGR?
Reply:	<rc> SVC:PRGR <ref_gain>

Description

The gain of the preamplifier stage of the reference adc can be direct set or query.

Parameters

<ref_gain>	0 .. 5 normal gain settings	(set & query)
	-1 special gain during calibration	(query only not settable!)
	-2 special gain during scan	(query only not settable!)

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual preamplifier gain of the reference adc

Instruction: SVC:PRGR?

Reply: RA 0000 SVC:PRGR 16

Reply:

24.11. VWD Test Functions

Name

TEST:STST
To start and stop a anonymous test

Synopsis

Instruction:	`TEST:STST <on/off>`
Reply:	`<rc> TEST:STST <on/off>`
Instruction:	`TEST:STST?`
Reply:	`<rc> TEST:STST <on/off>`

Description

This function turns the detector in a anonymous test.
Also with TEST:STST? it can be query whether the detector is in test mode or not.

Parameters

<on/off>	0 for off, 1 for on

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 test while startup
RE 6062 test while reset
RE 6063 test while test
RE 6064 test while prep
RE 6065 test while scan
RE 6066 test while calibrate
RE 6067 test while holmium check
RE 6071 test while leak
RE 6072 test while shutdown
RE 6074 test while run
RE 6075 not allowed during test
RE 6076 test not allowed now
RE 6077 test while abort
RE 6081 this test is not running

Examples

(a) Comment: set the detector in test mode
Instruction: `TEST:STST 1`
Reply: `RA 0000 TEST:STST 1`

Name

TEST:WCAL

Start and query complete wavelength calibration cycle

Synopsis

Instruction:	`TEST:WCAL`
Reply:	`<rc> TEST:WCAL`
Instruction:	`TEST:WCAL?`
Reply:	`<rc> TEST:WCAL <wl_gap_0>,<wl_gap_656>,<wl_gap_comp>`
Instruction:	`TEST:WCAL <wl_gap_0>,<wl_gap_656>,<wl_gap_comp>`
Reply:	`<rc> TEST:WCAL <wl_gap_0>,<wl_gap_656>,<wl_gap_comp>`
Instruction:	`TEST:WCAL? <cal_type>`
Reply:	`<rc> TEST:WCAL <cal_type>,<wl_gap>`
Instruction:	`ACT:WCAL?`
Reply:	`<rc> ACT:WCAL <cal_state>`
Instruction:	`TEST:WCAL:MOT?`
Reply:	`<rc> TEST:WCAL:MOT <mot_0>,<mot_656>,<mot_361>, <mot_419>,<mot_486>,<mot_537>`
Instruction:	`TEST:WCAL:MOT <mot_0>,<mot_656>,<mot_361>, <mot_419>,<mot_486>,<mot_537>`
Reply:	`<rc> TEST:WCAL:MOT <mot_0>,<mot_656>,<mot_361>, <mot_419>,<mot_486>,<mot_537>`
Instruction:	`TEST:WCAL:WGAP?`
Reply:	`<rc> TEST:WCAL:WGAP <wl_gap_0>,<wl_gap_656>, <wl_gap_361>,<wl_gap_419>, <wl_gap_486>,<wl_gap_537>`
Instruction:	`TEST:WCAL:WGAP <wl_gap_0>,<wl_gap_656>, <wl_gap_361>,<wl_gap_419>, <wl_gap_486>,<wl_gap_537>`
Reply:	`<rc> TEST:WCAL:WGAP <wl_gap_0>,<wl_gap_656>, <wl_gap_361>,<wl_gap_419>, <wl_gap_486>,<wl_gap_537>`
Instruction:	`TEST:WCAL:INTE?`
Reply:	`<rc> TEST:WCAL:INTE <intensity_0>,<intensity_656>`
Instruction:	`TEST:WCAL:INTE <intensity_0>,<intensity_656>`
Reply:	`<rc> TEST:WCAL:INTE <intensity_0>,<intensity_656>`
Instruction:	`TEST:WCAL:ALIG?`
Reply:	`<rc> TEST:WCAL:ALIG <wl_alignment>`
Instruction:	`TEST:WCAL:ALIG <wl_alignment>`
Reply:	`<rc> TEST:WCAL:ALIG <wl_alignment>`
Instruction:	`TEST:WCAL:ABRT`
Reply:	`<rc> TEST:WCAL:ABRT`

Description

TEST:WCAL starts a complete wavelength calibration cycle.
First 0 order calibraion is started. After finishing ok,the event 7313,0 is send out.
The result can be query by TEST:WCAL? 0
Then 656nm calibration started. After finishing ok, the event 7313,1 is send out.
The result can be query by TEST:WCAL? 1

Then wavelength compensation algorithm started. After finishing ok, the event 7313,2 is send out.
The result can be query by TEST:WCAL? 2
(this value wl_gap_comp is the maximum deviation of the wavelength compensation algorithm.
max amount of wl_gap_361, wl_gap_419, wl_gap_486, wl_gap_537 that can be query in detail with the TEST:WCAL:WGAP? instruction)
After complete finished event 7313,3 is send out.
The detector then get ready and idle.
The whole result then also can be query with TEST:WCAL?

For maintanance reason it is possible to enter the calibration values manually using the commands:
TEST:WCAL <wl_gap_0>,<wl_gap_656>,<wl_gap_comp>
TEST:WCAL:MOT <mot_0>,<mot_656>,<mot_361>,<mot_419>,<mot_486>,<mot_537>
TEST:WCAL:WGAP <wl_gap_0>,<wl_gap_656>,<wl_gap_361>,<wl_gap_419>,<wl_gap_486>, <wl_gap_537>
TEST:WCAL:INTE <intensity_0>,<intensity_656>
TEST:WCAL:ALIG <wl_alignment
To adjust the calculated values to the optical unit WCAL must be send.
(NOTE: all the 5 commands (TEST:WCAL <params>; TEST:WCAL:MOT <params>; TEST:WCAL:WGAP <params>; TEST:WCAL:INTE <params> and TEST:WCAL:ALIG <param>) MUST executed before the wavelength calibration values are written to HW resp. Flash-ram using the command WCAL!)

Abort of the the current wavelength calibration test can be done by TEST:WCAL:ABRT.

Parameters

<cal_state>	-1 for inactive 0 for 0 order calibration working 1 for 656nm calibration working 2 for wavelength compensation working
<cal_type>	0 for 0 order calibration 1 for 656nm calibration 2 for wavelength compensation
<wl_gap_...>	-100.0 .. 100.0 wl gap of the last calibration in nm
<mot_...>	0 .. 32000 value shows the resulting motor-steps after calibration
<intensity_...>	0 .. 2^31 intensity = adc-counts of reference diode of the last calibration
<wl_alignment>	-100.0 .. 100.0 wl alignment gap between sample an reference side in nm

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6110 calibration test not allowed while lamp off
RE 6111 calibration test not allowed while startup
RE 6112 calibration test not allowed while reset
RE 6113 calibration test not allowed while test
RE 6114 calibration test not allowed while prepare
RE 6115 calibration test not allowed while scan
RE 6116 calibration test not allowed while calibration
RE 6117 calibration test not allowed while holmium check
RE 6118 calibration test not allowed while lamp stabilize

Examples

(a) Comment: start calibration
Instruction: `TEST:WCAL`
Reply: `RA 0000 TEST:WCAL`

(b) Comment: query result after calibration has finished
Instruction: `TEST:WCAL?`
Reply: `RA 0000 TEST:WCAL 0.8,0.5,0.5`

Name

TEST:CALS
Test and query current wavelength calibration

Synopsis

Instruction:	`TEST:CALS <caltype>`
Reply:	`<rc> TEST:CALS <caltype>`
Instruction:	`TEST:CALS?`
Reply:	`<rc> TEST:CALS <caltype>,<c_mot>,<c_int>,<wl_gap>`
Instruction:	`TEST:CALS? <caltype>`
Reply:	`<rc> TEST:CALS <caltype>,<c_mot>,<c_int>,<wl_gap>`

Description

TEST:CALS starts a wavelength calibration test of the detector. At end of calibration test a event is send out. The result can be query by TEST:CALS?.
As return the current motor steps <c_mot>, the current intensity in counts <c_int>
and the wavelength deviation <wl_gap> is given.
The deviation of the current wavelength setting are given by the <wl_gap> in nm.
See also CALS.

Parameters

<caltype>	0 for starting 0th order calibration test 1 for starting calibration test at 656nm
<c_mot>	0 .. 32000 value showes the resulting motor-steps of the calibration test
<c_int>	0 .. 2^31 intensity = adc-counts of reference diode of the calibration test
<wl_gap>	-12.0 ... +12.0 nm

Default Values

none

Reply Codes <rc>

RA 0000	no error, instruction has been accepted
RE 6110	calibration test not allowed while lamp off
RE 6111	calibration test not allowed while startup
RE 6112	calibration test not allowed while reset
RE 6113	calibration test not allowed while test
RE 6114	calibration test not allowed while prepare
RE 6115	calibration test not allowed while scan
RE 6116	calibration test not allowed while calibration
RE 6117	calibration test not allowed while holmium check
RE 6118	calibration test not allowed while lamp stabilize

Examples

(a) Comment: start a 656 nm calibration test
Instruction: `TEST:CALS? 1`
Reply: `RA 0000 TEST:CALS 1`

Name

`TEST:WCMP`

Test and query wavelength compensation

Synopsis

Instruction:	`TEST:WCMP`
Reply:	`<rc> TEST:WCMP`
Instruction:	`TEST:WCMP?`
Reply:	`<rc> TEST:WCMP <mot_361>,<mot_419>, <mot_486>,<mot_537>, <wlgap_361>,<wlgap_419>, <wlgap_486>,<wlgap_537>`

Description

TEST:WCMP starts a wavelength compensation test of the detector. At end of calibration test a event is send out. The result can be query by TEST:WCMP?.
See also WCMP.

Parameters

<mot_...>	0 .. 32000	motor steps of given wavelengths
<wlgap_...>	0 .. 100.0	wavelength gap before compensation in nm

Default Values

none

Reply Codes <rc>

RA 0000	no error, instruction has been accepted
RE 6110	calibration test not allowed while lamp off
RE 6111	calibration test not allowed while startup
RE 6112	calibration test not allowed while reset
RE 6113	calibration test not allowed while test
RE 6114	calibration test not allowed while prepare
RE 6115	calibration test not allowed while scan
RE 6116	calibration test not allowed while calibration
RE 6117	calibration test not allowed while holmium check
RE 6118	calibration test not allowed while lamp stabilize

Examples

(a) Comment: start a compensation test
Instruction: `TEST:WCMP`
Reply: `RA 0000 TEST:WCMP`

Name

TEST:FILT

Test the filter motor.

Synopsis

Instruction:	`TEST:FILT`
Reply:	`<rc> TEST:FILT`
Instruction:	`TEST:FILT?`
Reply:	`<rc> TEST:FILT <ok/notok>,<motor_pos>`

Description

This function starts a automatic test for the filter motor. The test terminated automaticly.
At end of the test a event will be send out.
The result of the last test can be query always by the TEST:FILT? function.

Parameters

<ok/notok>	0 for last test terminate normal, otherwise errornumber = 1
<motor_pos>	resulting motor position of normal termination of motor test

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 test while startup
RE 6062 test while reset
RE 6063 test while test
RE 6064 test while prep
RE 6065 test while scan
RE 6066 test while calibrate
RE 6067 test while holmium check
RE 6071 test while leak
RE 6072 test while shutdown
RE 6074 test while run
RE 6075 not allowed during test
RE 6076 test not allowed now
RE 6077 test while abort
RE 6081 this test is not running

Examples

(a) Comment: start the filter motor test
Instruction: `TEST:FILT`
Reply: `RA 0000 TEST:FILT`

Name

TEST:GRAT
Test the grating motor.

Synopsis

Instruction:	TEST:GRAT
Reply:	<rc> TEST:GRAT
Instruction:	TEST:GRAT?
Reply:	<rc> TEST:GRAT <ok/notok>,<motor_pos>

Description

This function starts a automatic test for the grating motor. The test terminated automaticly.
At end of the test a event will be send out.
The result of the last test can be query always by the TEST:GRAT? function.

Parameters

<ok/notok>	0 for last test terminate normal, otherwise errornumber = 1
<motor_pos>	resulting motor position of normal termination of motor test

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 test while startup
RE 6062 test while reset
RE 6063 test while test
RE 6064 test while prep
RE 6065 test while scan
RE 6066 test while calibrate
RE 6067 test while holmium check
RE 6071 test while leak
RE 6072 test while shutdown
RE 6074 test while run
RE 6075 not allowed during test
RE 6076 test not allowed now
RE 6077 test while abort
RE 6081 this test is not running

Examples

(a) Comment: start the grating motor test
Instruction: TEST:GRAT
Reply: RA 0000 TEST:GRAT

Name

TEST:DARK

Turns on or switch off the dark current test mode.

Synopsis

Instruction: `TEST:DARK <on/off>`
Reply: `<rc> TEST:DARK <on/off>`
Instruction: `TEST:DARK?`
Reply: `<rc> TEST:DARK <on/off>`

Description

This function turns the detector in the dark current test mode or switch off this mode and returns to normal operation. For the dark current test the lamp will be switched off. After termination the lamp will be witched into the previous state.
This test mode must be terminated by the user.
With the sample and reference diode current output a verification of the dark current can be done.
During test mode no analysis can be start. The detector is notready.

Parameters

<on/off> 0 for off, 1 for on

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 test while startup
RE 6062 test while reset
RE 6063 test while test
RE 6064 test while prep
RE 6065 test while scan
RE 6066 test while calibrate
RE 6067 test while holmium check
RE 6071 test while leak
RE 6072 test while shutdown
RE 6074 test while run
RE 6075 not allowed during test
RE 6076 test not allowed now
RE 6077 test while abort
RE 6081 this test is not running

Examples

(a) Comment: set the detector into the darkcurrent test mode
Instruction: `TEST:DARK 1`
Reply: `RA 0000 TEST:DARK 1`

Name

TEST:ADC

Turns on or switch off the sample and reference ADC test mode.

Synopsis

Instruction: `TEST:ADC <on/off>`
Reply: `<rc> TEST:ADC <on/off>`
Instruction: `TEST:ADC?`
Reply: `<rc> TEST:ADC <on/off>`

Description

This function turns the detector in the ADC test mode or switch off this mode and returns to normal operation.
This test mode must be terminated by the user.
The ADCs of sample and reference diode will be stay in internal test mode.
With the sample diode current output a verification of the frontend noise can be done.

Parameters

<on/off> 0 for off, 1 for on

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 test while startup
RE 6062 test while reset
RE 6063 test while test
RE 6064 test while prep
RE 6065 test while scan
RE 6066 test while calibrate
RE 6067 test while holmium check
RE 6071 test while leak
RE 6072 test while shutdown
RE 6074 test while run
RE 6075 not allowed during test
RE 6076 test not allowed now
RE 6077 test while abort
RE 6081 this test is not running

Examples

(a) Comment: set the detector into the ADC test mode
Instruction: `TEST:ADC 1`
Reply: `RA 0000 TEST:ADC 1`

Name

TEST:DAC

Turns on or switch off the DAC test mode.

Synopsis

Instruction:	`TEST:DAC <on/off>`
Reply:	`<rc> TEST:DAC <on/off>`
Instruction:	`TEST:DAC?`
Reply:	`<rc> TEST:DAC <on/off>`

Description

This function turns the detector in the DAC test mode (testing the analog output) or switch off this mode and returns to normal operation.

This test mode must be terminated by the user.

A fix pattern is send to the analog output. This pattern and a specified noise must be measured at the analog output.

The signal is a rectangle:

Height is 10µV +/- 2µV, Noise must be <= 3µVpp, Absolute value is depending at Zero Offset.

This must be controlled and passed by a external device like a integrator.

During test mode no analysis can be start. The detector is notready.

Parameters

<on/off>	0 for off, 1 for on

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 6061 test while startup
RE 6062 test while reset
RE 6063 test while test
RE 6064 test while prep
RE 6065 test while scan
RE 6066 test while calibrate
RE 6067 test while holmium check
RE 6071 test while leak
RE 6072 test while shutdown
RE 6074 test while run
RE 6075 not allowed during test
RE 6076 test not allowed now
RE 6077 test while abort
RE 6081 this test is not running

Examples

(a) Comment: set the detector into the DAC test mode
Instruction: `TEST:DAC 1`
Reply: `RA 0000 TEST:DAC 1`

24.12. Compare of HP 1050 and Agilent 1200 Functions

1050		1100 / 1200
1	SAMPLE SCAN	SCAN:TAKE 0,190,800,1
2	REF. SCAN	SCAN:TAKE 3,190,800,1
3	SPECTRUM OUT	SCAN:PLOT
4	SCAN FROM	parameter included at SCAN:TAKE: <from_wl>,<to_wl>
5	SCAN TO	
6	ZERO OFFSET	paramter <offset%> of AZE1
7	RESPONSETIME	PKWD
8	AUTO LAMPOFF	TOUT (core function) and/or controller programmable
9	AUTO LAMPON	ATON (core function) and/or controller programmable
10	OUTP. DEVICE	see AAT1 and ASC1, (marks not supported)
11	OUTPUT VOLT	see ASC1
12	STATUS	core function: STAT? and IDN?
13	START MODE	core function: and cosy list controlled
14	OUTPUT CHECK	not longer implemented
15	RESET to default	core function: RSET
16	PHOTOCURRENT	for sample diode: CHAn 102,0 and ACT:VAL? 102 for reference diode: CHAn 103,0 and ACT:VAL? 103
17	PARAM. LOCK	core function:
18	λ SHIFT	not longer supported (because of GLP problems)
19	POLARITY	POL
20 21 22	(see service functions)	

Compare of HP 1050 and Agillnt 1200 Functions (continue)

1050		Leonardo
20	0th CALIB	WCAL 0
21	λ CALIBRATION	WCAL 1
22	HOLMIUMCHECK	WHOL
31	SET λ PARAM	SWLC
32	FIX SIGNAL	CHAn 2,0 or CHAn 3,0 (see CHAn <sigtype>
33	LEAK S. VOLT	CHAn 212,0 and ACT:VAL? 212
34	VOLTAGE TEST	CHAn 204,0 thru CHAn 207,0 or ACT:VAL? 204 .. 207
35	ADC NOISE	for sample TEST:ADCS and monitor CHAn 102,0 for reference TEST:ADCR and monitor CHAn 103,0
36	GRATING P.S.	monitor CHAn 208,0 or ACT:VAL? 208 use TEST:GRAT
37	FILTER P.S.	monitor CHAn 210,0 or ACT:VAL? 210 use TEST:FILT
38	REMOTE TEST	core function: REMO:STAT?
39	FILTER CHECK	TEST:FILT
40	0th TEST OFF	use: TEST:GRAT and SVC:GRAA and SVC:SHOM
41	DAC TEST	TEST:DAC
42	PREAMP GAIN	SVC:PRGS and SVC:PRGR
43	EEPROM TEST	core function: self test ...
44	DAC CALIB	not longer implemented
45	λ COMPENSATE	included at TEST:WCAL

25. Fluorescence Detector (G1321A)

25.1. FLD Method Parameter

Name

AMOD
Acquisition mode

Synopsis

Instruction:	`AMOD <acq_mode>`
Reply:	`<rc> AMOD <acq_mode>`
Instruction:	`AMOD?`
Reply:	`<rc> AMOD <acq_mode>`

Description

Switch the detector in different acquisition modes.

Parameters

<acq_mode>	0 for fluorescence acquisition
	1 for phosphorescence acquisition
	2 for chemoluminescence acquisition

Default Values

0 for fluorescence acquisition mode

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: switch to ’Phosphorescence Acquisition Mode’
Instruction: `AMOD 1`
Reply: `RA 0000 AMOD 1`

Name

MMOD
Multi-Wavelength Mode

Synopsis

Instruction:	`MMOD <mw_mode>`
Reply:	`<rc> MMOD <mw_mode>`
Instruction:	`MMOD?`
Reply:	`<rc> MMOD <mw_mode>`
Instruction:	`ACT:MMOD?`
Reply:	`<rc> ACT:MMOD <mw_mode>`

Description

Defines the Multi-Wavelength-Mode of the Detector. Switching between different multi-wavelength modes lasts from 20secs up to 50secs, the module becomes NOT_READY (see also ACT:NRDY?) inbetween!

Parameters

`<mw_mode>`	0 for OFF (no multi-wavelength mode) 1 for multi excitation wavelength and excitation spectrum 2 for multi emission wavelength and emision spectrum

Default Values

0 for OFF (no multi-wavelength mode)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set mode to multi excitation wavelength
Instruction: `MMOD 1`
Reply: `RA 0000 MMOD 1`

Name

EXn
Excitation base wavelength

Synopsis

Instruction:	`EXn <on/off>,<exwl>`
Reply:	`<rc> EXn <on/off>,<exwl>`
Instruction:	`EXn?`
Reply:	`<rc> EXn <on/off>,<exwl>`
Instruction:	`ACT:EXn?`
Reply:	`<rc> ACT:EXn <on/off>,<exwl>`
Instruction:	`AT:EXn <time>,<on/off>,<exwl>`
Reply:	`<rc> AT:EXn <time>,<on/off>,<exwl>`
Instruction:	`AT:EXn? <time>`
Reply:	`<rc> AT:EXn <time>,<on/off>,<exwl>`

Description

Set excitation base wavelength for channel 1, 2, 3 and 4. Setting wavelength for channel 2, 3 and 4 will only be valid if multi-wavelength mode is set to 1 (MMOD = 1).

Parameters

`<on/off>`	0 = OFF, 1 = ON
`<exwl>`	0 for zero order or 200 .. 1200nm
`<time>`	0 .. 99999.00 runtime in minutes, counted upward from 0 at start of run.

Default Values

EX1:	1 (= ON), 230nm
EX2, EX3, EX4:	0 (= OFF), 230nm

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set excitation wavelength of channel 1 to 360 nm.
Instruction: `EX1 1, 360`
Reply: `RA 0000 EX1 1, 360`

(b) Comment: get excitation wavelength (home value) of channel 3.
Instruction: `EX3?`
Reply: `RA 0000 EX3 0, 230`

Name

EMn

Emission base wavelength

Synopsis

Instruction:	`EMn <on/off>,<emwl>`
Reply:	`<rc> EMn <on/off>,<emwl>`
Instruction:	`EMn?`
Reply:	`<rc> EMn <on/off>,<emwl>`
Instruction:	`ACT:EMn?`
Reply:	`<rc> ACT:EMn <on/off>,<emwl>`
Instruction:	`AT:EMn <time>,<on/off>,<emwl>`
Reply:	`<rc> AT:EMn <time>,<on/off>,<emwl>`
Instruction:	`AT:EMn? <time>`
Reply:	`<rc> AT:EMn <time>,<on/off>,<emwl>`

Description

Set Emission base wavelength of channel 1, 2, 3 and 4. Setting wavelength for channel 2, 3 and 4 will only be valid if multi-wavelength mode is set to 2 (MMOD = 2).

Parameters

`<on/off>`	0 = OFF, 1= ON
`<emwl>`	0 for zero order or 200 .. 1200nm
`<time>`	0 .. 99999.00 runtime in minutes, counted upward from 0 at start of run.

Default Values

EM1:	1 (= ON), 460nm
EM2, EM3, EM4:	0 (= OFF), 460nm

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set emission wavelength of channel 1 to 500 nm.
Instruction: `EM1 1, 500`
Reply: `RA 0000 EM1 1, 500`

(b) Comment: get emission wavelength (home value) of channel 3.
Instruction: `EM3?`
Reply: `RA 0000 EM3 0, 600`

Name

EXST

Set Acquisition Type for Excitation Spectra

Synopsis

Instruction:	`EXST <type>`
Reply:	`<rc> EXST <type>`
Instruction:	`EXST?`
Reply:	`<rc> EXST <type>`
Instruction:	`ACT:EXST?`
Reply:	`<rc> ACT:EXST <type>`
Instruction:	`AT:EXST <time>,<type>`
Reply:	`<rc> AT:EXST <time>,<type>`
Instruction:	`AT:EXST? <time>`
Reply:	`<rc> AT:EXST <time>,<type>`

Description

Controls the acquisition of excitation spectra. Taking excitation spectra will only be possible if multi wavelength mode is set to 1 (MMOD = 1).

Parameters

`<type>`	0 = OFF (no excitation spectra are stored) 1 apex 2 all in peak (from begin to end of peak) 3 all spectra 4 all spectra only, but no chromtographical signal is generated! (EXn/EMn are ignored ->can be used for screening applications)
`<time>`	0 .. 99999.00 runtime in minutes, counted upward from 0 at start of run.

Default Values

0 (= OFF)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: acquire all excitation spectra

Instruction: `EXST 3`

Reply: `RA 0000 EXST 3`

Name

EXSP
Variables for Excitation Spectra

Synopsis

Instruction:	`EXSP <wlfrom>,<wlto>,<wlinc>`
Reply:	`<rc> EXSP <wlfrom>,<wlto>,<wlinc>`
Instruction:	`EXSP?`
Reply:	`<rc> EXSP <wlfrom>,<wlto>,<wlinc>`
Instruction:	`ACT:EXSP?`
Reply:	`<rc> ACT:EXSP <wlfrom>,<wlto>,<wlinc>`
Instruction:	`AT:EXSP <time><wlfrom>,<wlto>,<wlinc>`
Reply:	`<rc> AT:EXSP <time><wlfrom>,<wlto>,<wlinc>`
Instruction:	`AT:EXSP? <time>`
Reply:	`<rc> AT:EXSP <time><wlfrom>,<wlto>,<wlinc>`

Description

Set parameters for acquisition of excitation spectra. Taking excitation spectra will only be possible if multi wavelength mode is set to 1 (MMOD = 1).
Time for taking a single spectrum could calculated with following formula:

$$t_{spectrum} = \left(\frac{wlto - wlfrom}{wlinc} + 1\right) \times (NumChan + 1) \times 0.0135 \text{ sec}$$

'NumChan' is the number of active chromatogram channels (see EXn).
Acquiring of exitation spectra will only take place if EXST is not equal 0.
(e.g. EXST = 3: acquire all excitation spectra)

Parameters

`<wlfrom>`	200 .. 1200nm
`<wlto>`	200 .. 1200nm
`<wlinc>`	1 .. 20nm
`<time>`	0 .. 99999.00 runtime in minutes, counted upward from 0 at start of run.

Default Values

220nm, 300nm, 10nm

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set wavelength range for excitation spectrum.
Instruction: `EXSP 220, 360, 10`
Reply: `RA 0000 EXSP 220, 360, 10`

Name

EMST
Set Acquisition Type for Emission Spectra

Synopsis

Instruction:	`EMST <type>`
Reply:	`<rc> EMST <type>`
Instruction:	`EMST?`
Reply:	`<rc> EMST <type>`
Instruction:	`ACT:EMST?`
Reply:	`<rc> ACT:EMST <type>`
Instruction:	`AT:EMST <time>,<type>`
Reply:	`<rc> AT:EMST <time>,<type>`
Instruction:	`AT:EMST? <time>`
Reply:	`<rc> AT:EMST <time>,<type>`

Description

Controls the acquisition of emission spectra. Taking emission spectra will only be possible if multi wavelength mode is set to 2 (MMOD = 2).

Parameters

<type>	0 = OFF (no emission spectra are stored) 1 apex 2 all in peak (from begin to end of peak) 3 all spectra 4 all spectra only, but no chromtographical signal is generated! (EXn/EMn are ignored ->can be used for screening applications)
<time>	0 .. 99999.00 runtime in minutes, counted upward from 0 at start of run.

Default Values

0 (= OFF)

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: acquire all emission spectra
Instruction: `EMST 3`
Reply: `RA 0000 EMST 3`

Name

EMSP

Variables for Emission Spectrum

Synopsis

Instruction:	`EMSP <wlfrom>,<wlto>,<wlinc>`
Reply:	`<rc> EMSP <wlfrom>,<wlto>,<wlinc>`
Instruction:	`EMSP?`
Reply:	`<rc> EMSP <wlfrom>,<wlto>,<wlinc>`
Instruction:	`ACT:EMSP?`
Reply:	`<rc> ACT:EMSP <wlfrom>,<wlto>,<wlinc>`
Instruction:	`AT:EMSP <time>,<wlfrom>,<wlto>,<wlinc>`
Reply:	`<rc> AT:EMSP <time>,<wlfrom>,<wlto>,<wlinc>`
Instruction:	`AT:EMSP? <time>`
Reply:	`<rc> AT:EMSP <time>,<wlfrom>,<wlto>,<wlinc>`

Description

Set parameters for acquisition of emission spectra. Taking emission spectra will only be possible if multi wavelength mode is set to 2 (MMOD = 2).
Time for taking a single spectrum could calculated with following formula:

$$t_{spectrum} = \left(\frac{wlto - wlfrom}{wlinc} + 1\right) \times (NumChan + 1) \times 0.0135 \ \text{sec}$$

'NumChan' is the number of active chromatogram channels (see EMn).
Acquiring of emission spectra will only take place if EMST is not equal 0.
(e.g. EMST = 3: acquire all emission spectra)

Parameters

`<wlfrom>`	200 .. 1200nm
`<wlto>`	200 .. 1200nm
`<wlinc>`	1 .. 20nm
`<time>`	0 .. 99999.00 runtime in minutes, counted upward from 0 at start of run.

Default Values

0, 280nm, 400nm, 10nm

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set wavelength range for emission spectrum.
Instruction: `EMSP 280, 460, 10`
Reply: `RA 0000 EMSP 280, 460, 10`

Name

SFIT

Fit spectra range

Synopsis

Instruction:	`SFIT <on/off>`
Reply:	`<rc> SFIT <on/off>`
Instruction:	`SFIT?`
Reply:	`<rc> SFIT <on/off>`

Description

Switch on or off fitting of spectra range. With SFIT = ON the spectra range will be adjusted in that mannor that the peak of the direct straylight coupling will be cut off. The distance is 25nm to the direct straylight

Example:EX1 1,350; EMSP 340,500,5

the emission spectra range is fitted to 375,500,5

EXSP 230,350,5; EM1 1,330

the excitation spectra range is fitted to 230,305,5

Parameters

<on/off>	0 = OFF: don't fit spectra range 1 = ON: fit spectra range

Default Values

0 (= OFF)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: fit spectra range to avoid direct straylight coupling

Instruction: `SFIT 1`

Reply: `RA 0000 SFIT 1`

Name

`SFIL`

Set smooth and spline parameter for spectra

Synopsis

Instruction:	`SFIL <smooth>,<spline>`
Reply:	`<rc> SFIL <smooth>,<spline>`
Instruction:	`SFIL?`
Reply:	`<rc> SFIL <smooth>,<spline>`

Description

Switch on or off smooth or spline algorithm.
Note: smooth and spline algorithm only work in conjunction with the Agilent ChemStation!

Parameters

<smooth>	0 = OFF, 1 = ON
<spline>	0 = OFF, 1 = ON

Default Values

1,1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: switch off smooth and spline of spectra

Instruction: `SFIL 0,0`

Reply: `RA 0000 SFIL 0,0`

Name

EXSC

Default variables for Excitation Scan

Synopsis

Instruction:	`EXSC <wlfrom>,<wlto>,<wlinc>`
Reply:	`<rc> EXSC <wlfrom>,<wlto>,<wlinc>`
Instruction:	`EXSC?`
Reply:	`<rc> EXSC <wlfrom>,<wlto>,<wlinc>`

Description

Set default parameters for acquiring an excitation scan. The acquisition of an excitation scan is started with command STRS -17 (page 725).

These settings also specify the excitation wavelength range of a 3D-scan.

Parameters

`<wlfrom>`	200 .. 1200nm
`<wlto>`	200 .. 1200nm
`<wlinc>`	1 .. 20nm

Default Values

220nm, 300nm, 10nm

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set default variables for excitation scan.

Instruction:	`EXSC 220, 360, 10`
Reply:	`RA 0000 EXSC 220, 360, 10`

Name

EMSC

Default variables for Emission Scan

Synopsis

Instruction:	`EMSC <wlfrom>,<wlto>,<wlinc>`
Reply:	`<rc> EMSC <wlfrom>,<wlto>,<wlinc>`
Instruction:	`EMSC?`
Reply:	`<rc> EMSC <wlfrom>,<wlto>,<wlinc>`

Description

Set default parameters for acquiring an emission scan. The acquisition of an emission scan is started with command STRS -18 (page 726).

These settings also specify the emission wavelength range of a 3D-scan.

Parameters

<wlfrom>	200 .. 1200nm
<wlto>	200 .. 1200nm
<wlinc>	1 .. 20nm

Default Values

0, 280nm, 400nm, 10nm

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	set default variables for emission scan.
	Instruction:	`EMSC 300, 360, 10`
	Reply:	`RA 0000 EMSC 300, 360, 10`

Name

UVON

Flashlamp required

Synopsis

Instruction:	`UVON <yes/no>`
Reply:	`<rc> UVON <yes/no>`
Instruction:	`UVON?`
Reply:	`<rc> UVON <yes/no>`

Description

Determines whether a method requires the flashlamp or not. The module becomes NOT READY (see also ACT:NRDY?) if the flashlamp is switched off while the UVON parameter is set to 1. The UVON? query returns the actual setting of this method parameter.

Parameters

<yes/no>	0 = method does not require flashlamp
	1 = method requires flashlamp

Default Values

1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	method should require flashlamp
	Instruction:	`UVON 1`
	Reply:	`RA 0000 UVON 1`
(b)	Comment:	Query actual setting
	Instruction:	`UVON?`
	Reply:	`RA 0000 UVON 1`

Name

LMOD

Set Flashlamp mode

Synopsis

Instruction:	`LMOD <lampmode>`
Reply:	`<rc> LMOD <lampmode>`
Instruction:	`LMOD?`
Reply:	`<rc> LMOD <lampmode>`

Description

Set flashlamp off or on in different modes. LMOD parameter will come into effect if the flashlamp is switched on (LAMP = 1). Using the command LAMP = 1 will switch on the flashlamp regardless of LMOD parameter, after start of next analysis switching on or off of the flashlamp depends on LMOD.

Parameters

`<lampmode>`	0 = always ON 1 = only ON during analysis (from STRT to STOP)

Default Values

0 (= always ON)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set lamp to always ON

Instruction: `LMOD 0`

Reply: `RA 0000 LMOD 0`

Name

LECO

Set Flashlamp Economy Mode

Synopsis

Instruction: `LECO <on/off>`

Reply: `<rc> LECO <on/off>`

Instruction: `LECO?`

Reply: `<rc> LECO <on/off>`

Description

Swich on or off flashlamp economy mode.

	no multi wavelength mode (MMOD = 0)	multi wavelength mode (MMOD = 1, 2)
standard mode	flash frequence: 296.3 Hz flash energy: low	flash frequence: 74.074 Hz flash energy: high
economy mode	flash frequence: 74.074 Hz flash energy: low	flash frequence: 74.074 Hz flash energy: low

Parameters

`<on/off>` 0 = run in standard mode
1 = run in economy mode

Default Values

0 (= standard mode)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: switch on lamp economy mode
Instruction: `LECO 1`
Reply: `RA 0000 LECO 1`

Name

LREF

Lamp Energy Reference

Synopsis

Instruction:	`LREF <lref_mode>`
Reply:	`<rc> LREF <lref_mode>`
Instruction:	`LREF?`
Reply:	`<rc> LREF <lref_mode>`

Description

Defines whether the lamp energy signal is used as reference for the fluorescence signal or not. The lamp energy signal is measured with reference diode.

Parameters

`<lref_mode>`	0 for OFF
	1 for ON

Default Values

1 for ON

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: use lamp energy signal as reference

Instruction: `LREF 1`

Reply: `RA 0000 LREF 1`

Name

`PMT`

sensitivness of photomultiplier (PMT-gain)

Synopsis

Instruction:	`PMT <pmt_gain>`
Reply:	`<rc> PMT <pmt_gain>`
Instruction:	`PMT?`
Reply:	`<rc> PMT <pmt_gain>`
Instruction:	`ACT:PMT?`
Reply:	`<rc> ACT:PMT <pmt_gain>`
Instruction:	`AT:PMT <time>,<pmt_gain>`
Reply:	`<rc> AT:PMT <time>,<pmt_gain>`
Instruction:	`AT:PMT? <time>`
Reply:	`<rc> AT:PMT <time>,<pmt_gain>`

Description

set the sensitiveness (PMT-gain) of the photomultiplier.

Parameters

<pmt_gain>	0..18, exponent to base 2

Default Values

10

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)

Comment:	set PMT gain to 2^8
Instruction:	`PMT 8`
Reply:	`RA 0000 PMT 8`

Name

BBEH

baseline behavior at wavelength switch

Synopsis

Instruction:	`BBEH <behaviour>`
Reply:	`<rc> BBEH <behaviour>`
Instruction:	`BBEH?`
Reply:	`<rc> BBEH <behaviour>`
Instruction:	`ACT:BBEH?`
Reply:	`<rc> ACT:BBEH <behaviour>`
Instruction:	`AT:BBEH <time>, <behaviour>`
Reply:	`<rc> AT:BBEH <time>,<behaviour>`
Instruction:	`AT:BBEH? <time>`
Reply:	`<rc> AT:BBEH <time>,<behaviour>`

Description

Determines the baseline behaviour at wavelength's switch inside a run.
(outside run the baseline follows the physical acquisition, simular to free)

Parameters

<behaviour>
0 = append: lifts up or down the signal after wavelength switch that the baseline is continous.
1 = free: no adjusting of the signal -> follow physical acquisition
2 = zero: auto-balance after wavelength switch

Default Values

0 (= append)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set the baseline behaviour to zero (auto-balance after wavelength switch)
Instruction: `BBEH 2`
Reply: `RA 0000 BBEH 2`

Name

DDLY
Switch data delay of output data on or off.

Synopsis

Instruction:	`DDLY <on/off>`
Reply:	`<rc> DDLY <on/off>`
Instruction:	`DDLY?`
Reply:	`<rc> DDLY <on/off>`

Description

NOT IMPLEMENTED!   PLEASE ALWAYS USE 'DDLY 0' !

Because of the internal filters within the FLD the cell concentration is delayed. The delay-time dependes at the actual responsetime PKWD. At the analog output the delay is always active. At the digital path the filter-delay normaly is compensated. So the analog retention time depends at the peakwidth setting and the digital retention time is independend from filter setting (peakwidth). With DDLY 1 (=on) also the digital data path can be delayed!

Parameters

`<on/off>`	0 for OFF = data not delayed 1 for ON = data delayed

Default Values

0 (not delayed)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted
`RE 0305` not allowed during run

Examples

(a) Comment: switch data delay off
Instruction: `DDLY 0`
Reply: `RA 0000 DDLY 0`

Name

PKWD
Sets peakwidth of smallest peak expected

Synopsis

Instruction:	`PKWD <peakwidthnum>`
Reply:	`<rc> PKWD <peakwidthnum>`
Instruction:	`PKWD?`
Reply:	`<rc> PKWD <peakwidthnum>`
Instruction:	`ACT:PKWD?`
Reply:	`<rc> ACT:PKWD <peakwidthnum>`
Instruction:	`AT:PKWD <time>, <peakwidthnum>`
Reply:	`<rc> AT:PKWD <time>, <peakwidthnum>`
Instruction:	`AT:PKWD? <time>`
Reply:	`<rc> AT:PKWD <time>, <peakwidthnum>`

Description

The PKWD setpoint is the peakwidth at half hight of the smallest peak expected, it controls the signal filtering.

Parameters

`<peakwidthnum>` 0 .. 7

PKWD	peakwidth at half hight	risetime [10 .. 90 %]	signal data rate	signal slicewidth
0	<0.005 min	<0.12 sec	see DRAT	see DRAT
1	0.005min	0.12 sec		
2	0.01min	0.25 sec		
3	0.025 min	0.5 sec		
4	0.05 min	1 sec		
5	0.1 min	2 sec		
6	0.2 min	4 sec		
7	0.4 min	8 sec		

Default Values

6 (risetime: 4 sec)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set peakwidth parameter
Instruction: `PKWD 6`
Reply: `RA 0000 PKWD 6`

Name

DRAT

Set the output data rate of the detector.

Synopsis

Instruction: `DRAT <data_rate_num>`

Reply: `<rc> DRAT <data_rate_num>`

Instruction: `DRAT?`

Reply: `<rc> DRAT <data_rate_num>`

Description

This instruction set the data output rate of the detector to the workstation. This function is normally used together with the PKWD command.

The data rate cannot be changed during RUN mode of the detector.

Parameters

`<data_rate_num>` independent from flash frequence:

0 :37.04 Hz datarate =	27.0 msec signal slicewidth
1 :37.04 Hz datarate =	27.0 msec signal slicewidth
2 :37.04 Hz datarate =	27.0 msec signal slicewidth
3 :18.52 Hz datarate =	54.0 msec signal slicewidth
4 :9.26 Hz datarate =	108.0 msec signal slicewidth
5 :4.63 Hz datarate =	216.0 msec signal slicewidth
6 :2.31 Hz datarate =	432.0 msec signal slicewidth
7 :1.14 Hz datarate =	864.0 msec signal slicewidth

Default Values

6

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

`RE 0305` not allowed during run

Examples

(a) Comment: set data rate to 9.12Hz
Instruction: `DRAT 4`
Reply: `RA 0000 DRAT 4`

(b) Comment: try to set data rate setting during a run
Instruction: `DRAT 3`
Reply: `RE 0305 DRAT 3`

Name

PDPW

peakdetector peakwidth

Synopsis

Instruction:	`PDPW <peakwidthnum>`
Reply:	`<rc> PDPW <peakwidthnum>`
Instruction:	`PDPW?`
Reply:	`<rc> PDPW <peakwidthnum>`
Instruction:	`ACT:PDPW?`
Reply:	`<rc> ACT:PDPW <peakwidthnum>`
Instruction:	`AT:PDPW <time>,<peakwidthnum>`
Reply:	`<rc> AT:PDPW <time>,<peakwidthnum>`
Instruction:	`AT:PDPW? <time>`
Reply:	`<rc> AT:PDPW <time>,<peakwidthnum>`

Description

The peakdetector controls the automatic acquisition of spectra; it needs to be adjusted to the (expected) peakwidth and to an luminescence threshold (THRS on page 700).
To allow adjustment of the peakdetector settings during a separation, both, the peakdetector peakwidth and the threshold are timeprogrammable.

Parameters

`<time>`	0 .. 99999.00 runtime in minutes, counted upward from 0 at start of run.
`<peakwidth>`	0 .. 7 same meaning as for PKWD on page 697

Default Values

4

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: Change (expected) peakwidth setting for the peakdetector to 3 at 5.5 minutes.
Instruction: `AT:PDPW 5.5,3`
Reply: `RA 0000 AT:PDPW 5.50,3`

Name

THRS

set peakdetector threshold

Synopsis

Instruction:	`THRS <threshold>`
Reply:	`<rc> THRS <threshold>`
Instruction:	`THRS?`
Reply:	`<rc> THRS <threshold>`
Instruction:	`ACT:THRS?`
Reply:	`<rc> ACT:THRS <threshold>`
Instruction:	`AT:THRS <time>,<threshold>`
Reply:	`<rc> AT:THRS <time>,<threshold>`
Instruction:	`AT:THRS? <time>`
Reply:	`<rc> AT:THRS <time>,<threshold>`

Description

The peakdetector controls the automatic acquisition of spectra; it needs to be adjusted to the (expected) peakwidth of the detector PKWD on page 697 and PDPW on page 699 and to a luminescence threshold described here.

To allow adjustment of the peakdetector settings during a separation, both, the peakdetector peakwidth and the threshold are timeprogrammable.

Parameters

`<time>`	0 .. 99999.00 runtime in minutes, counted upward from 0 at start of run.
`<threshold>`	0.001 .. 1000.000 threshold in luminescence units (LU)

Default Values

5.000 LU

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set peakdetector threshold to 10 LU.
Instruction: `THRS 10`
Reply: `RA 0000 THRS 10.000`

Name

ACHn

select the source channel for analog output

Synopsis

Instruction:	`ACH1 <sigtype>`
Reply:	`<rc> ACH1 <sigtype>`
Instruction:	`ACH2 <sigtype>`
Reply:	`<rc> ACH2 <sigtype>`
Instruction:	`ACH1?`
Reply:	`<rc> ACH1 <sigtype>`
Instruction:	`ACH2?`
Reply:	`<rc> ACH2 <sigtype>`

Description

With this instruction a signal described in special section (see page 746) can be routed to the analog output.

Parameters

`<sigtype>`	1 .. 999

Default Values

1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	switch signal type 3 to analog output 1
	Instruction:	`ACH1 3`
	Reply:	`RA 0000 ACH1 3`
(b)	Comment:	query the actual channel of the anlog output 1
	Instruction:	`ACH1?`
	Reply:	`RA 0000 ACH1 3`

Name

AZEn

set zero offset of analog outputs

Synopsis

Instruction:	`AZE1 <offset%>`
Reply:	`<rc> AZE1 <offset%>`
Instruction:	`AZE2 <offset%>`
Reply:	`<rc> AZE2 <offset%>`
Instruction:	`AZE1?`
Reply:	`<rc> AZE1 <offset%>`
Instruction:	`AZE2?`
Reply:	`<rc> AZE2 <offset%>`

Description

Sets the zero offset voltage for analog output 1 (2) to a percentage of <offset%> of the full scale voltage. For information on the full scale voltage see ASC1 (ASC2) instruction.

Parameters

`<offset%>` 1 .. 99 [%]

Default Values

5

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

`RE 0502` out of range

Examples

(a) Comment: set zero offset voltage for analog output 1to 50% of full scale

Instruction: `AZE1 50`

Reply: `RA 0000 AZE1 50`

Name

AATn

select analog output attenuation

Synopsis

Instruction:	`AAT1 <att_id>`
Reply:	`<rc> AAT1 <att_id>`
Instruction:	`AAT2 <att_id>`
Reply:	`<rc> AAT2 <att_id>`
Instruction:	`AAT1?`
Reply:	`<rc> AAT1 <att_id>`
Instruction:	`AAT2?`
Reply:	`<rc> AAT2 <att_id>`

Description

The FLD offers two analog outputs: signal A at analog output 1 and signal B at analog output 2. The attenuation can be changed in power of 2 between 1600LU and 0.2LU.

Parameters

`<att_id>` 0..13

full range output is 2^{att_id-9} * 100LU.

13: 1600LU output, 9: 100LU output, ..., 0: $\frac{100}{512}$ *LU output

Default Values

9 (100 LU full range output)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

`RE 0502` out of range

Examples

(a) Comment: set 400 LU output at analog output 1:

Instruction: `AAT1 11`

Reply: `RA 0000 AAT1 11`

(b) Comment: query att selection for output 1

Instruction: `AAT1?`

Reply: `RA 0000 AAT1 9`

Name

POL

Set polarity of rawdata at analog output

Synopsis

Instruction:	`POL <polarity>`
Reply:	`<rc> POL <polarity>`
Instruction:	`POL?`
Reply:	`<rc> POL <polarity>`

Description

This instruction set the polarity of data handling for the detector. For positive polarity (0=normal = default) the detector outputs positive data if detecting higher fluorescence compared to the value during balance. With negative polarity (1) the detctor outputs positive data if detecting lower fluorescence compared to the value during balance.

This function affects all data generated from SIG1 to SIGn.

(only chromatoghraphic data are affected!)

Parameters

`<polarity>`	0 for positive polarity, 1 for negative polarity

Default Values

0 (positive polarity)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set negative polarity

Instruction: `POL 1`

Reply: `RA 0000 POL 1`

Name

DLAY
delay for phosphorescence

Synopsis

Instruction:	`DLAY <delay>`
Reply:	`<rc> DLAY <delay>`
Instruction:	`DLAY?`
Reply:	`<rc> DLAY <delay>`

Description

Determines the time between flash and Start_of_Integration. Will only be valid if acquisition mode is set to phosphorescence (AMOD = 1).

Parameters

<delay>	0 .. 10,000.0 measured in µ-seconds, step width is 0.1 µsec.

Default Values

50 µsec

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)

Comment:	set delay to 100 µsec
Instruction:	`DLAY 100`
Reply:	`RA 0000 DLAY 100`

Name

GATE

gate for integration in phosphorescence acquisition mode

Synopsis

Instruction:	`GATE <gate_time>`
Reply:	`<rc> GATE <gate_time>`
Instruction:	`GATE?`
Reply:	`<rc> GATE <gate_time>`

Description

Determines the time between Start_of_Integration and End_of_Integration. Will only be valid if acquisition mode is set to phosphorescence (AMOD = 1).
If flash frequence is 370 Hz (see LECO on page 692) and (DLAY + GATE) is greater than 4.5msec then the flash frequence will automatically reduced to 74 Hz!

Parameters

<gate_time>	0 .. 10,000.0 measured in µ-seconds, step width is 0.1 µsec.

Default Values

200 µsec

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: set gate for integration to 500 µsec
Instruction: `GATE 500`
Reply: `RA 0000 GATE 500`

25.2. FLD System Parameter

Name

D2ON
Turn on lamp automatically after power on

Synopsis

Instruction:	`D2ON <on/off>`
Reply:	`<rc> D2ON <on/off>`
Instruction:	`D2ON?`
Reply:	`<rc> D2ON <on/off>`

Description

see title.

Parameters

`<on/off>`	0: do not automatically turn on flash lamp 1: turn on automatically

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)

Comment:	set value to not automatically turn on flash lamp
Instruction:	`D2ON 0`
Reply:	`RA 0000 D2ON 0`

Name

RAWS
select the signals to be stored

Synopsis

Instruction:	RAWS <signalset>
Reply:	<rc> RAWS <signalset>
Instruction:	RAWS?
Reply:	<rc> RAWS <signalset>

Description

The FLD continuously generates different signals. A part of these signals can be accessed by a special Rawdata CU. The signals are described in special section (see page 746).
Although the RAWS setting is always accepted, it becomes active only at the time when the rawdata storing begins. Storing signals starts when the RAWD:STRT instruction is accepted or when an analytical run starts.

Parameters

<signalset> 0 .. 2^25
1 (bit 0 set) means: store signal 1
2 (bit 1 set) means: store signal 2
4 (bit 2 set) means: store signal 3
8 (bit 3 set) means: store signal 4
... etc.
any combination is allowed,
0 means: no signal will be stored (OFF)
11 means: signal type 1, 2 and type 4 (1+2+8 = 11) will be stored.

Default Values

1

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	select signal type 1 and 2 to be stored
	Instruction:	RAWS 3
	Reply:	RA 0000 RAWS 3
(b)	Comment:	query signal select setting
	Instruction:	RAWS?
	Reply:	RA 0000 RAWS 3

Name

ASCn
select analog output voltage range

Synopsis

Instruction:	`ASCn <range_id>`
Reply:	`<rc> ASCn <range_id>`
Instruction:	`ASCn?`
Reply:	`<rc> ASCn <range_id>`

Description

The FLD offer two analog outputs: signal A at analog output 1 and signal B at analog output 2. The voltage range can be switched between 0.1V and 1V full scale. This setpoint is not affected when changing the FLD method.

Parameters

`<range_id>`	0: 0.1 V full scale 1: 1 V full scale

Default Values

1 (1 V full scale)

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RE 0502 out of range

Examples

(a)	Comment:	select 0.1 V for output 1 (signal A)
	Instruction:	`ASC1 0`
	Reply:	`RA 0000 ASC1 0`
(b)	Comment:	query range selection for output 2
	Instruction:	`ASC2?`
	Reply:	`RA 0000 ASC2 0`

Name

PRPR

Define Prepare Program

Synopsis

Instruction: PRPR <cosy_list_num>
Reply: <rc> PRPR <cosy_list_num>
Instruction: PRPR?
Reply: <rc> PRPR <cosy_list_num>

Description

PRPR defines the cosylist to be used as prepare list. Both the PREP instruction and the default main list use this list. The internal reference is -6. This means that in a main cosylist the list number -6 has to be started to perform a prepare .

Syntax: cosy:run <main list number>, -6 [,sync]

Default list -9:

Cmd	Note
cosy:new -9, 1	Create the cosylist in RAM (", 1"). It is removed when the module is power-cycled.
cosy:iseq -9, "wwrm"	Wait till lamp warmup is done
cosy:iseq -9, "prep:strt"	Do statechange and clear diagnosis buffer
cosy:iseq -9, "prep:mott 0"	Turn excitation motor
cosy:iseq -9, "prep:mott 1"	Turn emission motor
cosy:iseq -9, "prep:lica"	Lamp intensity calibration
cosy:wait -9, 0.07	Wait 4.2sec to stabalize baseline
cosy:iseq -9, "prep:done"	Do statechange and enter diagnosis buffer.

Parameters

<cosy_list_num> -32768 .. 32767, negative numbers are automatically interpreted as command to reactivate the default list -9

Default Values

-9

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set default prepare list
Instruction: `PRPR -9`

	Reply:	`RA 0000 PRPR -9`
(b)	Comment:	set privately defined prepare list
	Instruction:	`PRPR 5`
	Reply:	`RA 0000 PRPR 5`
(c)	Comment:	set any negativ numbered list
	Instruction:	`PRPR -3`
	Reply:	`RA 0000 PRPR -9`

Name

POPR

Define Prepare Program

Synopsis

Instruction: -POPR <cosy_list_num>
Reply: <rc> POPR <cosy_list_num>
Instruction: POPR?
Reply: <rc> POPR <cosy_list_num>

Description

POPR defines the cosylist to be executed after the postrun elapsed when working with the default main cosylist. The internal reference is -7. This means that in a main cosylist the list number -7 has to be started to perform a prepare .

Syntax: cosy:run <main list number>, -7 [,sync]

Default list -9:

Cmd	Note
cosy:new -9, 1	Create the cosylist in RAM (", 1")
cosy:iseq -9, "wwrm"	Wait till lamp warmup is done
cosy:iseq -9, "prep:strt"	Do statechange and clear diagnosis buffer
cosy:iseq -9, "prep:mott 0"	Turn excitation motor
cosy:iseq -9, "prep:mott 1"	Turn emission motor
cosy:iseq -9, "prep:lica"	Lamp intensity calibration
cosy:wait -9, 0.07	Wait 4.2sec to stabalize baseline
cosy:iseq -9, "prep:done"	Do statechange and enter diagnosis buffer.

Parameters

<cosy_list_num> -32768 .. 32767, negative numbers are automatically interpreted as command to reactivate the default list -9

Default Values

-9

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: set default prepare list
Instruction: `POPR -9`
Reply: `RA 0000 POPR -9`

(b) Comment: set privately defined prepare list
Instruction: `POPR 5`
Reply: `RA 0000 POPR 5`

(c) Comment: set any negativ numbered list
Instruction: `POPR -3`
Reply: `RA 0000 POPR -9`

Name

NDCL

Cell needed/not needed

Synopsis

Instruction:	`NDCL <needed>`
Reply:	`<rc> NDCL <needed>`
Instruction:	`NDCL?`
Reply:	`<rc> NDCL <needed>`

Description

NDCL defines wether an error is generated if the cell is removed during the analysis.

Parameters

<needed>	0 for not needed, 1 for needed

Default Values

1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set cell not needed
Instruction: `NDCL 0`
Reply: `RA 0000 NDCL 0`

Name

DMUL

Response Multiply Factor of Data Acquisition

Synopsis

Instruction: `DMUL <mult_factor>`
Reply: `<rc> DMUL <mult_factor>`
Instruction: `DMUL?`
Reply: `<rc> DMUL <mult_factor>`

Description

The DMUL-factor changes the outcomming data of the fluorescence-, phosphorescence or chemoluminescence response by multiplying the original value with the given factor.
This factor can be used to change the output of instruments with different response to align each other.
! Using this factor different of 1.00 will impact the internal and external data processing !
a) a factor > 1.00 will increase the signal, so the dynamic range will be reduced. Check with the highest concentration that the data will be consistent and no overflow will occured!
b) a factor < 1.00 will decrease the signal, it could be (depending on the background of the cromatographical system), that now the decreased signal will fall into the quantisation of the analog to digital converter: in that case the noise will not be decreased by the given factor! (-> so Signal/Noise is increased!)
[example: with a factor 0.1at a specific chrommatographical system the signal decrease by 10, the noise get some quantisation effect (depending on the given system background!) and decreased only by 5. So Signal to Noise changed to (1/10)/(1/5) = 1/2 -> S/N will be half of the original S/N]

Recommendation: Use factors near 1.00, for example: ~ 0.25 ... 4.00
Strong Recommendation: never use factors smaller 0.1 or larger 10.00

Caution 1: Available since firmware revision 5.09
Caution 2: The DMUL-value is stored in the flash memory of the detector.
So at power cycles, coldstarts and also firmware updates the value will be stable.
Caution 3: The DMUL value is not a classical sys-value (not listed with LIST "SYS"),
because this values will be set back to default with a firmware update (coldstart is executed!)

Parameters

`<mult_factor>` 0.01 .. 100.00

Default Values

1.00

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual response factor
Instruction: `DMUL?`
Reply: `RA 0000 DMUL 1.24`

(b) Comment: change the response factor
Instruction: `DMUL 0.85`
Reply: `RA 0000 DMUL 0.85`

25.3. FLD Control Commands

25.3.1. Principles of Calibration

The calibration algorithm is implemented as a CosyList. The number of this CosyList is -10. Starting the CosyList the detector performs state changes to ANALYSIS (ES 0110) and to CALIBRATION (ES 6689). The actual state of the currently running calibration can be obtained with ACT:WCAL? (a return value of -1 means no calibration is running). The end of calibration is notified by the event CALIBRATION_FINISH (EV 6611). With the query TEST:WCAL? the actually measured calibration values can be asked. The values show the difference between the last performed calibration and the calibration before for both excitation and emission in nm. Saving the calibration values into diagnostic buffer RCAL and adjusting the hardware is done with the command WCAL. The query WCAL? returns the last entry from diagnostic buffer RCAL. Complete listing of the diagnostic buffer RCAL can be obtained with DIAG:LIST "RCAL".
The following table lists all events which show the actual state of the calibration procedure:

state change events	CALIBRATION	ES 6689	calibration procedure is running
	NO_CALIBRATION	ES 6690	no calibration
	CALIBRATION_LOST	ES 6691	no calibration values available, calibration is required that the detector becomes ready
info events	CALIBRATION_STEP	EV 6610,0	EX rot 0..450 deg, step 0.1 deg
		EV 6610,1	EX rot ZeroOrder +/- 1.0 deg, step 0.01 deg
		EV 6610,2	EX pos ZeroOrder +/- 3.0 deg, step 0.20 deg
		EV 6610,3	EX pos ZeroOrder +/- 1.0 deg, step 0.01 deg
		EV 6610,4	EM rot 0..450 deg, step 0.1 deg
		EV 6610,5	EM rot 1st Order +/- 1.0 deg, step 0.01 deg
		EV 6610,6	EM rot ZeroOrder +/- 1.0 deg, step 0.01 deg
		EV 6610,7	EM pos 1st Order +/- 3.0 deg, step 0.20 deg
		EV 6610,8	EM pos 1st Order +/- 1.0 deg, step 0.01 deg
	CALIBRATION_FINISH	EV 6611	calibration finished without error
	CALIBRATION_ABORT	EV 6612	calibration procedure aborted
error events	CALIBRATION_ERROR	EE 6703,1	no ZeroOrder peak found in first EX scan
		EE 6703,2	EX scan in saturation of reference diode
		EE 6703,3	peaks of EX scan are not separated
		EE 6703,4	relation of 1st and -1st order peaks in EX scan is wrong
		EE 6703,5	EM 1st order peak in saturation of PMT

error events	CALIBRATION_ERROR	EE 6703,6	1st order peak in EM scan is too wide
		EE 6703,9	distance between peaks in EM scan is wrong
		EE 6703,10	relation of 1st and -1st order peaks in EM scan is wrong
		EE 6703,11	peaks of EM scan are not separated
		EE 6703,16	not enough memory to examine scan
		EE 6703,5602	could not start scan: not in POS-POS mode
		EE 6703,5603	could not start scan: not in ROT-POS mode
		EE 6703,5604	could not start scan: not in POS-ROT mode

25.3.2. Calibration Control Commands

Name

STRS -10
Start calibration

Synopsis

Instruction: STRS -10
Reply: <rc> STRS -10

Description

The wavelength calibration procedure is implemented as internal cosy list -10. This command starts this special cosy list.

Parameters

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: start calibration
Instruction: STRS -10
Reply: RA 0000 STRS -10

Name

TEST:WCAL?

Query for the result of the last calibration

Synopsis

Instruction: `TEST:WCAL?`

Reply: `<rc> TEST:WCAL <Ex calib value>, <Em calib value>`

Description

The instruction TEST:WCAL? lists the last calculated/measured calibration values.

Parameters

<Ex calib value>

<Em calib value> difference to last stored calibvalue, measured in nm

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query calibration values

Instruction: `TEST:WCAL?`

Reply: `RA 0000 TEST:WCAL 1.3, -0.8`

Name

`WCAL`

Save the values of the last calibration

Synopsis

Instruction: `WCAL`

Reply: `<rc> WCAL`

Description

The instruction WCAL saves the last calculated/measured calibration values into non volatile ram and adjusts the hardware. The values are still available as last entry of the diagnostics buffer WCAL.

Parameters

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: save calibration values

Instruction: `WCAL`

Reply: `RA 0000 WCAL`

Name

`WCAL?`

Query for the last stored calibration values

Synopsis

Instruction: `WCAL?`

Reply: `<rc> WCAL <Ex calib value>, <Em calib value>`

Description

The instruction WCAL? lists the last calibration values out of diagnosis buffer RCAL.

Parameters

<Ex calib value>

<Em calib value> difference between the last stored calibvalue and the one before, measured in nm

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query last stored calibration values

Instruction: `WCAL?`

Reply: `RA 0000 WCAL 1.3, -0.8`

Name

`ACT:WCAL?`

Query for the actual status of ongoing calibration

Synopsis

Instruction: `ACT:WCAL?`

Reply: `<rc> ACT:WCAL <calib step>`

Description

The instruction ACT:WCAL? returns the current step of the ongoing calibration.

Parameters

<CalibStep> -1: no calibration in progress
0 .. 8: steps of a calibration

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query calibration step
Instruction: `ACT:WCAL?`
Reply: `RA 0000 ACT:WCAL -1`

Name

TEST:WCAL:ABRT
abort current calibration

Synopsis

Instruction: TEST:WCAL:ABRT
Reply: <rc> TEST:WCAL:ABRT

Description

The instruction TEST:WCAL:ABRT aborts the current calibration procedure.

Parameters

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: abort current calibration
Instruction: TEST:WCAL:ABRT
Reply: RA 0000 TEST:WCAL:ABRT

25.3.3. Principles of Taking Scans

The way how any scan is taken is implemented as a CosyList. For every type of scan a special CosyList is available:

CosyList number	type of scan
-17	excitation scan
-18	emission scan
-19	lamp characteristic scan
-20	3D-scan

Taking a scan is performed by starting the CosyList with the command STRS <CosyList no>.
The wavelength range for the excitation-, emission- and 3D-scans is specified by the settings of the commands EXSC <wlfrom>,<wlto>,<wlinc> resp. EMSC <wlfrom>,<wlto>,<wlinc>. For the lamp characteristic scan always the full excitation wavelength range (200nm - 1200nm) is used and can not be modified.
The scan is taken in the environment of an analysis, the scan (or multiple scans in case of a 3D-scan) is sent out at the rawdata cu. Additional the 2-dimensional scans are stored in a diagnostic buffer. The diagnostic buffer "STSX" stores the last excitation scan, "STSM" the last emission scan and "STLC" stores the last lamp characteristic scan.
Begin and end of the acquisition of a scan are indicated by the events EV 7051 (begin) and EV 7052 (end). For a 3D-scan additional the state change events ES 6695 (3D-SCAN) and ES 6696 (NO_3D-SCAN) are sent out.
Every scan is taken in the appropriate rotation mode: the excitation grating rotates in case of excitation scan, lamp characteristic scan and 3D-scan, for an emission scan the emission grating rotates. For an excitation or emission scan the wavelength of the non-rotating grating is specified by the method (parameter EX1 resp. EM1). The lamp characteristic scan is measured using the reference diode, so the wavelength of the emission grating doesn’t matter.

25.3.4. SCAN Commands

Name

STRS -17
Take an excitation scan

Synopsis

Instruction: `STRS -17`
Reply: `<rc> STRS -17`

Description

With this instruction the acquisition of an excitition scan is started. The scan is sent out at rawdata cu and is additional stored into the diagnosis buffer "STSX". The excitation wavelenght range is specified by the command EXSC <wlfrom>,<wlto>,<wlinc>, the emission wavelength is obtained from the current method (EM1 1,<wl>).

Parameters

<none>

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: take an excitation scan
Instruction: `STRS -17`
Reply: `RA 0000 STRS -17`

Name

STRS -18

Take an emission scan

Synopsis

Instruction: `STRS -18`

Reply: `<rc> STRS -18`

Description

With this instruction the acquisition of an emission scan is started. The scan is sent out at rawdata cu and is additional stored into the diagnosis buffer "STSM". The emission wavelenght range is specified by the command EMSC <wlfrom>,<wlto>,<wlinc>, the excitation wavelength is obtained from the current method (EX1 1,<wl>).

Parameters

<none>

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: take an emission scan

Instruction: `STRS -18`

Reply: `RA 0000 STRS -18`

Name

STRS -19

Take a lamp characteristic scan

Synopsis

Instruction: STRS -19

Reply: <rc> STRS -19

Description

With this instruction the acquisition of a lamp characteristic scan is started. The scan is sent out at rawdata cu and additional stored into the diagnosis buffer "STLC".

This scan measures always the full excitation wavelenght range (200nm - 1200nm). The wavelenght range can not modified.

The lamp characteristic scan is measured using the reference diode!

Parameters

<none>

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: take a lamp characteristic scan

Instruction: STRS -19

Reply: RA 0000 STRS -19

Name

`STRS -20`
Take a 3D-scan

Synopsis

Instruction: `STRS -20`
Reply: `<rc> STRS -20`

Description

With this instruction the acquisition of a 3D-scan is started. The 3D-scan is build up by multiple excitation scans which are sent out at rawdata cu. The wavelenght ranges for excitation and emission are specified by the commands EXSC <wlfrom>,<wlto>,<wlinc> resp. EMSC <wlfrom>,<wlto>,<wlinc>.

Parameters

<none>

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: take a 3D-scan
Instruction: `STRS -20`
Reply: `RA 0000 STRS -20`

Name

SCAN:ABRT
Abort current scan taken.

Synopsis

Instruction: SCAN:ABRT
Reply: <rc> SCAN:ABRT

Description

Any scan started can be aborted without storage.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: abort current scan
Instruction: SCAN:ABRT
Reply: RA 0000 SCAN:ABRT

25.3.5. Prepare Commands

Name

PREP
Start Prepare

Synopsis

Instruction: PREP
Reply: <rc> PREP

Description

With this instruction the 'prepare cosy list is started'. The module's default main cosy list also starts this list. The PREP command is only provided for interactively starting the prepare cycle. A new cosy list can be defined to what ever is needed, and than linked to the prepare list with the instruction PPRG <list number>. (see system variables)
The default prepare list containes the following instructions:

PREP:STRT	just to invoke a statechange to be shown on the controllers
PREP:MOTT 0	turn excitation grating
PREP:MOTT 1	turn emission grating
PREP:LICA	lamp intensity calculation (lifetime correction) (empty so far)
PREP:DONE	just to exit the prepare state

Parameters

<none>

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: interactively start a prepare cycle
Instruction: PREP
Reply: RA 0000 PREP

25.3.6. Other control commands

Name

CELL?
Query for the installed cell

Synopsis

Instruction: `CELL?`
Reply: `<rc> CELL <cell type>`

Description

The installed cell can be queried with the CELL? instruction. With each change of the cell, the event EV 6605,<cell type> is sent and a new entry is done to the diagnostic buffer CELL that holds the last 10 cell changes in none volatile ram. Associated with each cell type is a diagnostic counter, that counts in seconds the time a cell of this type is installed. When the cell is removed during an analysis, the error EE 6702 is generated, With the system variable NDCL 0, this error generation can be disabled.

Parameters

cell type 0 for 'no cell', 1 for standard cell, 2 for cuvette, 3 for futur use

Default Values

not applicaple

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query cell type
Instruction: `CELL?`
Reply: `RA 0000 CELL 1`

Name

CPMT:STRT
Pmt calculation

Synopsis

Instruction:	CPMT:STRT
Reply:	<rc> CPMT:STRT
Instruction:	CPMT:ABRT
Reply:	<rc> CPMT:ABRT
Instruction:	CPMT?
Reply:	<rc> CPMT <pmt value>

Description

With the instruction CPMT:STRT the optimal pmt value is calculated. The calculation takes several seconds . The process can be aborted with CPMT:ABRT or ABRT. When the calculation is done the event EV 6613,<pmt value> is generated. With the query CPMT? the last calculated value can be queried,this value is stored in non volatile ram. The last 20 calculation results with the time when they were terminated are available in the diagnostics buffer CPMT.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: invoke pmt calculation
Instruction: CPMT:STRT
Reply: RA 0000 CPMT:STRT

Name

ADOV
ADC overflow detection

Synopsis

Instruction: `ADOV <on/off>`
Reply: `<rc> ADOV <on/off>`
Instruction: `ADOV?`
Reply: `<rc> ADOV <on/off>`

Description

The instruction ADOV will switch on and off the overflow detecton of the ADC. Default is detection is on.
When the ADC overflow detection is switched on the signal of channel 1 is monitored. In case that the ADC switches into overflow the event EV 6618 EV_FLD_ADC_OVERFLOW is sent and the NotReady-condition "ADC overflow" is set. Switching back to normal operation the event EV 6619 EV_FLD_ADC_NO_OVERFLOW is sent and the NotReady-condition is cleared.

Parameters

<on/off> 0 = OFF, 1 = ON

Default Values

1 (= ON)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: switch off ADC overflow detection
Instruction: `ADOV 0`
Reply: `RA 0000 ADOV 0`

25.4. FLD Actuals

Name

`ACT:VAL?`
Output a internal or chromatographic or other value

Synopsis

Instruction: `ACT:VAL? <sigtype>`
Reply: `<rc> ACT:VAL <sigtype>,<value>`

Description

This function outputs a chromatographic or internal value of the detector for service or diagnostic use. If <sigtype> is not defined a 0 will be returned.

Parameters

`<sigtype>`	see sigtype special description: page 746
<value>	+/- 2^31 (long: depending at <sigtype>)

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: look at current sigtype 4
Instruction: `ACT:VAL? 4`
Reply: `RA 0000 ACT:VAL 4,200`

25.5. FLD Statemachines

Name

LampSTM

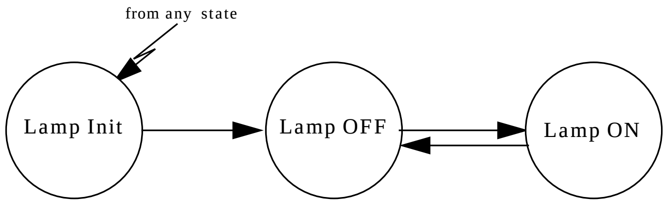
Statemachine of Lamp behaviour

Description

This statemachines build up the lamp states and possible changes.

State	State-Id	State Change event	Description
LampOff	0	ES 7403	Lamp is off
LampOn	1	ES 7404	Lamp is on
LampInit	2	none	Lamp initialization

State Diagram



Name

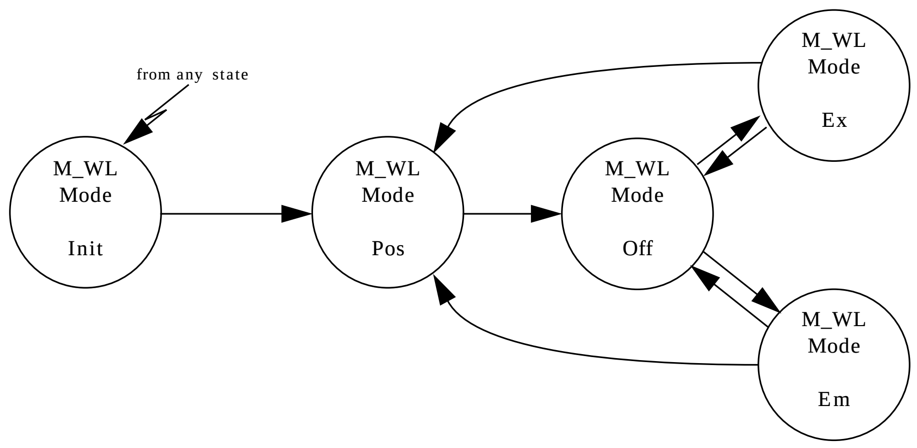
MultiWavelengthSTM
Statemachine of Multi Wavelength behaviour

Description

This statemachine builds up the multi wavelength states and possible changes.

State	State-Id	State Change event	Description
M_WL_Mode_Init	0	ES 6650	Initialization of gratings
M_WL_Mode_Pos	1	ES 6651	both gratings in position mode
M_WL_Mode_Ex	2	ES 6652	multi excitation mode
M_WL_Mode_Em	3	ES 6653	multi emission mode
M_WL_Mode_Off	4	ES 6654	grating(s) not in defined position

State Diagram



Name

PMTCalcSTM

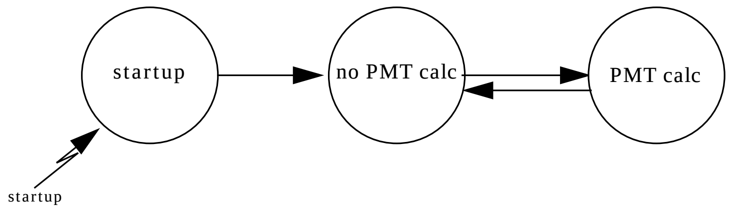
Statemachine of PMT gain calculation

Description

This statemachines build up the PMT gain calculation states and possible changes.

State	State-Id	State Change event	Description
PMT gain calculation	0	ES 6659	calculating the most appropriate PMT gain
no PMT gain calculation	1	ES 6660	no PMT gain calculation
startup	2	none	during startup

State Diagram



Name

AcqModeSTM

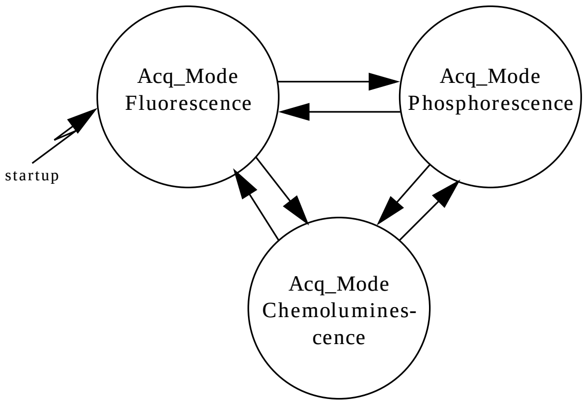
Statemachines of Acquisition Mode behaviour

Description

These statemachines build up the acquisition mode states and possible changes.

State	State Id	State Change event	Description
Acq_Mode_Fluor	0	ES 6671	fluorescence mode
Acq_Mode_Phos	1	ES 6672	phosphorescence mode
Acq_Mode_Chem	2	ES 6673	chemoluminescence mode

State Diagram



Name

LecoSTM

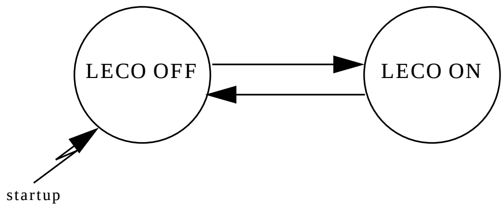
Statemachine of Lamp Economy mode behaviour

Description

This statemachines build up the lamp economy mode states and possible changes.

State	State-Id	State Change event	Description
LecoOff	0	ES 6680	Lamp Economy mode is off
LecoOn	1	ES 6681	Lamp Economy mode is on

State Diagram



Name

LrefSTM

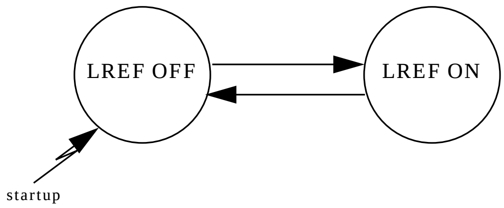
Statemachine of Lamp energy reference mode behaviour

Description

This statemachines build up the lamp energy reference mode states and possible changes.

State	State Change event	Description
LrefOn	ES 6685	Lamp energy reference mode is on
LrefOff	ES 6686	Lamp energy reference mode is off

State Diagram



Name

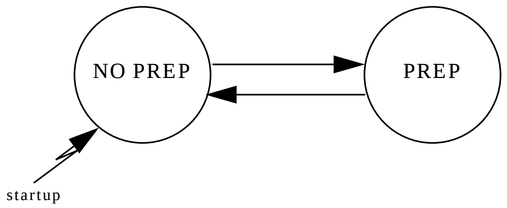
PrepSTM
Statemachine of Prepare behaviour

Description

This statemachines build up the Prepare states and possible changes.

State	State-Id	State Change event	Description
Prepare	0	ES 6687	during Prepare
No Prepare	1	ES 6688	no Prepare

State Diagram



Name

CalibSTM

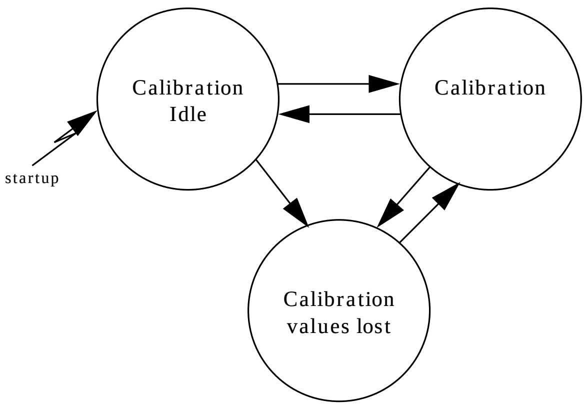
Statemachines of Wavelength calibration behaviour

Description

These statemachines build up the wavelength calibration states and possible changes.

State	State Id	State Change event	Description
Calibration	0	ES 6689	calibration procedure is running
Calibration Idle	1	ES 6690	not during calibration
Calibration Lost	2	ES 6691	no calibration values available
startup	3	none	during startup

State Diagram



Name

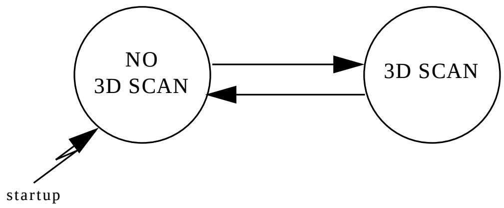
Scan_3dSTM
Statemachines of 3D-Scan behaviour

Description

These statemachines build up the states of a 3D-Scan and possible changes.

State	State Change event	Description
3D-Scan	ES 6695	taking 3D-Scan
no 3D-Scan	ES 6696	not taking 3D-Scan

State Diagram



25.6. FLD Rawdata File Description

This description is a supplement to: Rawdata File on page 253.
The Agilent 1200 FLD rawdata file records can be signal records. Up to 4 different signals: RA, RB, RC, RD or spectra records SL.

25.6.1. Signal Records

The Agilent 1200 FLD generates up to 4 signals. The rawdata file can store these signals as records RA, RB, RC and RD, where RA corresponds to SIG1, and so on. To select, which signal should be stored, see instruction RAWS on page 258. For selection of the data representation (decimal, hexa-decimal or decimal) see instruction RAWF on page 259.

The data in the signal records are Fluorescence data.
The data representation is 100*2^-24 LU per count (1 count is 100*2^-24LU~= 5.96uLU).

25.6.2. Spectra Records

Every spectra record contains parameter information and the fluorescence data. The fluorescence data representation is 100*2^-24 LU per count (1 count is 100*2^-24LU~= 5.96uLU).

#	description	unit	example (DEC format)
	header (always 12 bytes fix, independent of the used format)		SL DEC,0135;
	number of parameter values following (currently 21)		21,
1	time of start of analysis in seconds since 1970	s	848075283,
2	time of acquisition relative to start of analysis	ms	7605,
3	time of acquisition relative to start of run	ms	4030,
4	time used to acquired this spectrum	ms	200,
5	center wavelength of first data-point	pm	230000,
6	step between two adjacent data points	pm	2000,
7	Smooth control	-	1,
8	Smooth parameter	-	1,
9	undefined	-	0,
10	complementary wavelength	pm	450000,
11	this spectrum belongs to a 3D-scan	-	0,
12	spectrum type (excitation or emission)	0: ex 1: em	0,

#	description	unit	example (DEC format)
13	peakdetector status 0-3: unused 4: inflection point upslope 1 5: inflection point upslope 2 6: inflection point donwslope 1 7: inflection point donwslope 2 8: unused 9: begin of peak 10: top of peak 11: top of small peak 12: cance 13: valley 14: end of peak 15: unknown 16: force baseline 80: periodic 81: all in peak) NOTE: Although this rawdata format description also applies for the specta data out of the SPECTRA MONITOR CU, the peakdetector information is available only with the data out of the RAW-DATA CU, otherwise it is 0.	-	81,
14	current peakwidth (0 .. 7 , see PKWD on page 697)	-	5,
15	peakdetector peakwidth (0 .. 7 , see PDPW on page 699)	-	4,
16	Spline control	-	1,
17	Spline parameter	-	1,
18	undefined	-	0,
19	cell type	-	1,
20	undefined	-	0,
21	number of data points following (N)	-	356,
1	1. data point	5.96uLU	2414,
...	...	...	...
N	Nth data point (last)	5.96uLU	6345

25.7. FLD Signals Description

Description

The table below describes all signals available in the detector. Signals can be accessed by:
ACT:VAL? <sigtype> command to query a signal
ACH1 <sigtype> , ACH2 <sigtype> to route a signal to the analog outputs
RAWS <signalset> to get signals by rawdata
LU = luminescence units

Table

signal type	num	RAWS		description	rawdata unit	actual units	analog scale / 1V	file type
raw-data output	1	1	A	EX1 / EM1	2^{-24} LU	LU	1 LU	.ch
	2	2	B	EX2 / EM1 or EX1 / EM2	2^{-24} LU	LU	1 LU	.ch
	3	4	C	EX3 / EM1 or EX1 / EM3	2^{-24} LU	LU	1 LU	.ch
	4	8	D	EX4 / EM1 or EX1 / EM4	2^{-24} LU	LU	1 LU	.ch
	5	16	E	RESERVED [for Absorbance]	2^{-24} AU	AU	1 AU	.ch
	6	32	F	Signal without reference [EX1/EM1]	2^{-24} LU	LU	1 LU	.ch
	7	64	G	Reference only in LU [EX1]	2^{-24} LU	LU	1 LU	.ch
	8	128	H	Sample counts [EX1/EM1]	1	counts	2^{16}counts	.ch
	9	256	I	Reference counts [EX1]	1	counts	2^{16}counts	.reg
	10	512	J	Test Chromatogram output	2^{-24} LU	LU	1 LU	.reg
	11	1024	K	Analog Output Test Function				.reg
	12	2048	L	Sample/Ref counts (raw, unfiltered)	1	counts		.reg
	13	4096	M	Diagnostic Data (sample min/max)	1	counts		.reg
	14	8192	N	Diagnostic Data (reference min/max)	1	counts		.reg
	15	16384	O	Diagnostic Data (main filter A input)	1	counts		.reg
	16	32768	P	board temperature	0.1 degC	degC	2^{10}x0.1degC	.reg
	17	65536	Q	PTC current	0.1 mA	mA	2^{12}x0.1mA	.reg
	18	2^17	R	Diagnostic Data (ex-mot cl_difference)	1	counts		.reg
	19	2^18	S	Diagnostic Data (em-mot cl_difference)	1	counts		.reg
	20	2^19	T	Diagnostic Data (ex-mot cl_y_out)	1	counts		.reg
	21	2^20	U	Diagnostic Data (em-mot cl_y_out)	1	counts		.reg
	22	2^21	V	Diagnostic Data (peak-detector-data)	1	counts		.reg
	23	2^22	W	Diagnostic Data (short allowed)	1	counts		.reg
	24	2^23	X	Diagnostic Data (short allowed)	1	counts		.reg
	25	2^24	Y	Diagnostic Data (short allowed)	1	counts		.reg
	26	2^25	Z	Diagnostic Data (short allowed)	1			.reg
no raw-data output	100			leak status	enum	enum	8	.reg
	101				1	1	1	.reg
	102				1	1	1	.reg
	103				1	1	8	.reg
	104				1	1	8	.reg
	105			NTC voltage	mV	V	2^{14}mV	.reg
	106			PTC voltage	mV	V	2^{14}mV	.reg
	107			0 (future use)	1			.reg
	108			0 (future use)	1			.reg
	109			0 (future use)	1			.reg
	110			0 (future use)	1			.reg

25.8. FLD Diagnose Buffers

Name

CELL (diag_buf)
History of cell changes

Characteristics

max entries	10
max characters	3
storage	none volatile
availability	always
storage frequency	with each detection of a differenct cell type (includes no cell!)

Contents

time, cell type

time	seconds since 1970, the time when a cell was removed or a new cell was put in.
cell type	type of cell: 0 = no cell, 1 = standard cell, 2 = cuvette, 3 = tbd

Description

Every put in or remove of a cell is filed.

Name

MOTR (diag_buf)

number of revolutions for excitation and emisson gratings

Characteristics

max entries	50
max characters	12
storage	none erasable (flash)
availability	always
storage frequency	at every power cycle, time between 2 entries is at least 1 week

Contents

time, revolution_ex, revolution_em

time	seconds since 1970
revolution_ex	number of revolutions of excitation grating
revolution_em	number of revolutions of emission grating

Description

After power cycle of the detector both gratings are kicked on and turn free until they stop. The revolutions are counted and filed.

Name

ENCL (diag_buf)

encoder signal for excitation and emisson gratings

Characteristics

max entries	50
max characters	47
storage	none erasable (flash)
availability	always
storage frequency	at every power cycle, time between 2 entries is at least 1 week

Contents

time,
offset_ex_a, ampl_ex_a, offset_ex_b, ampl_ex_b,
offset_em_a, ampl_em_a, offset_em_b, ampl_em_b

time	seconds since 1970
offset_ex_a	offset of excitation encoder phase A
ampl_ex_a	amplitude of excitation encoder phase A
offset_ex_b	offset of excitation encoder phase B
ampl_ex_b	amplitude of excitation encoder phase B
offset_em_a	offset of emission encoder phase A
ampl_em_a	amplitude of emission encoder phase A
offset_em_b	offset of emission encoder phase B
ampl_em_b	amplitude of emission encoder phase B

Description

After power cycle of the detector the signal of both encoders gratings are measured.

Name

LK (diag_buf)
last ptc and ntc values before leak

Characteristics

max entries	40
max characters	24
storage	none volatile
availability	always
storage frequency	every second until leak occures

Contents

time, ptc, return1, ntc, return2, leak_state

time	seconds since 1970
ptc	value of ptc
return1	return code of adc
ntc	value of ntc
return2	return code of adc
leak_state	current leak state

Description

Internal use only.

Name

RCAL (diag_buf)
result of the wavelength calibration test

Characteristics

max entries	10
max characters	32
storage	none erasable
availability	always
storage frequency	on request: command WCAL

Contents

entry_time, calibration_time, ex_value, em_value

entry_time	seconds since 1970, the time when this entry was made
calibration_time	seconds since 1970, the time when the last calibration was finished
ex_value	the difference in nm between the last calibration and the calibration before for excitation grating
em_value	the difference in nm between the last calibration and the calibration before for emission grating

Description

This buffer stores the differences between 2 consecutive calibration test for both excitation and emission grating. The values stored are in nm. An entry is made by calling the command WCAL.

Name

PREP (diag_buf)
prepare result history

Characteristics

max entries	10
max characters	32
storage	non volatile
availability	always
storage frequency	one entry per request, one entry per analysis

Contents

cosy_list_ no, time, end, tbd1, tbd2

cosy_list_no	number of the PREP cosy list
time	seconds since 1970, the time when the preparation started.
end	type of end: 0 = normal end , 1 = aborted
tbd1	not used so far
tbd2	not used so far

Description

At end of each preparation the values above will be stored to this buffer.

Name

`LMPI (diag_buf)`

result of the lamp intensity test

Characteristics

max entries	50
max characters	23
storage	none erasable
availability	always
storage frequency	on request, time between 2 entries is at least 1 week

Contents

time, cts_250, cts_350, cts_450, cts_600

time	seconds since 1970, the time when the lamp intensity test started
cts_250	reference diode counts at 250nm
cts_350	reference diode counts at 350nm
cts_450	reference diode counts at 450nm
cts_600	reference diode counts at 600nm

Description

The 4 values filed in this buffer are extracted from a lamp intensity scan (stored in buffer STLC (diag_buf) (page 759)).

Name

DARK (diag_buf)
result of dark current test

Characteristics

max entries	50
max characters	12
storage	none erasable
availability	always
storage frequency	on request, time between 2 entries is at least 1 week

Contents

time, reference_diode, photomultiplier

time	seconds since 1970, the time when the dark current started
reference_diode	dark current counts of reference diode
photomultiplier	dark current counts of PhotoMultiplier

Description

Start of dark current test is on user request (command STRS -21 (page 773)).

Name

`LMPF (diag_buf)`

results of flash lamp fluctuation test

Characteristics

max entries	50
max characters	15
storage	non erasable
availability	always
storage frequency	on request, time between 2 entries is at least 1 week

Contents

time, wavelength, average, std_deviation

time	time of entry, in seconds since 1970
wavelength	wavelength at which the test was made
average	average of 20 flashes
std_deviation	standard deviation of the 20 flashes

Description

For this test 20 flashes are measured at specified wavelength through the reference diode. The 20 flashes from which average and standard deviation are calculated are stored in buffer LPFF (diag_buf). The flash lamp fluctuation test is started with command STRS -22 (page 774).

Name

LPFF (diag_buf)

20 intensity values of flash lamp fluctuation test

Characteristics

max entries	21
max characters	5
storage	volatile
availability	always
storage frequency	on request

Contents

cts

cts	intensity values in counts, measured through reference diode, first entry holds wavelength at which the lamp fluctuation test was made.

Description

The flash lamp fluctuation test is started with command STRS -22 (page 774).

Name

STSX (diag_buf)
last excitation scan

Characteristics

max entries	1001
max characters	20
storage	volatile
availability	always
storage frequency	on request: whole buffer for 1 scan

Contents

lu

lu	Luminescence units

Description

Contains the last scan generated by STRS -17. Header of the buffer holds start wavelength and step width. Excitation wavelength range is specified by EXSC <wlfrom>,<wlto>,<wlinc>. Emission wavelength is specified by EM1 parameter of the current method.

Name

STSM (diag_buf)

last emission scan

Characteristics

max entries	1001
max characters	20
storage	volatile
availability	always
storage frequency	on request: whole buffer for 1 scan

Contents

lu

lu	Luminescence units

Description

Contains the last scan generated by STRS -18. Header of the buffer holds start wavelength and step width. Emission wavelength range is specified by EMSC <wlfrom>,<wlto>,<wlinc>. Excitation wavelength is specified by EX1 parameter of the current method.

Name

STLC (diag_buf)
last lamp characteristic scan

Characteristics

max entries	1001
max characters	20
storage	non volatile
availability	always
storage frequency	on request: whole buffer for 1 scan

Contents

cts

cts	counts of reference diode

Description

Contains the last lamp characteristic scan generated by STRS -19 (page 772). Wavelength range is always 200nm .. 1200nm, step width 1nm. The scan is measured with reference diode.

Name

SC3D (diag_buf)
History of 3D scans

Characteristics

max entries	20
max characters	12
storage	non volatile
availability	always
storage frequency	with each start of a 3D scan

Contents

time

time	seconds since 1970

Description

Holds the time of the start of the last 20 3D scans.

Name

SART (diag_buf)

information about last angle scan in rotation mode

Characteristics

max entries	30
max characters	32
storage	non volatile
availability	always
storage frequency	on request: at end of each angle scan in rotation mode

Contents

time, duration, angle start, angle end, step width, bunch

time	start time, seconds since 1970
duration	duration of the scan in msec
angle start	start angle of the scan
angle end	end angle of the scan
step width	step width of the scan
bunch	bunch value

Description

Holds information about the last angle scan in rotation mode. All angle values are measured in units of 0.001 degree.

Name

SANG (diag_buf)

last angle scan in rotation mode

Characteristics

max entries	4501
max characters	10
storage	volatile
availability	always
storage frequency	on request: whole buffer for 1 scan

Contents

cts

cts	counts of reference diode or PhotoMultiplier

Description

Holds the last angle scan in rotation mode generated by SCAN:ANGL 1,...

Name

SAPT (diag_buf)

information about last angle scan in position mode

Characteristics

max entries	30
max characters	32
storage	non volatile
availability	always
storage frequency	on request: at end of each angle scan in position mode

Contents

time, duration, angle start, angle end, step width, bunch

time	start time, seconds since 1970
duration	duration of the scan in msec
angle start	start angle of the scan
angle end	end angle of the scan
step width	step width of the scan
bunch	bunch value

Description

Holds information about the last angle scan in position mode. All angle values are measured in units of 0.001 degree.

Name

SSSD (diag_buf)

last angle scan in position mode

Characteristics

max entries	4501
max characters	10
storage	volatile
availability	always
storage frequency	on request: whole buffer for 1 scan

Contents

cts

cts	counts of reference diode or PhotoMultiplier

Description

Holds the last angle scan in position mode generated by SCAN:ANGL 0,...

Name

CPMT (diag_buf)
History of pmt calculations

Characteristics

max entries	20
max characters	16
storage	non volatile
availability	always
storage frequency	with each end of a pmt calculation

Contents

time, pmt

time	seconds since 1970
pmt	the calculated pmt value

Description

Holds the time and the result of the pmt calculations.

Name

CLCP (diag_buf)

Information of the last pmt calculation

Characteristics

max entries	6
max characters	20
storage	volatile
availability	always
storage frequency	with end of a pmt calculation

Contents

pmt gain, amplitude, signal level

pmt gain	pmt gain of this step of the last pmt calculation
amplitude	amplitude
signal level	signal level

Description

Holds information about each step of the last pmt calculation.

25.9. FLD Diagnose Counters

25.9.1. Lamp

Name

`LMPL (diag_counter)`
Low Power Flashes (# * 1000)

Description

Counts the flashes of the currently installed lamp in low power mode.

Characteristics

counting frequency	1000 flashes in low power mode
storage	none erasable (flash)
history entry	none
EMF-event	none

Name

`LMPH (diag_counter)`
High Power Flashes (# * 1000)

Description

Counts the flashes of the currently installed lamp in high power mode.

Characteristics

counting frequency	1000 flashes in high power mode
storage	none erasable (flash)
history entry	none
EMF-event	none

Name

`F_TI (diag_counter)`
Total Flash lamp on time

Description

Gives the on time in seconds of the currently installed lamp.
When a new lamp is installed the counter F_TI which stores the 'lamp-on-time' of the current lamp must be reset. Before reseting the value will be stored together with the current date code into the associated diagnosis buffer. The buffer now shows the dates of lamp changes and the on-time in seconds of those lamps. If the economy mode is on, the time elapses aprox. 5 times slower.

Characteristics

counting frequency	seconds (of complete lamp on time),
storage	none erasable (flash)
history entry	10
EMF-event	EF 6750 EF_FLD_LAMP_ON

Name

`F_ON (diag_counter)`

Actual flash lamp on time

Description

Gives the time since the last 'switch on' of the lamp

Characteristics

counting frequency	seconds
storage	volatile (ram)
history entry	none
EMF-event	none

Name

`FLLT (diag_counter)`

relative Flash lamp lifetime

Description

Gives the relative lifetime in steps of 0.1% of the currently installed lamp. 100.0% of the Flash lamp lifetime is reached when the lamp intensity is dropped down to approximately 35% of the intensity of a new flash lamp.

When a new lamp is installed the counter FLLT which stores the 'relative lamp lifetime' of the current lamp must be reset. Before reseting the value will be stored together with the current date code into the associated diagnosis buffer. The buffer now shows the dates of lamp changes and the relative lifetime in percent of those lamps.

Characteristics

counting frequency	
storage	none erasable (flash)
history entry	10
EMF-event	EF 6752 EF_FLD_LAMP_LIFETIME

25.9.2. Motors

Name

`XRON (diag_counter)`

Excitation motor 'rotation time'

Description

Gives the total seconds of motor rotation.

Characteristics

counting frequency	seconds
storage	none erasable (flash)
history entry	none
EMF-event	none

Name

`XPON (diag_counter)`

Excitation motor 'position time'

Description

Gives the total seconds of motor position mode.

Characteristics

counting frequency	seconds
storage	none erasable (flash)
history entry	none
EMF-event	none

Name

MRON (diag_counter)

Emission motor 'rotation time'

Description

Gives the total seconds of motor rotation mode.

Characteristics

counting frequency	seconds
storage	none erasable (flash)
history entry	none
EMF-event	none

Name

MPON (diag_counter)

Emission motor 'position time'

Gives the total seconds of motor position mode.

Characteristics

counting frequency	seconds
storage	none erasable (flash)
history entry	none
EMF-event	none

25.9.3. Cell

Name

CEL0 (diag_counter)

time without cell

Description

Before reseting the counter its value will be stored together with the current date code into the associated diagnosis buffer.

Characteristics

counting frequency	seconds
storage	non erasable (flash)
history entry	10
EMF-event	EF 6760 EF_CELL_NO_CELL

Name

CEL1 (diag_counter)

time with standard cell

Description

Before reseting the counter its value will be stored together with the current date code into the associated diagnosis buffer. The buffer now shows the dates of introducing a new cell and the usage time of those cells.

Characteristics

counting frequency	seconds
storage	non erasable (flash)
history entry	10
EMF-event	EF 6761 EF_CELL_STANDARD

Name

CEL2 (diag_counter)
time with cuvette

Description

Before reseting the counter its value will be stored together with the current date code into the associated diagnosis buffer. The buffer now shows the dates of introducing a new cuvette and the usage time of those cuvettes.

Characteristics

counting frequency	seconds
storage	non erasable (flash)
history entry	10
EMF-event	EF 6762 EF_CELL_CUVETTE

Name

`CEL3 (diag_counter)`

time cell type 3

Description

not used so far

Characteristics

counting frequency	seconds
storage	non erasable (flash)
history entry	10
EMF-event	EF 6763 EF_CELL_3

25.9.4. Others

Name

`PMTI (diag_counter)`

Photo multiplier exposure counter

Description

Gives the time in seconds since the last counter reset and the instrument is on. Before reseting the counter its value will be stored together with the current date code into the associated diagnosis buffer.

Characteristics

counting frequency	seconds
storage	none erasable (flash)
history entry	5
EMF-event	EF 6751 EF_FLD_PMT_ON

Name

`SC3D (diag_counter)`

counts each start of a 3d scan

Description

Characteristics

counting frequency	each start
storage	non volatile
history entry	none
EMF-event	none

25.10. FLD Test Functions

Name

STRS -19
start lamp characteristic/lamp intensity test

Synopsis

Instruction: `STRS -19`
Reply: `<rc> STRS -19`

Description

The lamp characteristic/lamp intensity test is implemented as internal cosy list -19. This command starts this special cosy list.
This test takes an excitation scan with a wavelength range from 200nm to 1200nm, step width 1nm. The scan is measured using reference diode. The scan is stored in diagnostig buffer STLC (diag_buf) (page 759). From this scan the values at 250nm, 350nm, 450nm and 600nm are extracted and stored in diagnostic buffer LMPI (diag_buf) (page 753).

Parameters

none

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: start lamp characteristic/lamp intensity test
Instruction: `STRS -19`
Reply: `RA 0000 STRS -19`

Name

STRS -21
start dark current test

Synopsis

Instruction: STRS -21
Reply: <rc> STRS -21

Description

The dark current test is implemented as internal cosy list -21. This command starts this special cosy list.
This test measures the dark current of both reference diode and PhotoMultiplier. The measured values are stored in diagnostig buffer DARK (diag_buf) (page 754).

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: start dark current test
Instruction: STRS -21
Reply: RA 0000 STRS -21

Name

STRS -22

start lamp fluctuation test

Synopsis

Instruction: STRS -22

Reply: <rc> STRS -22

Description

The lamp fluctuation test is implemented as internal cosy list -22. This command starts this special cosy list.

For this test 20 flashes at excitation wavelength which is specified with command LMPF <wl> are measured through reference diode. The result of this test can be ask with query LMPF?. Wavelength, intensity average and standard deviation are stored in diagnostig buffer LMPF (diag_buf) (page 755), the intensity values of the last test are stored in diagnostig buffer LPFF (diag_buf) (page 756).

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: start lamp fluctuation test

Instruction: STRS -22

Reply: RA 0000 STRS -22

Name

LMPF !!NOT IMPLEMENTED!!
set wavelength for next Lamp Fluctuation Test

Synopsis

Instruction:	`LMPF <Wavelenght>`
Reply:	`<rc> LMPF <wl>`
Instruction:	`LMPF?`
Reply:	`<rc> LMPF <wl>,<int_av>,<std_dev>`

Description

With the instruction LMPF <wl> the wavelength for the next lamp fluctuation test is specified. The query LMPF? returns the result of the last lamp fluctuation test. The results of the last 50 tests are stored in diagnostic buffer LMPF (diag_buf) (page 755). The intensity values of the 20 flashes of the last test are stored in LPFF (diag_buf) (page 756).
With LMPF? the result of last Lamp Fluctuation Test can be query.

Parameters

<wl>	wavelenght (nm)
<int_av>	intensity average (cts)
<std_dev>	standard deviation

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set wavelength for next Lamp Fluctuation test to 250 nm
Instruction: `LMPF 250`
Reply: `RA 0000 LMPF 250`

Name

SCAN:ANGL
Take an angle scan

Synopsis

Instruction: `SCAN:ANGL <PosRot>,<ExEm>,<From>,<To>,<Inc>,<Bunch>,<Mode>`
Reply: `<rc> SCAN:ANGL <PosRot>,<ExEm>,<From>,<To>,<Inc>,<Bunch>,<Mode>`

Description

With the instruction SCAN:ANGL an angle scan could be taken. The last scan taken in position mode is stored in diagnostic buffer SSSD (diag_buf) (page 764), information about this scan is stored in SAPT (diag_buf) (page 763), in rotation mode the last scan is stored in SANG (diag_buf) (page 762), information to the scan is stored in SART (diag_buf) (page 761). The diagnostic buffers SSSD and SANG can only store up to 4501 points, if a scan has more than 4501 points the begin of the scan will be overwritten. To take an angle scan in rotation mode the correct rotation mode (MMOD 1 = excitation rotation mode; MMOD 2 = emission rotation mode) has to be preset.
Note: zero point of encoder differs from module to module, it has no fixed position relative to vertex of grating.

Parameters

<PosRot>	0 = position mode; 1 = rotation mode
<ExEm>	0 = excitation scan; 1 = emission scan
<From>	0..360000; angle relative to zero of encoder, step width 0.001 deg
<To>	0..360000; angle relative to zero of encoder, step width 0.001 deg
<Inc>	1..1000; step width 0.001 deg
<Bunch>	1..1000, bunch count
<Mode>	0 = reference diode; 1 = PhotoMultiplier

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: full circle excitation scan, step 0.1deg, measured with reference diode
Instruction: `SCAN:ANGL 1,0,0,360000,100,1,0`
Reply: `RA 0000 SCAN:ANGL 1,0,0,360000,100,1,0`

26. Refractive Index Detector (G1362A)

26.1. RID Method Parameter

Name

TEMP
Optical Unit Temperature

Synopsis

Instruction:	`TEMP <temperature>`
Reply:	`<rc> TEMP <temperature>`
Instruction:	`TEMP?`
Reply:	`<rc> TEMP <temperature>`
Instruction:	`ACT:TEMP?`
Reply:	`<rc> ACT:TEMP <temperature>`

Description

The TEMP parameter defines the temperature of the optical unit to be controlled if the heat control is turned on with the HEAT 1 instruction. The heat control is automatically turned on after power un, when the system paramter HON is set to 1.

Parameters

<temperature> -274.00, 5.00 ... 55.00; -274 for off; units are degrees Celsius

Default Values

35 'C

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Set temperature to 45 'C
	Instruction:	`TEMP 45`
	Reply:	`RA 0000 TEMP 45.00`
(b)	Comment:	Query the temperature setpoint
	Instruction:	`TEMP?`
	Reply:	`RA 0000 TEMP 45.00`
(c)	Comment:	Query the actual temperature
	Instruction:	`ACT:TEMP?`
	Reply:	`RA 0000 ACT:TEMP 37.65`

Name

AZA

Automatic Zero during Analysis

Synopsis

Instruction:	`AZA <on/off>`
Reply:	`<rc>  <on/off>`
Instruction:	`AZA?`
Reply:	`<rc> AZA <on/off>`

Description

The AZA instruction defines if the RI signal is automatically set to zero as part of the analysis. If set to 'on', the RI value is set to zero either due to a remote prepare puls or due to the instruction AZOT that is part of the default cosy list. The AZOT instruction is placed before the injection.

Parameters

<on/off>	0 for off; 1 for on = automatic zero

Default Values

<on/off> on

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Set automatic zero to off
	Instruction:	`AZA 0`
	Reply:	`RA 0000 AZA 0`
(b)	Comment:	Query the automatic zero setpoint
	Instruction:	`AZA?`
	Reply:	`RA 0000 AZA 0`

Name

POL

Polarity Value

Synopsis

Instruction:	`POL <not reverted/reverted>`
Reply:	`<rc> POL <not reverted/reverted>`
Instruction:	`POL?`
Reply:	`<rc> POL <not reverted/reverted>`
Instruction:	`ACT:POL?`
Reply:	`<rc> ACT:POL <not reverted/reverted>`
Instruction:	`AT:POL <time>, <not reverted/reverted>`
Reply:	`<rc> AT:POL <time>, <not reverted/reverted>`

Description

The POL instruction defines the polarity of the RI difference calculation.
The ACT:POL? query returns the actual polarity setting.

Parameters

<not reverted/reverted>	0 for not reverted; 1 for reverted
<time>	0 .. 99999.00 [min]

Default Values

<not reverted/reverte> not reverted

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Set polarity to reverted
	Instruction:	`POL 1`
	Reply:	`RA 0000 POL 1`
(b)	Comment:	Query polarity setpoint
	Instruction:	`POL?`
	Reply:	`RA 0000 POL 1`
(c)	Comment:	Query for actual polarity
	Instruction:	`ACT:POL?`
	Reply:	`RA 0000 ACT:POL 1`
(d)	Comment:	Define time table entry for a polarity setting
	Instruction:	`AT:POL 5,1`
	Reply:	`RA 0000 AT:POL 5.00, 1`

Name

TPRG
Time for Automatic Purge

Synopsis

Instruction:	`TPRG <purge time>, <wait after purge>`
Reply:	`<rc> TPRG <time>, <wait after purge>`
Instruction:	`TPRG?`
Reply:	`<rc> TPRG <time>, <wait after purge>`

Description

The TPRG instruction defines the duration for switching the purge valve during the analysis. If the value is zero, no purge will be done. If greater 0, the purge valve is switched to direct the mobile phase through both cell compartements for the given time. During the purge the nrdy bit ’purge’ is set. The purge cycle is part of the default COSY list, the command for starting the purge is APRG.For the interactive purge the instruction VREF <state>,<time> is available, to query the actual purge time, the instruction ACT:VREF? ==> .... VREF <state>,<actual time> may be used.
The second parameter defines the length of a period following the end of the purge. During this time the nrdy bit ’purge wait’ is set.

Parameters

<purge time>	0 .. 99999.00
<wait after purge>	0 .. 99999.00

Default Values

<purge time>	0
<wait after purge>	0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set a purge time of 10 minutes, with 2 minutes to wait after purge
	Instruction:	`TPRG 10, 2`
	Reply:	`RA 0000 TPRG 10.00, 2`
(b)	Comment:	Query the setpoint
	Instruction:	`TPRG?`
	Reply:	`RA 0000 TPRG 10.00, 2.00`

Name

AZE1

Set Zero for the Analog Output

Synopsis

Instruction:	`AZE1 <value>`
Reply:	`<rc> AZE1 <value>`
Instruction:	`AZE1?`
Reply:	`<rc> AZE1 <value>`

Description

The AZE1 instruction sets and queries the zero value for the analog output

Parameters

<value>	1 .. 99 percent of full scale

<Default Values

5

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set analog zero to 100mv (10% of full scale)
	Instruction:	`AZE1 10`
	Reply:	`RA 0000 AZE1 10`
(b)	Comment:	Query analog zero setpoint
	Instruction:	`AZE1?`
	Reply:	`RA 0000 AZE1 10`

Name

AAT1
Analog Attenuation

Synopsis

Instruction:	`AAT1 <value>`
Reply:	`<rc> AAT1 <value>`
Instruction:	`AAT1?`
Reply:	`<rc> AAT1 <value>`

Description

The full range output is calculated as $2^{\langle value\rangle-10}$ times full nRIU range.

Parameters

<value>	0 .. 11;

Default Values

10 , full analog output (0.1V or 1V depending on the ASC instruction) equals 512000 nRIU.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set attenuation to 1 (double the range)
	Instruction:	`AAT1 1`
	Reply:	`RA 0000 AAT1 1`
(b)	Comment:	Query the attenuation setpoint
	Instruction:	`AAT1?`
	Reply:	`RA 0000 AAT1 1`

Name

VCON

Recycle after Analysis

Synopsis

Instruction:	`VCON <value>`
Reply:	`<rc> VCON <value>`
Instruction:	`VCON?`
Reply:	`<rc> VCON <value>`

Description

The VCON instruction allows to automatically switch the recycle valve to the 'reservoir' position after the analysis is done. With the begin of the analysis, the valve is switched back to direct the mobile phase to the 'waste' exit.

Parameters

<value>	0 for no automatic recycle; 1 for automatic recycle

Default Values

0 (no automatic recycle)

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set automatic recycle
	Instruction:	`VCON 1`
	Reply:	`RA 0000 VCON 1`
(b)	Comment:	Query the automatic recycle setpoint
	Instruction:	`VCON?`
	Reply:	`RA 0000 VCON 1`

Name

PKWD

Sets peakwidth of smallest peak expected

Synopsis

Instruction: PKWD <peakwidthnum>
Reply: <rc> PKWD <peakwidthnum>
Instruction: PKWD?
Reply: <rc> PKWD <peakwidthnum>
Instruction: ACT:PKWD?
Reply: <rc> ACT:PKWD <peakwidthnum>
Instruction: AT:PKWD <time>, <peakwidthnum>
Reply: <rc> AT:PKWD <time>, <peakwidthnum>
Instruction: AT:PKWD? <time>
Reply: <rc> AT:PKWD <time>, <peakwidthnum>

Description

The PKWD setpoint is the peakwidth at half hight of the smallest peak expected, it controls the signal filtering.

Parameters

<peakwidthnum> 0 .. 7

<table>
<tr><th>PKWD</th><th>peakwidth
at half hight</th><th>rise time
[10 .. 90 %]</th><th>signal
data rate</th><th>signal
slicewidth</th></tr>
<tr><td>0</td><td><0.005 min</td><td><0.12 sec</td><td rowspan="8">see
DRAT</td><td rowspan="8">see
DRAT</td></tr>
<tr><td>1</td><td>0.005min</td><td>0.12 sec</td></tr>
<tr><td>2</td><td>0.01min</td><td>0.25 sec</td></tr>
<tr><td>3</td><td>0.025 min</td><td>0.5 sec</td></tr>
<tr><td>4</td><td>0.05 min</td><td>1 sec</td></tr>
<tr><td>5</td><td>0.1 min</td><td>2 sec</td></tr>
<tr><td>6</td><td>0.2 min</td><td>4 sec</td></tr>
<tr><td>7</td><td>0.4 min</td><td>8 sec</td></tr>
</table>

Default Values

6 (risetime: 4 sec)

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: set peakwidth parameter
Instruction: PKWD 3
Reply: RA 0000 PKWD 3

Name

DRAT

Set the output data rate of the detector.

Synopsis

Instruction:	`DRAT <data rate>`
Reply:	`<rc> DRAT <data rate>`
Instruction:	`DRAT?`
Reply:	`<rc> DRAT <data rate>`

Description

This instruction set the data output rate of the detector for the rawdata CU. This function often is used together with the PKWD command.

If timetable entry of PKWD is used the datarate should be combined to smallest entry of PKWD. If no timetable entry is used <data rate> can be the same value as peakwidthnum (see PKWD instruction).

Parameters

`<data_rate_num>` independent from flash frequence:

0 :36.50 Hz datarate =	27.4 msec signal slicewidth
1 :36.50 Hz datarate =	27.4 msec signal slicewidth
2 :36.50 Hz datarate =	27.4 msec signal slicewidth
3 :18.25 Hz datarate =	54.8 msec signal slicewidth
4 :9.12 Hz datarate =	109.6 msec signal slicewidth
5 :4.56 Hz datarate =	219.2 msec signal slicewidth
6 :2.28 Hz datarate =	438.4 msec signal slicewidth
7 :1.14 Hz datarate =	876.8 msec signal slicewidth

Default Values

6

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set data rate to 9.12Hz

Instruction: `DRAT 4`

Reply: `RA 0000 DRAT 4`

(b) Comment: try to set data rate setting during a run

Instruction: `DRAT 3`

Reply: `RE 0305 DRAT 3`

26.2. RID System Parameter

Name

HON

Turn on Temperature Control after Power On

Synopsis

Instruction:	`HON <value>`
Reply:	`<rc> HON <value>`
Instruction:	`HON?`
Reply:	`<rc> HON <value>`

Description

The HON paramter allows to automatically turn on the heat control after power is applied.

Parameters

<value>	0 for no automatic turn on; 1 for automatic turn on

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set automatic turn on
	Instruction:	`HON 1`
	Reply:	`RA 0000 HON 1`
(b)	Comment:	Query the setpoint
	Instruction:	`HON?`
	Reply:	`RA 0000 HON 1`

Name

ASC1
Set Analog Scale

Synopsis

Instruction:	`ASC1 <value>`
Reply:	`<rc> ASC1 <value>`
Instruction:	`ASC1?`
Reply:	`<rc> ASC1 <value>`

Description

The ASC1 instruction allows to change the analog output range to either 0.1V full scale or 1V full scale.

Parameters

<value> 0 for 0.1V; 1 for 1V

Default Values

1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Set analog scal to 0.1V
	Instruction:	`ASC1 0`
	Reply:	`RA 0000 ASC1 0`
(b)	Comment:	Query the analog scale setting
	Instruction:	`ASC1?`
	Reply:	`RA 0000 ASC1 0`

Name

RAWS

Select Signals for the Rawdata CU

Synopsis

Instruction:	`RAWS <signalset>`
Reply:	`<rc> RAWS <signalset>`
Instruction:	`RAWS?`
Reply:	`<rc> RAWS <signalset>`

Description

A variety of different signals is continously generated. A part of these signals can be selected by the RAWS instruction.

Although the RAWS setting is always accepted, it becomes active only at the time when the rawdata storing begins. Storing signals starts when the RAWD:STRT instruction is accepted or when an analytical run starts.

Parameters

`<signalset>`	0 .. 2^25
	1 (bit 0 set) means: store signal 1
	2 (bit 1 set) means: store signal 2
	4 (bit 2 set) means: store signal 3
	8 (bit 3 set) means: store signal 4
	... etc.
	any combination is allowed,
	0 means: no signal will be stored (OFF)
	11 means: signal type 1, 2 and type 4 (1+2+8 = 11) will be stored.

Default Values

1

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	select signal type 1 and 2 to be stored
	Instruction:	`RAWS 3`
	Reply:	`RA 0000 RAWS 3`
(b)	Comment:	query signal select setting
	Instruction:	`RAWS?`
	Reply:	`RA 0000 RAWS 3`

26.3. RID Control Commands

Name

HEAT
Turn heat control on or off
ACT:HEAT?
Query actual heat control state

Synopsis

Instruction:	`HEAT <on/off>`
Reply:	`<rc> HEAT <on/off>`
Instruction:	`ACT:HEAT?`
Reply:	`<rc> ACT:HEAT <on/off>`

Description

The HEAT instruction changes the heat control state to either on or off. After power on the heat control may automatically be turned on when the sysem variable HON is set to 1. After turning on the heat control, the control loop tends to reach and keep the temperature setpoint as defined by the TEMP <temperature> instruction.

Parameters

`<on/off>`	`0 for Off, 1 for On`

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Turn on heat control	
	Instruction:	`HEAT 1`	
	Reply:	`RA 0000 HEAT 1`	
(b)	Comment:	Query the actual heat control state	
	Instruction:	`ACT:HEAT?`	
	Reply:	`RA 0000 HEAT 1`	

Name

AZOT
trigger automatic zero

Synopsis

Instruction: AZOT
Reply: <rc> AZOT

Description

With the AZOT instruction the automatic zero during the analysis is done, if the parameter AZA is set to 1. This instruction if part of the default cosy list, right before the injection.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Start automatic purge
Instruction: AZOT
Reply: RA 0000 AZOT

Name

APRG

start automatic purge

Synopsis

Instruction: APRG

Reply: <rc> APRG

Description

With the APRG instruction the automatic purge is started. The appropriate valve for flushing both reference and sample compartement of the cell is switch accordingly. The duration of the purge cycle is determined by the instruction TPRG <time>,<waittime>. During each analysis the APRG instruction is sent by the default cosy list, as part of the detectors prepare cycle. The nrdy bit for 'reference valve switch time in progress' is set. A cosy sync is sent as soon as the switching time has elapsed.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Start automatic purge

Instruction: APRG

Reply: RA 0000 APRG

Name

WPRG

start wait time after automatic purge

Synopsis

Instruction: WPRG

Reply: <rc> WPRG

Description

With the WPRG instruction the wait time after the automatic purge is started. The duration of the wait is determined by the instruction TPRG <time>,<waittime>. During each analysis the WPRG instruction is sent by the default cosy list, as part of the detectors prepare cycle. The nrdy bit for 'wait after reference valve switch' is set. A cosy sync is sent as soon as the wait time has elapsed.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Start wait after automatic purge

Instruction: WPRG

Reply: RA 0000 WPRG

Name

NRIU

calibrate the actual ADC counts to nRIU

Synopsis

Instruction: `NRIU <nano RI units>`

Reply: `<rc> NRIU <nano RI unints>`

Description

The NRIU instruction recalculates the relationship between the ADC counts and the nRIU's. Typically a calibration starts with purging the cell, then setting the actual nRIUs to zero using the ZOUT instruction, then filling the sample compartement of the cell with a compound of a known refrective index difference to the liquid in the reference compartement, then the NRIU instruction is used to initate the calibration/recalculation.

Parameters

`<nano RI units> 50000 .. 700000`

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: Calibrate to a water/sugar standard

Instruction: `NRIU 512000`

Reply: `RA 0000 NRIU 512000`

Name

VREF
switch valve for reference cell

Synopsis

Instruction:	`VREF <on/off>`
Reply:	`<rc> VREF <on/off>`
Instruction:	`VREF <on/off>,<time>`
Reply:	`<rc> VREF <on/off>,<time>`
Instruction:	`ACT:VREF?`
Reply:	`<rc> ACT:VREF <on/off>,<time since>`

Description

The VREF instruction allows to switch the valve connected to the capillaries of the cell compartements. VREF <on/off> activates or deactivates the valve. When activated, the mobile phase passes through both cell compartements.
The VREF <on/off>,<time> instruction allows to specify the time for the switching. While the given time is not yet reached, the nrdy bit for 'reference valve time' is set.
A cosy sync is sent as soon as the time has elapsed or a new VREF instruction is sent.

Parameters

<on/off>	`0 for off ==  electrically not activated,` `1 for on == electrically activated`
<time>	0 .. 99999.00 minutes
<time since>	0 .. any signed number of 32 bit

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Purge for 5 minutes	
	Instruction:	`VREF 1,5`	
	Reply:	`RA 0000 VREF 1,5.00`	
(b)	Comment:	Query actual reference valve state	
	Instruction:	`ACT:VREF>`	
	Reply:	`RA 0000 VREF 1,0.32`	`! valve is on since 0.32 minutes`

Name

VREC

switch valve for recycling

Synopsis

Instruction:	`VREC <on/off>`
Reply:	`<rc> VREC <on/off>`
Instruction:	`VREC <on/off>,<time>`
Reply:	`<rc> VREC <on/off>,<time>`
Instruction:	`ACT:VREC?`
Reply:	`<rc> ACT:VREC <on/off>,<time since>`

Description

The VREC instruction allows to switch the valve connected to the capillaries of the optical unit exit. VREC <on/off> activates or deactivates the valve. When activated, the mobile phase is directed to the solvent reservoir.

The VREC <on/off>,<time> instruction allows to specify the time for the switching. While the given time is not yet reached, the nrdy bit for 'recycle valve time' is set.

A cosy sync is sent as soon as the time has elapsed or a new VREC instruction is sent.

Whit the instruction VCON 1 (method parameter!) the recycle valve is automatically switched to recycle after each analysis (and at once if there is no analysis in progress), and to waste with each begin of the analysis.

Parameters

<on/off>	0 for off == electrically not activated, 1 for on == electrically activated
<time>	0 .. 99999.00 minutes
<time since>	0 .. any signed number of 32 bit

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Direct flow path to the solvent reservoir
Instruction: `VREC 1`
Reply: `RA 0000 VREC 1`

(b) Comment: Query actual recycle valve state
Instruction: `ACT:VREC?`
Reply: `RA 0000 VREC 1,300.5 ! recycling since 300.5 minutes`

Name

ZOUT
Zero the RI-Signal

Synopsis

Instruction: ZOUT
Reply: <rc> ZOUT

Description

The ZOUT instruction keeps the actual RI value as given offset. The next RI value is zero.
This instruction is part of the default cosy list, it is distributed after the LADJ instruction as part of the detector's prepare activities.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Set RI signal to 0
Instruction: ZOUT
Reply: RA 0000 ZOUT

Name

ATE

Start/Stop the analog test

Synopsis

Instruction:	`ATE <on/off>`
Reply:	`<rc> ATE <on/off>`
Instruction:	`ATE?`
Reply:	`<rc> ATE <on/off>`

Description

The ATE instruction starts and stops the analog output test. During this test a square outputsignal of approx. 10 microVolts with 24 seconds of period is generated instead of the RI signal.

Parameters

`<on/off>` `0 for stop; 1 for start`

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: Start the analog output test

Instruction: `AZE 1`

Reply: `RA 0000 AZE 1`

Name

SIMU:DFLT

provides simulated data with the next analysis

Synopsis

Instruction: SIMU:DFLT

Reply: <rc> SIMU:DFLT

Description

The SIMU:DFLT instruction sets a flag that with the next analysis, simulated data will be generated instead of normal run data.

Parameters

none

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Prepare for simulated data

Instruction: SIMU:DFLT

Reply: RA 0000 SIMU:DFLT

26.4. RID Actuals

Name

`ACT:RI?`

Actual Value of the RI Signal

Synopsis

Instruction: `ACT:RI?`

Reply: `<rc> ACT:RI? <nRIUs>`

Description

The ACT:RI? instruction queries for the last measured RI value.

Parameters

<nRIUs> -700000.00 ... +700000.00 nRIU

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual RI value

Instruction: `ACT:RI?`

Reply: `RA 0000 ACT:RI 826.41`

Name

ACT:TEMP?
Actual Optical Unit Temperature

Synopsis

Instruction: `ACT:TEMP?`
Reply: `<rc> ACT:TEMP? <temperature>`

Description

The ACT:TEMP? instruction queries for the actual flow value.

Parameters

<temperature> -5.00 .. 140.00

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query actual temperature
Instruction: `ACT:TEMP?`
Reply: `RA 0000 ACT:TEMP 44.36`

Name

ACT:ITMP?
Actual Inlet Capillary Temperature

Synopsis

Instruction: `ACT:ITMP?`
Reply: `<rc> ACT:ITMP? <temperature>`

Description

The ACT:ITMP? instruction queries for the temperature seen at the inlet capillary sensor.

Parameters

<Temperature> -40.00 .. +140.00

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual inlet temperature
Instruction: `ACT:ITMP?`
Reply: `RA 0000 ACT:ITMP 44.67`

Name

`ACT:DIOD?`
Actual Balance of the Diodes

Synopsis

Instruction: `ACT:DIOD?`
Reply: `<rc> ACT:DIOD? <ratio>`

Description

The ACT:DIOD? instruction queries for the actual balance of the diodes, calculated as difference of the signals devided by the sum of the signals.

Parameters

<ratio> -1.000 ... +1.000

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual balance of the diodes
Instruction: `ACT:DIOD?`
Reply: `RA 0000 ACT:DIOD 0.003`

Name

ACT:D1?
Actual ADC counts of diode 1

Synopsis

Instruction: ACT:D1?
Reply: <rc> ACT:D1 <ADC counts>

Description

The ACT:D1? instruction queries for the actual counts delivered from the ADC for the diode. If this value is near the lower or upper limit of the value range, no valid RI signal may be available.

Parameters

<ADC counts> 0 .. 64000

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual ADC counts of Diode 1
Instruction: ACT:D1?
Reply: RA 0000 ACT:D1 32000

Name

`ACT:D2?`

Actual ADC counts of diode 2

Synopsis

Instruction: `ACT:D2?`

Reply: `<rc> ACT:D2 <ADC counts>`

Description

The ACT:D1? instruction queries for the actual counts delivered from the ADC for the diode. If this value is near the lower or upper limit of the value range, no valid RI signal may be available.

Parameters

<ADC counts> 0 .. 64000

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual ADC counts of Diode 2

Instruction: `ACT:D2?`

Reply: `RA 0000 ACT:D2 32000`

Name

ACT:HEAT?
Actual state of the heater

Synopsis

Instruction: `ACT:HEAT?`
Reply: `<rc> ACT:HEAT <on/off>`

Description

The ACT:HEAT? instruction queries for the actual control state of the heater.

Parameters

<on/off>	0 for off, 1 for heat control on

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual heater state
Instruction: `ACT:HEAT?`
Reply: `RA 0000 ACT:HEAT 1`

Name

ACT:POL?
Actual Polarity

Synopsis

Instruction: `ACT:POL?`
Reply: `<rc> ACT:POL <not reverted/reverted>`

Description

The ACT:POL? instruction queries for the actual polarity setting.

Parameters

<not reverted/reverted> 0 for not reverted, 1 for reverted

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual polarity
Instruction: `ACT:POL?`
Reply: `RA 0000 ACT:POL 0`

Name

ACT:PKWD?

Actual peak width

Synopsis

Instruction: `ACT:PKWD?`

Reply: `<rc> ACT:PKWD <value>`

Description

The ACT:PKWD? instruction queries for the actual setting for the peak width. See also the method parameter PKWD

Parameters

<value> 0 .. 7

Default Values

6

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual peakwidth

Instruction: `ACT:PKWD?`

Reply: `RA 0000 ACT:PKWD 6`

Name

ACT:CVAL?

Actual Calibration limits

Synopsis

Instruction: `ACT:CVAL?`

Reply: `<rc> ACT:CVAL <nRIU>,<upper limit>,<lower limit>`

Description

The ACT:CVAL? instruction queries for the calibration limits appropriate for the actual RI value.

Parameters

<nRIU>	-1000000 .. +1000000
<lower limit>	<nRIU> * 0.8
<upper limit>	<nRIU> * 1.2

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RE 2201 RI value too low for calibration, if nRIU < 50000

Examples

(a) Comment: query the actual calibration limits

Instruction: `ACT:CVAL?`

Reply: `RA 0000 ACT:CVAL 500000,600000,400000`

Name

ACT:VREF?
Actual state ofthe reference valve

Synopsis

Instruction:	`ACT:VREF?`
Reply:	`<rc> ACT:VREF <on/off>, <time>`

Description

The ACT:VREF? instruction queries for the actual state of the reference valve. See also the control parameter VREF. If the reference valve is activated, both cell compartements are flushed in series.

Parameters

<on/off>	0 for off(not activated), 1 for on (activated)
<time>	0..99999.00

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual peakwidth
Instruction: `ACT:VREF?`
Reply: `RA 0000 ACT:VREF 1,5.3`

Name

ACT:VREC?

Actual state of the recycle valve

Synopsis

Instruction: `ACT:VREC?`

Reply: `<rc> ACT:VREC <on/off>, <time>`

Description

The ACT:VREC? instruction queries for the actual state of the recycle valve. See also the method parameter VREC. If the recycle valve is activated, the flow is directed back to the solvent reservoir exit of the RI detector.

Parameters

<on/off>	0 for off(not activated), 1 for on (activated)
<time>	0..99999.00

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual peakwidth

Instruction: `ACT:VREC?`

Reply: `RA 0000 ACT:VREC 1,6.3`

Name

`ACT:WPRG?`

Actual time of wait after automatic purge

Synopsis

Instruction: `ACT:WPRG?`
Reply: `<rc> ACT:WPRG <time>`

Description

The ACT:WPRG? instruction queries for the actual time of the wait specified after automatic purge.

Parameters

<time>	0 .. 99999.00

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual time of the wait after purge
Instruction: `ACT:WPRG?`
Reply: `RA 0000 ACT:WPRG 6.4`

Name

`ACT:WTMP?`

Actual time of the 'wait for temperature' function

Synopsis

Instruction: `ACT:WTMP?`

Reply: `<rc> ACT:WTMP <time>`

Description

The ACT:WTMP? instruction queries for the actual time elapsed since the function 'wait for temperature' has been started. See also the WTMP instruction. If the time reaches the timeout value of the WTMP instruction, the error event EE 2696 is generated.

Parameters

<time> 0..99999.00

Default Values

none

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query the actual time

Instruction: `ACT:WTMP?`

Reply: `RA 0000 ACT:WTMP 6.54`

Name

ACT:WRI?

Actual time of the 'wait for nRIU' function

Synopsis

Instruction: ACT:WRI?

Reply: <rc> ACT:WRI <value>

Description

The ACT:WRI? instruction queries for the actual time since the 'wait for nRIU' function has been started. See also the instruction WRI. If the value returned reaches the timeout value given with WRI, the error event EE 2696 is generated.

Parameters

<value> 0 .. 99999.00

Default Values

6

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query the actual time

Instruction: ACT:WRI?

Reply: RA 0000 ACT:WRI 6.78

26.5. RID Rawdata File Description

The Agilent 1200 RI Detector generates 6 signals. The rawdata file can store these signals as records RA, RB, RC, RD, RE and RG.
To select, which signal should be stored, see instruction RAWS on page 258.
For selection of the data representation (decimal, hexadecimal or decimal) see instruction RAWF on page 259.
The values are all represented in signed 32, the Resolution gives the factor to convert the integer to the unit.

signal id	Description	Unit	Resolution
RA	RI Signal	nRIU	0.01
RB	Optical Unit Temperature	degrees Celsius	0.01
RC	Diode Balance	none	0.00001
RD	Polarity	none	1
RE	Diode1	Counts	1
RF	Diode 2	Counts	1

26.6. RID Diagnose Buffers

Name

VREF (diag)

History of purge valve switches

Characteristics

max entries	20
max characters	40
storage	battery buffered
availability	always
entry	once per VREF 1,<time> instruction, or auto purge

Contents

Time, Bits before Pulse, Bits during Pulse, Bits after Pulse

Time	Seconds since 1970
Bits	Voltage level at comparators, 0 for deactivation entries

Description

This buffer keeps the history of the last purge valve switches. one entry for activation and one entry for deactivation

Name

RFHS (diag)

History of the purge valve switches

Characteristics

max entries	30
max characters	20
storage	battery buffered
availability	always
entry	once per VREF instruction

Contents

Time, OnOff

Time	Seconds since 1970
OnOff	1 for On, 0 for Off

Description

This buffer keeps the times and states the purge valve was switched to.

Name

VREC (diag)

History of recycle valve switches

Characteristics

max entries	20
max characters	40
storage	battery buffered
availability	always
entry	once per VREC 1,<time> instruction, or auto recycle

Contents

Time, Bits before Pulse, Bits during Pulse, Bits after Pulse

Time	Seconds since 1970
Bits	Voltage level at comparators, or zero for deactivation entries

Description

This buffer keeps the history of the last recycles valve switches. One entry for activation and one entry for deactivation.

Name

RCHS (diag)

History of the recycle valve switches

Characteristics

max entries	30
max characters	20
storage	battery buffered
availability	always
entry	once per VREC instruction

Contents

Time, OnOff

Time	Seconds since 1970
OnOff	1 for On, 0 for Off

Description

This buffer keeps the times and states the recycle valve was switched to.

Name

VTRF (diag)
Purge History

Characteristics

max entries	20
max characters	40
storage	battery buffered
availability	always
entry	once per VREF 0,<time> instruction

Contents

Time, Duration

Time	Seconds since 1970
Duration	The duration the purge valve was activated in seconds

Description

This buffer keeps the history of the purge cycles.

Name

VTRC (diag)
Recycle History

Characteristics

max entries	20
max characters	20
storage	battery buffered
availability	always
entry	once per VREC 0,<time> instruction

Contents

Time, Duration

Time	Seconds since 1970, time when deactivated
Duration	The duration the recycle valve was activated in seconds

Description

This buffer keeps the history of the recycle periods.

Name

CTPS (diag)

Temperature History (seconds) of inlet sensor

Characteristics

max entries	50
max characters	10
storage	volatile
availability	always
entry	once per second

Contents

temperature

temperature	the actual temperature with 0.01'C resolution

Description

This buffer keeps the shortterm temperature history.

Name

`CTPM (diag)`

Temperature History (minutes) of inlet temperature

Characteristics

max entries	30
max characters	10
storage	volatile
availability	always
entry	once per minute

Contents

temperature

temperature	the actual temperature with 0.01'C resolution

Description

This buffer keeps the longterm temperature history.

Name

RIDD (diag)
RI Data History

Characteristics

max entries	50
max characters	10
storage	volatile
availability	always
entry	once per data point (after pre bunch)

Contents

RI Value

RI Value	One Value each 27.4 ms. Not recalculated to nRIU.

Description

This buffer keeps a short history of the RI values.

Name

DADC (diag)
RI Raw Data History

Characteristics

max entries	100
max characters	10
storage	volatile
availability	always
entry	once per data point (directly from ADC)

Contents

RI Value

RI Value	One Value each 4.5 ms. Not filtered, not adjusted to nRIU

Description

This buffer keeps a short history of the raw RI values.

Name

HEAT (diag)

History of Heat Control PWM

Characteristics

max entries	30
max characters	10
storage	volatile
availability	always
entry	once per 0.1 seconds

Contents

PWM

PWM	The actual high cycle of the heat control pwm.

Description

This buffer keeps a short history of the heat control logic.

Name

CHNG (diag)

History of Temperature Setpoint Changes

Characteristics

max entries	10
max characters	50
storage	battery buffered
availability	always
entry	once per temperature setpoint change

Contents

start time, start temp, goal temp, time to reach, ticks needed

start time	Seconds since 1970.
start temp	Actual temperature when setpoint was set
goal temp	The new temperature setpoint
time to reach	Time for the temperature slope in 0.1 seconds units
ticks needed	Total time needed to change the setpoint

Description

This buffer keeps a history of the temperature setpoint changes.

Name

OTPS (diag)

History of the Optical Unit Temperature (seconds)

Characteristics

max entries	150
max characters	10
storage	volatile
availability	always
entry	once per second

Contents

Temperature

temperature	in units of 0.01 degrees Celsius.

Description

This buffer keeps a history of the optical unit's temperature.

Name

OTPM (diag)

History of the Optical Unit Temperature (minutes)

Characteristics

max entries	150
max characters	10
storage	volatile
availability	always
entry	once per minute

Contents

Temperature

temperature	in units of 0.01 degrees Celsius.

Description

This buffer keeps a history of the optical unit's temperature.

Name

CAL (diag)

History of the Calibrations

Characteristics

max entries	20
max characters	50
storage	none erasible
availability	always
entry	once per calibration (per NRIU instruction)

Contents

Time, nRIUs, CalibFactor, ADCCounts, Offset

time	Seconds since 1970
nRIU	The number given with the NRIU instruction
CalibFactor	The relation nRIUs <==> ADCCounts
ADCCounts	The ADC counts when the NRIU instruction was received
Offset	The offset kept with the last ZOUT instruction

Description

This buffer keeps a history of the calibrations.

Name

`LVLT (diag)`

History of the Lamp Voltage

Characteristics

max entries	100
max characters	10
storage	volatile
availability	always
entry	once per second

Contents

Voltage

voltage	Lamp voltage in mV

Description

This buffer keeps a history of the lamp voltage.

Name

HVLT (diag)

History of the Heater Voltage

Characteristics

max entries	100
max characters	10
storage	volatile
availability	always
entry	once per second

Contents

Voltage

voltage	Heater voltage in mV

Description

This buffer keeps a history of the heater voltage.

Name

LK (diag)

History of the Leak verifications

Characteristics

max entries	20
max characters	15
storage	volatile
availability	always
entry	once per second

Contents

NTC value, PTC value

NTC , PTC value	ADC Counts from the NTC and PTC multiplexer channel

Description

This buffer keeps a history of the values that have been used to detect a leak.

26.7. RID Diagnose Counters

Name

CALT (diag)
Lamp On Time
Increments by 1 each second the lamp is on

storage	None erasible
Event	EF 2679
History	5

Name

CLOT (diag)
Lamp On Time
Increments by each second the lamp is on. Counter is reset with each Lamp off.

storage	volatile
Event	none
History	none

Name

CREF (diag)
Purge Valve Switches
Increments by 1 with each VREF 1 instruction.

storage	none erasible
Event	EF 2670
History	10

Name

CSRF (diag)
Increments by 1 each second the purge valve is activated.

storage	none erasible
Event	EF 2675
History	10

Name

CORF (diag)
pruge off time
Increments by 1 each second the purge valve is not activated.

storage	none volatile
Event	EF 2677
History	5

Name

CREC (diag)
Recycle Valve Switches
Increments by 1 with each VREC 1 instruction.

storage	none erasible
Event	EF 2671
History	10

Name

CSRC (diag)

Increments by 1 each second the recycle valve is activated.

storage	none erasible
Event	EF 2676
History	10

Name

CORC (diag)

recycle off time

Increments by 1 each second the recycle valve is not activated.

storage	none volatile
Event	EF 2678
History	5

Name

HTON (diag)

counts the time the heater was activated

Increments by 1 with each second.

storage	none erasible
Event	EF 2672
History	2

Name

LPOW (diag)

Increments by one with each 0.1 W fed into the lamp

storage	none erasible
Event	2673
History	5

Name

HPOW (diag)

Increments by one with each 0.1 W fed into the heater

storage	none erasible
Event	EF 2674
History	5

26.8. RID Instructions to build Applications

Name

WTMP
Wait Till Optical Unit Temperature has reached a defined Value

Synopsis

Instruction:	WTMP <temperature>, <time>
Reply:	<rc> WTMP <temperature>, <time>
Instruction:	ACT:WTMP?
Reply:	<rc> ACT:WTMP <time>
Instruction:	STOP:WTMP
Reply:	<rc> STOP:WTMP

Description

The WTMPinstruction allows to synchronize any other command to the event that a given temperature value is reached. If the <temperature> value is not reached within the <time> parameter an appropriate error is generated.
STOP:WTMP ends waiting for the temperature, but acts as if the temperature value was reached. The 'wait temperature' function also reacts to the ABRT instruction.

Parameters

<temperature>	15 .. 55 ['C]
<time>	0 .. 99999.00 [minutes]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Wait till temperature reaches 35'C, error after 12 minutes
	Instruction:	WTMP 35,12
	Reply:	RA 0000 WTMP 35.00,12.00
(b)	Comment:	Query the actual pressure wait time
	Instruction:	ACT:WTMP?
	Reply:	RA 0000 WTMP 1.12
(c)	Comment:	Stop waiting for temperature
	Instruction:	STOP:WTMP
	Reply:	RA 0000 STOP:WTMP

Name

WRI

Wait till a defined RI value is seen

Synopsis

Instruction:	`WRI <nRIU>, <time>`
Reply:	`<rc> WRI <nRIU>, <time>`
Instruction:	`ACT:WRI?`
Reply:	`<rc> ACT:WRI <time>`
Instruction:	`STOP:WRI`
Reply:	`<rc> STOP:WRI`

Description

The WRI instruction allows to synchronize any other command to the event that a given nRIU value is measured. If the <nRIU> value is not reached within the <time> parameter an appropriate error is generated.
STOP:WRI, ends waiting for the value, but acts as if the value had been reached. The 'wait for RI value' function also reacts to the ABRT instruction.

Parameters

<nRIU>	-500000 .. +500000 [nRIU]
<time>	0 .. 99999.00 [minutes]

Default Values

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Wait till -100000 nRIU are measured, error after 10 minutes
	Instruction:	`WRI -100000, 10`
	Reply:	`RA 0000 WRI -100000,10.00`
(b)	Comment:	Query the actual 'RI wait' time
	Instruction:	`ACT:WRI?`
	Reply:	`RA 0000 WRI 1.12`
(c)	Comment:	Stop waiting for RI value
	Instruction:	`STOP:WRI`
	Reply:	`RA 0000 STOP:WRI`

27. HPLC-Chip Cube Interface (G4240A)

27.1. Method Parameters of the HPLC-Chip Cube Interface

Name

VOP
Outer Valve's position

Synopsis

Instruction:	`VOP <position>`
Reply:	`<rc> VOP <position>`
Instruction:	`VOP?`
Reply:	`<rc> VOP <position>`
Instruction:	`AT:VOP <time>, <position>`
Reply:	`<rc> AT:VOP <time>, <position>`
Instruction:	`ACT:VOP?`
Reply:	`<rc> ACT:VOP <position>`

Description

Sets (and moves if appropriate) the outer valve's position.

Parameters

<position>	0,1,2 for corresponding position
	-1: no switch at parameter restoration
	-2: next position at end of run
	-3: next position at end of postrun
	-4: next position at end of analysis
	-5: next position at prepare (also APG remote prepare pulse)
	-6: next position at begin of run
	-7: next position as Time Table comand

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: set valve's home position to 1
Instruction: VOP 1
Reply: RA 0000 VOP 1

Name

VIP

Set Inner Valve's position

Synopsis

Instruction:	`VIP <position>`
Reply:	`<rc> VIP <position>`
Instruction:	`VIP?`
Reply:	`<rc> VIP <position>`
Instruction:	`AT:VIP <time>, <position>`
Reply:	`<rc> AT:VIP <time>, <position>`
Instruction:	`ACT:VIP?`
Reply:	`<rc> ACT:VIP <position>`

Description

Sets (and moves to if appropriate) the inner valve's position.

Parameters

<position>	0 ,1 for corresponding position -1: no switch at parameter restoration -2: next position at end of run -3: next position at end of postrun -4: next position at end of analysis -5: next position at prepare (also APG remote prepare pulse) -6: next position at begin of run -7: next position as Time Table comand

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	set inner valve's position 1
	Instruction:	VIP 1
	Reply:	`RA 0000 VIP 1`

27.2. HPLC-Chip Cube System Parameter

Name

TVOL
Set Trap Volume

Synopsis

Instruction:	`TVOL <volume>`
Reply:	`<rc> TVOL <volume>`
Instruction:	`TVOL?`
Reply:	`<rc> TVOL <volume>`

Description

The 'Trap volume' is the volume that is waited to be pumped by the 'Tap Pump' during the trapping/ enrichment phase.

Parameters

<volume> 0 .. 100000 [ul]

Default Values

0

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Set Trap Volume
	Instruction:	`TRAP 50`
	Reply:	`RA 0000 TRAP 50`
(b)	Comment:	Query the current setpoint
	Instruction:	`TRAP?`
	Reply:	`RA 0000 TRAP 50`
	Reply:	

Name

OAPO

Move to Operate after Power On

Synopsis

Instruction:	`OAPO <on/off>`
Reply:	`<rc> OAPO <on/off>`
Instruction:	`OAPO?`
Reply:	`<rc> OAPO <on/off>`

Description

When set to 1, the chipcube moves the chip to the operate position after power on.

Parameters

<on/off>	0 for off, 1 for on

Default Values

0

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)	Comment:	Move to operate after power on
	Instruction:	`OAPO 1`
	Reply:	`RA 0000 OAPO 1`
(b)	Comment:	Query the current setpoint
	Instruction:	`OAPO?`
	Reply:	`RA 0000 OAPO 1`

27.3. Control instructions for HPLC-Chip Cube Interface

Name

LOAD
Load/unload Chip

Synopsis

Instruction:	LOAD <position>
Reply:	<rc> LOAD <position>
Instruction:	ACT:LOAD?
Reply:	<rc> ACT:LOAD <actual position>

Description

Load chip to spray position, or load chip to standby position, or load chip to manual access position. ACT:LOAD? returns the actual position of the chip.

Parameters

<position>	0 for unload position, 1 for standby position, 2 for operate position
<actual position>	as position plus 3 for unknown position

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)

Comment:	load chip to spray position
Instruction:	LOAD 2
Reply:	RA 0000 LOAD 2

Name

ADXR

Adjust/query X offset relative

Synopsis

Instruction:	`ADXR <new_relative_offset>`
Reply:	`<rc> ADXR <new_relative_offset>`
Instruction:	`ADXR?`
Reply:	`<rc> ADXR <act_relative_offset>, <reserved>`

Description

Adjusts/queries the chip's relative tip offset in the X direction (horizontal direction). Positive X-axis is directed to the rear side of the instrument. The offset is relative to the absolute offset of SCX/SCXQ?. The total offset of the X-axis is the sum of ADXR? and SCXQ? and may not exceed the range of approx. -9000 ...6000 micro meter. Typical operate positions are about a total offset of 0 micro meter. If a requested adjustment move exceeds the valid range, no movement is performed, neither error is replied.

Parameters

<new_relative_offset>--10000 .. +10000 set relative offset in micro meter

<act_relative_offset> actual relative offset in micro meter

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Adjust X direction by 1mm
	Instruction:	`ADXR 1000`
	Reply:	`RA 0000 ADXR 1000`
(b)	Comment:	Query actual actual values
	Instruction:	`ADXR?`
	Reply:	`RA 0000 ADXR 200,-1500`

Name

ADYR

Adjust/query Y offset relative

Synopsis

Instruction: `ADYR <new_relative_offset>`

Reply: `<rc> ADYR <new_relative_offset>`

Instruction: `ADYR?`

Reply: `<rc> ADYR <act_relative_offset>, <reserved>`

Description

Adjusts/queries the chip's relative tip offset in the Y direction (vertical direction). Positive Y-axis is directed to the top of the instrument. The offset is relative to the absolute offset of SCY/SCYQ?. The total offset of the Y-axis is the sum of ADYR? and SCYQ? and may not exceed the range of approx. -9900 ...0 micro meter. Typical operate positions are about a total offset of -8000 micro meter. If a requested adjustment move exceeds the valid range, no movement is performed, neither error is replied.

Parameters

<new_relative_offset>--2000 .. +2000 set relative offset in micro meter

<act_relative_offset> actual relative offset in micro meter

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Adjust Y direction by approx. -0.2 mm

Instruction: `ADYR -200`

Reply: `RA 0000 ADYR -200`

(b) Comment: Query actual offset

Instruction: `ADYR?`

Reply: `RA 0000 ADYR -212,-1500`

Name

ADXS
Adjust X direction stepwise incremental

Synopsis

Instruction: `ADXS <steps>`
Reply: `<rc> ADXS <steps>`

Description

Adjusts the chip's tip in the X direction (horizontal direction) by the given steps number. Positive x-axis is directed to the rear side of the instrument. A step is approx. 100 micro meter. The end of the movement of the adjust is indicated by the info event 5407. The event's parameter holds the new relative offset in micro meters (same value as returned by ADXR?). The total offset of the X-axis is the sum of ADXR? and SCXQ? and may not exceed the range of approx. -9000 ...6000 micro meter. Typical operate positions are about a total offset of 0 micro meter. If a requested adjustment move exceeds the valid range, no movement is performed, neither error is replied.

Parameters

<steps>	-50 .. 50 steps valid range of requested incremental steps, regardless of current position.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Adjust X direction
Instruction: `ADXS 1`
Reply: `RA 0000 ADXS 1`

Name

ADYS

Adjust Y direction stepwise incremental

Synopsis

Instruction: ADYS <steps>

Reply: <rc> ADYS <steps>

Description

Adjusts the chip's tip in the Y direction (vertical direction) by the given steps. Positive y-axis is directed to the top of the instrument. A step is approx. 89 micro meter. The end of the movement of the adjust is indicated by the info event 5408. The event's parameter holds the new relative offset in micro meters (same value as returned by ADYR?). The total offset of the Y-axis is the sum of ADYR? and SCYQ? and may not exceed the range of approx. -9900...0 micro meter. Typical operate positions are about a total offset of -8000 micro meter. If a requested adjustment move exceeds the valid range, no movement is performed, neither error is replied.

Parameters

<steps> -50 .. 50 steps valid range of requested incremental steps, regardless of current position.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Adjust Y direction

Instruction: ADYS -1

Reply: RA 0000 ADYS -1

Name

SCX

Save/query X adjust absolute offset

Synopsis

Instruction:	`SCX`
Reply:	`<rc> SCX <act_abs_offset>`
Instruction:	`SCX <new_abs_offset>`
Reply:	`<rc> SCX <new_abs_offset>`
Instruction:	`SCXQ?`
Reply:	`<rc> SCXQ <act_abs_offset>`

Description

Saves and sets/queries the actual X absolute offset to/from pemanent memory. Without parameters, SCX sets and saves the current total offset (sum of ADXR? and SCXQ?) as the new absolute offset and resets the relative offset (ADXR?) to zero. With the argument `<new_abs_offset>`, the new absolute offset is defined and saved, and the relative offset is recalculated such that the relative offset reflects correctly the current X-axis position. No movements are done at all.

Parameters

<new_abs_offset>	-5000...5000 micro meter, new absolute offset used for ADXR-adjustments.
<act_abs_offset>	actual absolute offset used for ADXR-adjustments. Typical absolute offset is 0 micro meter.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Save X adjust
	Instruction:	`SCX`
	Reply:	`RA 0000 SCX`
(b)	Comment:	Query X adjust
	Instruction:	`SCXQ?`
	Reply:	`RA 0000 SCXQ 0`

Name

SCY

Save/query Y adjust absolute offset

Synopsis

Instruction:	SCY
Reply:	<rc> SCY <act_abs_offset>
Instruction:	SCY <new_abs_offset>
Reply:	<rc> SCY <new_abs_offset>
Instruction:	SCYQ?
Reply:	<rc> SCYQ <act_abs_offset>

Description

Saves and sets/queries the actual Y absolute offset to/from pemanent memory. Without parameters, SCY sets and saves the current total offset (sum of ADYR? and SCYQ?) as the new absolute offset and resets the relative offset (ADYR?) to zero. With the argument <new_abs_offset>, the new absolute offset is defined and saved, and the relative offset is recalculated such that the relative offset reflects correctly the current Y-axis position. No movements are done at all.

Parameters

<new_abs_offset>	-9500...-6000 micro meter, new absolute offset used for ADYR-adjustments.
<act_abs_offset>	actual absolute offset used for ADYR-adjustments. Typical absolute offset is -8000 micro meter.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	Save Y adjust
	Instruction:	SCY
	Reply:	RA 0000 SCY
(b)	Comment:	Query Y adjust
	Instruction:	SCYQ?
	Reply:	RA 0000 SCYQ -8120

Name

TRP

Define Trap Pump

Synopsis

Instruction:	TRP <pump id>
Reply:	<rc> TRP <pump id>
Instruction:	TRP?
Reply:	<rc> TRP <pump id>

Description

Defines a pump to be used for the generation of the enrichtment/injection/trapping flow.

Parameters

<pump id>	string in double quotes, consiting of product number and serial number of the pump, seperated by a colon.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)

Comment:	Set trap pump
Instruction:	TRP "G1310A:DE72004312"
Reply:	RA 0000 TRP "G1310A:DE72004312"

Name

GRP
Define Separation Pump

Synopsis

Instruction:	`GRP <pump id>`
Reply:	`<rc> GRP <pump id>`
Instruction:	`GRP?`
Reply:	`<rc> GRP <pump id>`

Description

Defines a pump to be used for the generation of the separation flow.

Parameters

`<pump id>`	string in double quotes, consiting of product number and serial number of the pump, seperated by a colon.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: Set separation pump
Instruction: GRP "G1310A:DE72002941"
Reply: `RA 0000 GRP "G1310A:DE72002941"`

Name

PTRP?

Query for Possible Trap Pumps

Synopsis

Instruction: `PTRP?`

Reply: `<rc> PTRP <pump ids>`

Description

Lists all actually CAN connected pumps that are valid to be defined for trapping.

Parameters

`<pump ids>` Pairs of product and serial numbers, seperated by semicolon.

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: Query separation pumps

Instruction: PTRP "G1310A:DE72002941;G1310A:DE72004312"

Reply: `RA 0000 PTRP "G1310A:DE72002941;G1310A:DE72004312"`

Name

PGRP?

Query for Separation Pumps

Synopsis

Instruction: PGRP?

Reply: <rc> PGRP <pump ids>

Description

Lists all actually CAN connected pumps that are valid to be defined for separation.

Parameters

<pump ids> Pairs of product and serial numbers, seperated by semicolon.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Query separation pumps
Instruction: PGRP "G1310A:DE72002941;G1310A:DE72004312"
Reply: RA 0000 PGRP "G1310A:DE72002941;G1310A:DE72004312"

Name

EXRO

Exchange Rotor

Synopsis

Instruction: EXRO

Reply: <rc> EXRO

Description

Moves the mechanical hardware to a position appropriate for exchanging the valve rotor.

Parameters

none.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Exchange rotor
Instruction: EXRO
Reply: RA 0000 EXRO

Name

INDL
Indoor Light

Synopsis

Instruction: INDL <on/off>
Reply: <rc> INDL <on/off>

Description

Turn on or off indoor light.

Parameters

<on/off> 0 for off, 1 for on.

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Turn on indoor light
Instruction: INDL 1
Reply: RA 0000 INDL 1

Name

VACA
Valve calibration inner rotor

Synopsis

Instruction:	`VACA`
Reply:	`<rc> VACA`
Instruction:	`VACA?`
Reply:	`<rc> VACA <status>`

Description

Starts calibration run of inner valve rotor.

Parameters

`<status>:`	`0: current calibration failed`
	`1: current calibration successfull`

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a)

Comment:	Start valve calibration
Instruction:	`VACA`
Reply:	`RA 0000 VACA`

Name

VACB
Valve calibration outer rotor

Synopsis

Instruction: VACB
Reply: <rc> VACB
Instruction: VACB?
Reply: <rc> VACB <status>

Description

Starts calibration run of outer valve rotor.

Parameters

<status>: 0: current calibration failed
1: current calibration successfull

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Start valve calibration
Instruction: VACB
Reply: RA 0000 VACB

Name

VABT
Valve Calibration abort

Synopsis

Instruction: VABT
Reply: <rc> VABT

Description

Aborts running valve calibrations (Performs abort of operations on local module).

Parameters

<none>

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Start valve calibration
Instruction: VABT
Reply: RA 0000 VABT

Name

SRST
Spray Restart

Synopsis

Instruction: `SRST`
Reply: `<rc> SRST`

Description

Closes valves, moves to standby position , then back to operate and opens valves. Spray is restarted.

Parameters

<none>

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: Spray Restart
Instruction: `SRST`
Reply: `RA 0000 SRST`

27.4. Actual Values of the HPLC-Chip Cube Interface

Name

ACT:VOP?
Actual Outer Valve's position

Synopsis

Instruction:	ACT:VOP?
Reply:	<rc> ACT:VOP <position>

Description

Query the outer valve's actual position.

Parameters

<position>	-1 for off position 0,1,2 for corresponding position

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a)	Comment:	query the valve's actual position
	Instruction:	ACT:VOP?
	Reply:	RA 0000 ACT:VOP 1

Name

ACT:VIP

Query Inner Valve's position

Synopsis

Instruction: ACT:VIP?

Reply: <rc> ACT:VIP <position>

Description

query the inner valve's actual position.

Parameters

<position> -1 for off position, 0 ,1 for corresponding position

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query inner valve's position

Instruction: ACT:VIP?

Reply: RA 0000 ACT:VIP 1

Name

ACT:TRAP
Query Trap State

Synopsis

Instruction: `ACT:TRAP?`
Reply: `<rc> ACT:TRAP <on/off>`

Description

query the actual trappin state.

Parameters

<on/off> 0 for no trapping in progress, 1 for trapping in progress

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query actual trap state
Instruction: ACT:TRAP?
Reply: `RA 0000 ACT:TRAP 0`

Name

ACT:ACHP
Query Actual Chip

Synopsis

Instruction: `ACT:ACHP?`
Reply: `<rc> ACT:ACHP <chip id>`

Description

query the actually loaded chip's id.

Parameters

<chip id> string

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query actual chip
Instruction: ACT:ACHP?
Reply: `RA 0000 ACT:ACHP "G4244A003"`

Name

ACT:TVOL
Query Actual Trap Volume

Synopsis

Instruction: `ACT:TVOL?`
Reply: `<rc> ACT:TVOL <volume>`

Description

query the actual trap volume. If no trapping is in progress the actual volume is 0.

Parameters

<volume> 0 .. TVOL-Parameter

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query actual trap volume
Instruction: ACT:TVOL?
Reply: RA 0000 ACT:ACHP 22

Name

ACT:LTME
Query Actual Load Time

Synopsis

Instruction: ACT:LTME?
Reply: <rc> ACT:LTME <position>, <time>

Description

query the actual time since the chip is in it's actual position.

Parameters

<time>	0 .. 32bit integer
<position>	0 .. 3

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query actual time since loaded
Instruction: ACT:LTME?
Reply: RA 0000 ACT:LTME 2,465

Name

ACT:LOAD
Query Actual Load State

Synopsis

Instruction: ACT:LOAD?
Reply: <rc> ACT:LOAD <position>

Description

query the actual load state.

Parameters

<position> 0 .. 2 , 0 for unload, 1 for standby, 2 for operate

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query actual time since loaded
Instruction: ACT:LTME?
Reply: RA 0000 ACT:LTME 2,465

Name

ACT:CMNT
Query actual Chip Comment

Synopsis

Instruction: `ACT:CMNT?`
Reply: `<rc> ACT:CMNT <comment>`

Description

query the actualchip comment.

Parameters

<comment> operator defined string

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: query actual comment
Instruction: ACT:CMNT?
Reply: `RA 0000 ACT:CMNT "blue bird"`

Name

ACT:TOPT

Query Total Operate Time

Synopsis

Instruction: `ACT:TOPT?`

Reply: `<rc> ACT:TOPT time>`

Description

query the total time the chip was in operate position.

Parameters

<time> time in units of 0.1 hours

Reply Codes <rc>

`RA 0000` no error, instruction has been accepted

Examples

(a) Comment: query total operate time

Instruction: ACT:TOPT?

Reply: `RA 0000 ACT:TOPT 82`

27.5. Instructions to access chip tag data

The following 4 instructions give acces to the tag data of the chip:

CTAG:ID?	to query for identification info
CTAG:OP?	to get information of how to operate the chip
CTAG:HIST?	to get some history information of the chip
CTAG:CMNT	to set or read a user define comment

Name

CTAG:ID?

Read Chip Identification

Synopsis

Instruction: CTAG:ID?

Reply: <rc> CTAG:ID <ProductNum>, <SerialNum>, <revision>

Description

returns the information that was read when the last chip was loaded

Parameters

<ProductNum>	example G4240-12345
<SerialNum>	example DE12340987
<Revision>	1 .. 255, actual revision of this product

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RA 0610 actually no chip is installed.

Examples

(a) Comment: Read chip id

Instruction: CTAG:ID?

Reply: RA 0000 CTAG:ID G4240-12345, DE12340987, 1

Name

CTAG:OP?

Read Operational Chip Info

Synopsis

Instruction: CTAG:OP?

Reply: <rc> CTAG:OP <appl>,<tip>,<trap>,<an.col>,<max press>

Description

returns the information that was read when the last chip was loaded

Parameters

<appl>	0 .. 255, application key (1 for enrichment + gradient)
<tip>	0 .. 16, number of tips of the chip
<trap col>	0 .. 16, number of enrichment (trapping) columns
<an. col>	0 .. 16, number of analytical columns
<max press>	0 .. 400 bar

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RA 0610 actually no chip is installed.

Examples

(a) Comment: Read operation related data

Instruction: CTAG:OP?

Reply: RA 0000 CTAG:OP 1, 1, , 2, 2, 50

Name

CTAG:HIST?
Read Chip History

Synopsis

Instruction: CTAG:HIST?
Reply: <rc> CTAG:HIST <inj>, <first load>, <last load>

Description

returns the information that was read when the last chip was loaded

Parameters

<inj>	0 .. 2^16, number of injections
<first load>	0 .. 2^32, time of first load
<last load>	0 .. 2^32, time of last load

Reply Codes <rc>

RA 0000 no error, instruction has been accepted
RA 0610 actually no chip is installed.

Examples

(a) Comment: Read chip history
Instruction: CTAG:HIST?
Reply: RA 0000 CTAG:HIST 52, 5234, 1222333444, 1222333555

Name

CTAG:CMNT?

Read/Write Chip Comment

Synopsis

Instruction:	CTAG:CMNT?
Reply:	<rc> CTAG:CMNT <Comment>
Instruction:	CTAG:CMNT <Comment>
Reply:	<rc> CTAG:CMNT <Comment>

Description

returns the information that was read when the last chip was loaded

Parameters

<Comment> string of max 16 characters

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

RA 0610 actually no chip is installed.

Examples

(a)	Comment:	Set chip comment
	Instruction:	CTAG:CMNT "My first Chip"
	Reply:	RA 0000 CTAG:CMNT "My first Chip"
(b)	Comment:	Read chip comment
	Instruction:	CTAG:CMNT? "My first Chip"

27.6. Instructions to build HPLC-Chip Cube Applications

Name

TRAP
start trap

Synopsis

Instruction: TRAP
Reply: <rc> TRAP

Description

Once trapping is started, the instrument counts the trap pumps volume, till the value defined by TVOL is reached.

Parameters

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: start trapping
Instruction: TRAP
Reply: RA 0000 TRAP

Name

GIVO

Get Injection Volumes

Synopsis

Instruction: GIVO

Reply: <rc> GIVO

Description

retrieves injection volume for correct trapping from injectors.

Parameters

<none>

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment:

Instruction: GIVO

Reply: RA 0000 GIVO

Name

AVNP

Valve(s) to next position at prepare

Synopsis

Instruction: AVNP

Reply: <rc> AVNP

Description

A valve having its position setup with -5 switches to next position at prepare.

Parameters

none

Reply Codes <rc>

RA 0000 no error, instruction has been accepted

Examples

(a) Comment: Valve(s) to next position

Instruction: COSY:ISEQ <list>, "AVNP"

Reply: RA 0000 COSY:ISEQ <list>, "AVNP"

27.7. HPLC-Chip Cube Diagnose Buffers

Name

CHHI (diag)
History of chips

Characteristics

max entries	20
max characters	40
storage	battery buffered
availability	always
entry	with each unload of a chip

Contents

time, product #, serial #, injections, time in spray

time	seconds since 1970 when chip was unloaded
product #	product number of chip
serial #	serial number of chip
injections	number of injections as stored in the chip's tag
time in spray	accumulated seconds of total time the chip was in operate position

Description

History of chips that had been installed in the instrument.

27.8. HPLC-Chip Cube Diagnose Counters

Name

IVLV (diag)

Switches of inner valve

Increments by one, each time the inner valve moves. Includes also initialization movements and switches to off and restore positions

storage	battery buffered
Event	5483
History	none

Name

OVLV (diag)

Switches of outer valve

Increments by one, each time the outer valve moves.

storage	battery buffered
Event	5482
History	none

Name

LOAD (diag)

Chip load cycles

Increments by one with each loading of a chip.

storage	battery buffered
Event	5485
History	none

Name

TOPE (diag)

Time in operate position

Counts seconds while the instrument is in Operate Position.

storage	battery buffered
Event	5480
History	none

Name

MVOL (diag)

Moves/inits of operating lever

Counts the number of moves.

storage	battery buffered
Event	none
History	none

Name

MVWD (diag)

Moves of withdraw drive

Counts the number of moves.

storage	battery buffered
Event	none
History	none

Name

MVMT (diag)

Moves of multi tip drive

Counts the number of moves.

storage	battery buffered
Event	none
History	none

Name

MVCL (diag)

Moves of clamp drive

Counts the number of moves.

storage	battery buffered
Event	none
History	none

28. Agilent 1200 Series Events

28.1. General Description of the Event CU

The Agilent 1200 Series modules generate events to tell the outer world, that something happened within the module, that is considered worth to be mentioned. This can be a state change, any error, changes on the remote lines, module restarts... An event consists of a prefix and a 4 digit number, and optionally: the time when it was generated and a parameter. The prefix tells what kind of event it is, the number identifies the event istself. The numbers are unique over all Agilent 1200 Series modules.

kind of event	prefix
error event	EE
state change event	ES
reset event	EC
info event	EV
inj prog event	EI
remote event	ER
feedback event	EF
peak detector event	EP

The event CU stores up to the last 40 events for each controller connected to the module; using first-in first-out.One event is transmitted each time the event CU is read. If the storage space is filled up , the event EV 0067, <time>, <param> is inserted in the queu, further events are skipped.
The event numbers ares sorted according to the module that generated the event. Common events may be generated by any Agilent 1200 Series module.

from	to	module
0001	0999	common events
1000	1999	SPM
2000	2599	PUMP
2600	2799	RID
2800	2999	COL COMP
3000	3999	MWD, DAD
4000	4999	ALS

from	to	module
5000	5999	CE
6000	6999	FLD
7000	7999	VWD

C-code reference find in ./conf/src/events.h and in allfiles ./*/src/*.cfg

28.2. Events after Opening the Event CU

Each time an instrument controller connects to a module (via LICOP) and opens the event cu, a series of events are generated for this controller. The first event is EC 0003 as connection event. the second one is EC0010 and indicates that the module is ready for communication e.g that all configuration detection and initialization after powerfail is done.
The next events are one for each (macroscopic) statemachine of the module:
that is one out of:

ES 0103 prerun
ES 0104 run
ES 0105 postrun
ES 0100 leak
ES 0101 shutdown

and one out of:

ES 0102 error
ES 0107 noerror

and one out of:

ES 0110 analysis
ES 0111 no analysis

and one out of:

ES 0109 ready
ES 0108 not ready

and one out of:

ES 0112 test
ES 0113 no test

then followed by the module specific statechange events for each externally propagated statemachine: for the pumps these are the pump on/off/init event, for the detectors that are the lamp on/off/init events....

28.3. Common Events

28.3.1. Restart Events

EC 0001 coldstart
parameter reset, hard- and software reset
EC 0002 warmstart
hard- and software reset, parameter remain unchanged
EC 0003 hotstart, new connection
parameters and states remain unchanged
EC 0004 resident start
device ready for use (resident system)
after this event the module is configured and ready to communicate with any host.
EC 0010 device ready for use
after this event the module is configured and ready to communicate with

any host.

28.3.2. Common Error Events

EE 0050 FPGA load error
FPGA cannot load, (detailed load error can be viewed in FPIH diagnostic buffer)
EE 0062 timeout
system did not get ready, timeout occured
EE 0063 shutdown
serious error in other module, shutdown
EE 0064 leak detected

EE 0068 fan failed
hall sensor feedback indicates rotational speed below acceptable limit
EE 0070 remote wait timed out
remote pulse or line state has not been seen within specified time
EE 0071 lost can partner
can communication has lost partner while an analysis was in progress
EE 0072 controler wait timed out
at begin of an analysis one of the controllers didn't show ready within timeout.
EE 0080 leak hardware error: ntc-board-sensor short
the ntc sensor on the pc-board is shorted
EE 0081 leak hardware error: ntc-board-sensor open
the ntc sensor on the pc-board is open or not assembled
EE 0082 leak hardware error: ptc-leak-sensor short
the ptc sensor that indicates the leak is shorted
EE 0083 leak hardware error: ptc-leak-sensor open
the ptc sensor that indicates the leak is open or not connected or the wires to this sensor are broken
EE 0090 covertop not closed
the top cover has been opened while opening was not allowed violation of cover top open
EE 0098 incompatible firmware
the firmware versions of two modules that are linked by an internal remote service are incompatible
EE 0301 no COSY list
requested cosy list is not available
EE 0302 COSY list running
a cosy list is still running
EE 0304 Too many COSY:REPT
There are too many COSY:REPT-loops nested in the cosylist. The parameter of the event shows the maximum allowed number of nested REPT's.
EE 0306 cosy clock not started.
Without having the clock started the command COSY:WCLK (Parameter = 1), COSY:AT ("EE 0306, 2") or COSY:PER ("EE 0306, 2") is executed
EE 0307 Memory Overflow in COSY.
Parameter = 1,2: Overflow of a queue handling the COSY-syncs.
Parameter = 3: Overflow of a queue handling the COSY-counters.
Parameter = 4: Too many counters in use. The maximum number is 15.
Parameter = 900: The system ran out of memory (RAM) executing one of the commands COSY:SYNP, COSY:CAPT or COSY:NEW (as cosy element)
EE 0308 COSY timed out.
After starting a list no new COSY-command was executed for the time specified by the command COSY:TOUT
EE 0309 COSY structure error.
Parameter = 1: COSY:IF(N)-condition is false and there is no corresponding CO-

SY:ELSE or COSY:FI until end of COSY list.
Parameter = 2: COSY:COND is last element of a COSY list and the condition is evaluated as FALSE.
Parameter = 3: COSY:NCND is last element of a COSY list and the condition is evaluated as FALSE.

28.3.3. Common Statechange Events

ES 0100 leak
this event indicates, that a statechange to leak occured

ES 0101 shutdown
ths event indicates, that a statechange to shutdown occured

ES 0102 error
this event indicates, that a statechange to error occured

ES 0103 prerun
ths event indicates, that a statechange to prerun occured

ES 0104 run
this event indicates, that a statechange to run occured

ES 0105 postrun
this event indicates, that a statechange to postrun occured

ES 0107 noerror
this event indicates, that a statechange to noerror occured

ES 0108 not ready
ths event indicates, that a statechange to notready occured

ES 0109 ready
this event indicates, that a statechange to ready occured

ES 0110 analysis state entered
the module takes part in an analysis

ES 0111 no analysis
the analysis is done

ES 0112 test
the event indicates that a statechange to TEST occured

ES 0113 no test
the event indicates that a statechange to NO_TEST occured

ES 0114 analysis pending
an analysis has been initiated, the can wired modules start synchronizing and wait for their controllers to show ready

ES 0128 not ready for start
ths event indicates that the module is not ready to start an analysis

ES 0129 ready for start
this event indicates, that the module is ready to start an analysis

28.3.4. Common Info Events

EV 0051 Cannot start because FPGA is loaded with error
a 'start' is declined, because at least one FPGA is not succesfully loaded

EV 0067 event has been lost
host didn't read events, so buffer overflow occured, further events are lost

EV 0069 battery low
indicates that power of the lithium battery is below a limit;
possible loss of data

EV 0150 starting analysis aborted via CAN
attmept to start an analysis was rejected by another CAN-module

EV 0151 start of analysis rejected during 'startup' phase

EV 0152 start of analysis rejected during 'leak' state

EV 0153	start of analysis rejected during 'shutdown' state
EV 0154	start of analysis rejected during 'analysis' state
EV 0155	start of analysis rejected during 'run' state
EV 0156	start of analysis rejected during 'abort' state
EV 0157	start of analysis rejected during 'no fan rotation' state
EV 0160	controller stop request a controller is requested to abort the automation
EV 0161	controller start request a controller is requested to begin with the automation
EV 0170	Clock changed The module's internal clock has been set by a controller
EV 0171	Instruction lock has changed the instruction classes locked have been changed.
EV 0200	method modified this event indicates, that the actual method has been modified
EV 0201	sysvar modified this event indicates, that the sysvars have been modified
EV 0202	home parameter modified this event indicates, that the home parameter have been modified
EV 0203	timetable movified this event indicates, that the timetable has been modified
EV 0204	cosy list movified this event indicates, that a COSY list has been modified
EV 0205	top cover open the on-board sensor detected , that the top foam part cover of the module is removed
EV 0206	method saved
EV 0207	method loaded
EV 0208	method deleted
EV 0209	flash mismatch info in FLASH memory not supported (e.g. old firmware revision)
EV 0210	flash inconsistent info in FLASH memory inconsistent
EV 0211	runbuffer full runbuffer memory has overflow
EV 0212	auto-on method auto-on method loaded
EV 0213	error method error method loaded
EV 0214	load status change status of load task change
EV 0215	data modified any change in generic data storage
EV 0216	data changed at least one change since last listing
EV 0217	raw signals changed number of send rawdata signals changed
EV 0300	synchronization between module and controller This event is generated by the cosy command COSY:EVNT.

EV 0305 reply error during cosy list execution
A reply error occured during cosy list execution. The error number is given as parameter to the event.

28.3.5. Common Limit Events

EF 0400 unspecified limit reached
a diagnose counter reached its limit set by DIAG:CLMT

EF 0401 ON-time limit reached
diagnose counter TONT reached its limit set by DIAG:CLMT

EF 0402 RUN-counter limit reached
diagnose counter CRUN reached its limit set by DIAG:CLMT

EF 0403 ANALYSIS-counter limit reached
diagnose counter CNAN reached its limit set by DIAG:CLMT

28.4. Pump Events

28.4.1. Pump Error Events

EE 2014 pressure above upper limit

A pressure monitor filters for three successive values above the limit defined by HIPR. Pressure values are monitored once per second.

The parameter for the event is the last used pressure value.

EE 2015 pressure below lower limit

The monitor for lower limit detection is activated once the pressure exceeds the limit defined by LOPR. The event is generated if the pressure drops for more than ten seconds below the limit.

The parameter for the event is the last used pressure value.

EE 2016 invalid pressure signal

The measured pressure is below the valid range. This usually indicates a faulty or missing pressure sensor.

EE 2040 valve failed to switch

During boot up all valves are switched once and the current through the coils is measured whiled switching. (this is the typical click-click-click... of the quaternary pump after power on!) . If the current doesn't mach a predefined profile, due to either open circuitry or shortcut, the event is generated and the valve option is disabled => a quaternary pump will be isocratic; a binary pump with solvent selection will show no solvent selection option!

The parameter of the event defines the defective valve and/or electronics:

for the quaternary pump that is

0 for channel A
1 for channel B
2 for channel C
3 for channel D

for the binary pump these are

0 for solvent A1
1 for solvent B1
2 for solvent A2
3 for solvent B2

EE 2041 channel B motor power too high

Once the pump drive is initialized, the motor power is periodically monitored. Actually it is not the real power but the difference between the actual position and the 'should be position'. The difference is (in the first approach) linear to the current through the motor's coils. The event is generated if for successive readings are out of limit.

EE 2042 channel A motor power too high

See description for EE 2041

EE 2043 electronic fuse of MCGV

The electronic fuses are checked twice per second

param: 0 coil A or B, 1 for coil C or D

EE 2044 electronic fuse of active inlet valve

The electronic fuses are checked twice per second

a parameter of 0 means the left AIV, a parameter of 1 means the right AIV.

EE 2045 electronic fuse of purge valve

The electronic fuses are checked twice per second

EE 2046 encoder cable of channel B missing

Connectores are checked twice per second

EE 2048 channel B active inlet valve missing

Connectores are checked twice per second

EE 2050 encoder cable of channel A missing

Connectores are checked twice per second

EE 2052 channel A active inlet valve missing
Connectores are checked twice per second

EE 2053 time out for applicational wait function
See instructions WPRS, WDEL, WSUC, WVOL for applicational wait functions. All wait functions are started with a maximum time to wait . The event is generated if the time is counted back to zero before the goal is reached.
(e.g. pressure didn’t reach a given limit within the predicted time)

EE 2054 missing pressure readings
a parameter of zero indicates problems with the A/D converter, a parameter of 1 indicates problems with the interrupt system.

EE 2055 solvent counter reached zero
If a solvent counter’s limit is greater 0, it counted back to zero.
The parameter of the event indicates the counter:
1 for BOTA,
2 for BOTB,
3 for BOTC,
4 for BOTD

EE 2060 new pump configuration
After power on the pumps compare the actual hardware with hardware that should be availabe (stored in FLASH memory). A detected mismatch leads to the event. The parameter of the event defines what should be:
0 for quaternary
1 for high pressure gradient
2 for isocratic
see also instructions TYPE? and OPT?

EE 2061 pump turned off due to open cover
If the cover of the instrument is open, the fans are turned off. Without fan a continous operation of the pump could lead to overheating the motors, therefore after not longer than 30 seconds, the pump drive’s electronic is shut off with generating the event.

EE 2062 wrong readings from hybrid sensor
the temperature readings from the motors power hybrids are expected to be within a certain limit. If the limit is violated, no sensful temperature can be calculatet therfore the event is generated.
param= 0 for channel A, param = 1 for channel B

EE 2063 hybrid over temp
if the motor’s power hybrid temperature exceeds 100 ’C the electronics are shut off to prevent the motors from being overheated.
param = 0 for channel A, param = 1 for channel B

EE 2064 flow exceeds limit while pressure control
if pressure control is active and the allowed maximum flow is exceeded. If the flow is specified with a negativ sign, no error event is generated.

EE 2065 Pump has been turned off during analysis
EE 2066 Column flow unstable
EE 2067 No EMPV connected
EE 2068 No flow sensor connected
EE 2069 EMPV initialization failed
EE 2090 Flow sensor not supported by pump
EE 2091 Unsupported HW configuration

Initialization errors for channel A:

EE 2201 servo restart failed
Motor didn’t turn on or no feed back from initial movement
Either motor is defective or not connected

EE 2202	pump head missing mechanical limit not found
EE 2203	home position out of limit index too close to pump head, mis-/disadjusted/ loose plunger
EE 2204	home position not found index too for from pump head, broken cable, blocked index slot on encoder
EE 2205	no index found broken encoder cable
EE 2206	stroke length misadjusted mechanical obstacle, not enough length for total stroke
EE 2207	init failed due to unknown defect general init time out

Initialization errors for channel B:

EE 2211	servo restart failed See 2201
EE 2212	pump head missing See 2202
EE 2213	home position out of limit See 2203
EE 2214	home position not found See 2204
EE 2215	no index found See 2205
EE 2216	stroke length misadjusted See 2206
EE 2217	init failed due to unknown defect See 2207

28.4.2. Pump State Change Events

Remark: These statechange events are also part of the logon events of any controller. The events belong all to only one state machine. Since by principle for the Agilent 1200 series modules ,the statechange events are entry actions of a state, the last event generated gives always the actual state.

Pump Statemachine:

ES 2100	pump off Pump has been turned off. Either due to an error or due to PUMP 0 or due to auto off.
ES 2101	init pump Either due to PUMP 1 or due to automatic turn on.
ES 2102	pump standby Due to PUMP 2.
ES 2104	pump on This event indicates the transition from init to On or from standby to On. The pump can now deliver mobile phase.

Composition Ramp Statemachine:

ES 2110	composition ramp is off This event indicates the transition to no ramp processing.
ES 2111	composition ramp is on This event indicates the start and/or continue of a composition ramp.
ES 2112	composition ramp is in hold state This event indicates the transition from ramp on to ramp hold.

Flow Ramp Statemachine:

ES 2113	flow ramp is off This event indicates the transition to no ramp processing.
ES 2114	flow ramp is on This event indicates the start and/or continue of a composition ramp.
ES 2115	flow ramp is in hold state This event indicates the transition from ramp on to ramp hold.

Sealwash pump Statemachine:

ES 2552	Sealwash pump off (not running, also asserted when not running in periodic mode).
ES 2553	Sealwash pump running in single mode.
ES 2554	Sealwash pump running in periodic mode.

Operating pressure control Statemachine:

Operating pressure control is only available for Binary Pump SL (G1312B).

ES 2123	Operating pressure control enabled.
ES 2124	No operating pressure control.

28.4.3. Pump Info Events

EV 2225	No analysis possible while pump is off a request for an analysis via CAN had to be rejected due to the 'pump off" state.
EV 2226	No analysis possible while pump is in test mode
EV 2227	No analysis possible while pump is purging
EV 2228	No analysis possible while pump is performing a fast composition change
EV 2229	Column flow in limit
EV 2230	Column flow out of limit
EV 2231	Fast composition change started
EV 2232	Fast composition change complete
EV 2233	Calibration table stored. Parameter: table reference, 1..10
EV 2234	Calibration table deleted. Parameter: table reference, 1..10
EV 2235	Fast Composition Change failed
EV 2236	Passed FSAC step. Parameter: step# -3..4
EV 2240	Solvent name changed

28.4.4. Pump Counter Limit Events

Counter limit events are related to a Diagnosis Counter (see seperate section). A counter consists of mainly three parts: the total counter value, a presettable counter value and a limit. Once the presettable counter value exceeds the limit the associated event is generated with the parameter 1. If the limit is set to zero the limit verification is off. The event is also generated if either limit or preset value are modified. A parameter of 0 means always that the presettable value of the counter is below the limit, a parameter of 1 means that this value is equal or above the limit. Changes from above to above or from below to below are not reflected as an event.
The name after the event number is the name of the counter the event belongs to.

EF 2070	CHWC, Wear of left drive assembly
EF 2071	CUWC, Wear of right drive assembly
EF 2072	LITR, Liters pumped
EF 2073	CHVO, Liters pumped through left drive assembly
EF 2074	CUVO, Liters pumped through right drive assembly

EF 2075	OVC, Outlet valve switches of left drive assembly
EF 2076	COVC, Outlet valve switches of right drive assembly
EF 2077	IVC, Inlet valve switches of left drive assembly
EF 2078	CIVC, Inlet valve switches of right drive assembly
EF 2079	GCKA, MCGV valve A switches
EF 2080	GCKB, MCGV valve B switches
EF 2081	GCKC, MCGV valve C switches
EF 2082	GCKD, MCGV valve D switches
EF 2083	BOTA, Liters of Bottle A
EF 2084	BOTB, Liters of Bottle B
EF 2085	BOTC, Liters of Bottle C
EF 2086	BOTD, Liters of Bottle D
EF 2087	SSVA, Switches of Solvent Selection Valve A1/A2
EF 2088	SSVB, Switches of Solvent Selection Valve B1/B2

28.4.5. Pump Observation Events

EO 2301	Backflow of left drive assembly
EO 2302	Backflow of right drive assembly
EO 2303	No backflow of left drive assembly
EO 2304	No backflow of right drive assembly
EO 2305	Piston leak of left drive assembly
EO 2306	Piston leak of right drive assembly
EO 2307	No Piston leak of left drive assembly
EO 2308	No Piston leak of right drive assembly
EO 2309	Gas bubble of left drive assembly
EO 2310	Gas bubble of right drive assembly
EO 2311	No gas bubble of left drive assembly
EO 2312	No gas bubble of right drive assembly

EO 2313	Compressibility setpoint for left drive assembly has changed
EO 2314	Compressibility setpoint for right drive assembly has changed
EO 2315	Flow has been changed
EO 2316	Composition has been changed
EO 2317	Flow has been turned on
EO 2318	Flow has been turned off

28.5. Column Compartment Events

28.5.1. Column Compartment Error Events

EE 2810 failure of power circuitry detected

EE 2811 'wait for temperature' function of left heat exchanger assembly timed out.

EE 2812 'wait for temperature' function of right heat exchanger assembly timed out.

EE 2821 temperature sensor readings failed
parameter:
- 0 left heat exchanger column sensor
- 1 left heat exchanger heat sink sensor
- 2 right heat exchanger column sensor
- 3 right heat exchanger heat sink sensor
- 4 ambient correction sensor
- 5 auxiliary external sensor

EE 2823 trend analysis for sensor failed, e.g. misconnected sensor or peltier
parameter: same as EE 2821

EE 2825 valve failed to switch
param:
- 0 for position 1->2
- 1 for position 1->6

EE 2829 left fan failed
param: last measured rotations per minute

EE 2830 right fan failed
param: last measured rotations per minute

EE 2833 column compartment automatically turned off due to open cover
param: none

EE 2836 maximum heat exchanger column side temperature exceeded
(>105') param: 0 for left heat exchanger
param:1 for right heat exchanger

EE 2837 maximum heat exchanger heat sink temperature exceeded
(>70') param:1 for left heat exchanger
param 3 for right heat exchanger

28.5.2. Column Compartment State Change Events

Temperature Control Statemachine:

ES 2800 temperature control on
ES 2801 temperature control off

Calibration Statemachine:

ES 2802 calibration in progress
ES 2803 no calibration in progress

Left Column ID Statemachine:

ES 2832 no column id in left heat exchanger assembly
ES 2844 left column id valid

Right Column ID Statemachine:

ES 2843 no column id in right heat exchanger assembly
ES 2845 right column id valid

28.5.3. Column Compartment Info Events

EV 2826	valve switched to position 0
EV 2827	valve switched to position 1
EV 2831	left column id found
EV 2835	condens water
EV 2838	left column id modified
EV 2839	right column id modified
EV 2840	calibration allowed
EV 2841	calibration not allowed
EV 2842	right column id found
EV 2846	verifying left column id failed
EV 2847	verifying right column id failed
EV 2848	incremented injection counter of left column id
EV 2849	incremented injection counter of right column id
EV 2856	no start allowed while temperature control off

28.5.4. Column Compartment Feedback Events

EF 2824	column switching valve counter for switching cycles exceeds feedback limit
EF 2850	left injection counter feedback limit for left injection counter exceeded
EF 2851	right injection counter feedback limit for right injection counter exceeded
EF 2852	left peltier power feedback limit for left peltier power exceeded
EF 2853	right peltier power feedback limit for right peltier power exceeded
EF 2854	operation time of left peltiers feedback limit of left peltier heating and cooling time exceeded
EF 2855	operation time of right peltiers feedback limit of right peltier heating and cooling time exceeded

28.6. Autosampler Events

28.6.1. ALS Error Events

EE 4002 Movement failed.
One of the four axis of the sample transport failed to move. The parameter of the event defines the defective axis:
0 for X-axis
1 for Z-axis
2 for Gripper rotation (Theta)
3 for Gripper finger

EE 4014 Valve to bypass failed.
The lever arm of the switch mechanism didn't reach the touch sensor.

EE 4015 Valve to mainpass failed.
The lever arm of the switch mechanism didn't reach the touch sensor.

EE 4017 Needle up failed.
The needle up movement didn't reach the upper sensor.

EE 4018 Needle down failed.
The needle down movement didn't reach the lower sensor.

EE 4019 Missing vial.
The gripper arm didn't find a vial at the location given by the parameter of the event.

EE 4020 Initialization failed.

EE 4026 Cannot find metering sensor.
The metering device failed to find its home position.

EE 4027 Motor overtemp.
One of the four sample transport motors got high temperature. The number of the motor is given by the parameter of the event:
0 for X-axis
1 for Z-axis
2 for Gripper rotation (Theta)
3 for Gripper finger

EE 4028 Initialization with vial in gripper.
The instrument cannot initialize with a vial in gripper.

EE 4032 Safety flap missing.
The needle covering safety flap is missing.

EE 4033 Vial in Gripper.
The gripper arm failed to put a vial back to its place. The vial is still held in the gripper finger.

EE 4035 No wash vial.
The vial specified as wash vial is not on the sample tray. The parameter of the event is the vial number of the wash vial.

EE 4042 Invalid vial position.
The parameter of the event is the vial number.

28.6.2. ALS State Change Events

Injection Statemachine:

ES 4007 Injecting.
This event indicates, that a state change to INJECTING occured.

ES 4008 No Injecting.
This event indicates, that a state change to NO NJECTING occured.

ES 4036 Break Injection.
A state change to BREAK INJECTION is made when a controller sends SUSP during injection.

ES 4037 Resume Injection.
A state change to RESUME INJECTION is made when a controller sends RESU while the instrument is in the state BREAK INJECTION during an injection.

Initialization Statemachine:

ES 4022 Init.
This event indicates, that a state change to INITIALIZING occured.

ES 4023 Ready.
This event indicates, that a state change to READY occured.

Service Mode Statemachine:

ES 4046 Service Mode.
A state change to SERVICE MODE is made when a controller starts one of the maintenance functions: change needle, change piston or change arm.

ES 4047 No Service Mode.
This event is sent when one of the above service function is finished.

Overlapped Injection Statemachine:

ES 4048 No Overlap Injection.

ES 4049 Overlap Injection Active
The time is running until Overlap Injection begins.

ES 4050 Overlap Injection Running.

ES 4051 Overlap Injection Ready.
Injector program finished

ES 4053 Prefetch Sample Vial.

28.6.3. ALS Counter Limit Events

EF 4038 Needle movement cycles exceeded.
The number of "needle into seat" movements exceeded the limit.

EF 4039 Valve switching cycles exceeded.
The number of "valve to mainpass" switchings exceeded the limit.

28.6.4. ALS Info Events

EV 4010 Start Draw.
This event indicates, that drawing sample starts now.

EV 4011 End Draw.
This event indicates, that drawing sample is finished.

EV 4012 Start Eject.
This event indicates, that ejecting sample starts now.

EV 4013 End Eject.
This event indicates, that ejecting sample is finished.

EV 4021 Tray Changed.
The sample tray has changed or was removed. The parameter of the event describes the type of the new tray (see TRAY:TYPE? on page 523).

EV 4024 No Start While Init.
This event indicates, that a controller tried to start an analysis during initialization.

EV 4025 Vial Changed.
This event indicates, that the number of the sample vial has changed (see VIAL on page 483). The parameter of the event shows the new vial number.

EV 4030 Door Open.
This event indicates, that the front door was opened.

EV 4031 Door Closed.
This event indicates, that the front door was closed.

EV 4034 Missing Vial.
This event indicates, that the sample vial for injection was missing. The parameter of the event holds the vial number. This event is related to the setting of MIVI on page 477.

EV 4040 Motor Warm.
This event indicates, that the motor specified by the event's parameter gets high temperature.
The motors are:0 for X-axis
1 for Z-axis
2 for Gripper rotation (Theta)
3 for Gripper finger

EV 4041 Motor Ok.
This event indicates, that the motor temperature is ok.

EV 4043 No start while service mode.
This event indicates that an attempt was made to start an analysis while the module was in service mode.

EV 4044 No start while "not ready".
This event indicates that an attempt was made to start an analysis while the module was in "not ready" state.

EV 4045 No start while "defect".
This event indicates that an attempt was made to start an analysis while the module was in "defect" state.

EV 4052 Next vial number set.
Used in overlap injection mode.

EV 4054 Firmware internal
(diff between plunger pos and moved steps after home)

EV 4055 Firmware internal
(last needle pos in steps, before needle up)

EV 4056 Firmware internal
(moved steps after needle up)

28.6.5. ALS Injector Program Events

EI 4001 Tray Command Done.
This event indicates, that an instruction for a movement of the sample transport has successfully finished. The parameter of the event gives information about the kind of movement: 0 for TRAY:MOVE
1 for TRAY:PICK
2 for TRAY:PUT

EI 4016 Command Finished.
This event is sent at the end of each instruction.

EI 4029 Step Injector Program.
This event indicates the begin of the execution of an instruction.

28.7. Thermostatted Autosampler Events

28.7.1. Thermostatted Autosampler Error Events

EE 4109 Fan defective.
One of the five fans does not work. The parameter of the event gives the fan #:
1 .. 4 means one of the four heat-sink fans
5 is the fan blowing the air stream to sample vials

EE 4110 Temperature control failed.
One of the three blocks of peltier elements draws overcurrent. The parameter of the event specifies the block: 1..3

EE 4111 Temperature sensor defective.
One of six temperature sensors is defective. The parameter of the event gives the number of the sensor:
1 sensor for temperature controlled air blown to sample vials
2 peltier sensor heat exchange left
3 peltier sensor heat-sink left
4 peltier sensor heat exchange right
5 peltier sensor heat-sink right
6 auxiliary temperature sensor

EE 4112 Lost contact to thermostat module.
Autosampler cannot talk to the thermostat module. Possible reason: rear cable disconnected.

EE 4113 Power failed for thermostat module.
The raw voltage for the thermostat module is low. Possible reason: power cord disconnected, fuse defect or auto range fault (didn't switch to 110V).

EE 4114 Temperature out of range.
One of the peltier sensors shows a temperature out of range >65°C or < -3°C

EE 4120 Bad cooling/heating performance.
This error occures when after giving a new temperature setpoint the actual temperature doesn’t change properly.

28.7.2. Thermostatted Autosampler State Change Events

Temperature Control Statemachine:

ES 4101 Heating up.
This event indicates that the thermostat module is heating up sample vials.

ES 4102 Cooling down.
This event indicates that the thermostat module is cooling down sample vials.

ES 4103 At set-point.
This event indicates that temperature reached the set-point.

Operating Mode Statemachine:

ES 4104 Turned on.
This event indicates that the thermostat module is turned on.

ES 4105 Turned off.
This event indicates that the thermostat module is turned off.

ES 4106 Not Connected.
This event indicates that no thermostat module is connected to the autosampler.

ES 4119 Standby.
Thermostat in standby mode.

28.7.3. Thermostatted Autosampler Info Events

EV 4107 Condensation.
This event indicates that there is condensation in the thermostat module.

EV 4108 No condensation.
This event indicates that condensation is gone in the thermostat module.

EV 4121 Sensor ok.
This event indicates that the temperature sensor which was reported as defective (see EE 4111) is working again.

EV 4122 Ice Protect.
The ice-protect mechanism was started to prevent freezing.

EV 4123 Ice Protect off.
The ice-protect mechanism has finished.

28.7.4. Thermostatted Autosampler Counter Limit Events

EF 4115 Peltier On/Off cycles exceeded.

EF 4116 Peltier on-time exceeded.

EF 4117 Peltier load factor.

EF 4118 Peltier total energy consumption.

28.8. Detector Common Events

Following events are common for the Agilent 1200 Series VWD (G1314B, G1314C), DAD (G1315A/B/D), DAD-SL (G1315C), MWD (G1365A/B/D), MWD-SL (G1365C), FLD (G1321A) and SPM (G8453A).

28.8.1. Detector Lamp related Events

UV-Lamp Statemachine: LampSTM

ES 7400 UV lamp ignition
start of lamp ignition
ES 7403 UV lamp off
UV lamp switched off
ES 7404 UV lamp on
UV lamp is switched on
ES 7406 UV lamp warm up (only VWD)
UV lamp is on, but warming up to be stable (only VWD)

VIS-Lamp Statemachine: VISLampSTM

ES 1023 VIS lamp: ON
VIS lamp has been switched on
ES 1024 VIS lamp: OFF
VIS lamp has been switched off

Lamp Error Events

EE 7450 UV lamp: no current
no anode current during normal operation
EE 7451 UV lamp: no voltage
no anode voltage during normal operation
EE 7452 Ignition failed
no anode current sensed just after UV- lamp ignition during ignition cycle
EE 7453 No heater current
no heater current sensed during UV- lamp ignition cycle
EE 1019 VIS lamp: no voltage
no lamp voltage during normal operation
EE 1018 VIS lamp: no current
no lamp current during normal operation

Lamp Info Events

EV 7401 UV lamp: retry ignition
if the ignition of the D2-lamp failed (no lamp current or voltage sensed) the module automatically tries to ignite the lamp for a second time after a delay of 30 seconds and generates this event.

28.8.2. Detector Info Events

Simulated Chromatogramm

EV 7210 Prepared for send simulated Chromatogramm
Now at next run the detector will send a simulated chromatogramm instead of a real measured chromatogramm
EV 7211 Start sending a simulated Chromatogramm
Now at this run the detector will send a simulated chromatogramm instead of a real measured chromatogramm.
This event is sent out to together with the start-run event.
EV 7212 End sending a simulated Chromatogramm
Now the detector finished sending a simulated chromatogramm,

This event can be sent together with a stop or with disable simulated chromatogramm.

Others

EV 7372	DAC test started event is sent out if DAC test is started

28.8.3. Detector Limit Events

EF 7470	UV- lamp life-time exceeded
EF 7471	VIS- lamp life-time exceeded (only MWD, DAD and SPM)
EF 7472	UV- lamp on-time exceeded
EF 7473	VIS- lamp on-time exceeded (only MWD, DAD and SPM)
EF 7474	UV-lamp ignition cycles exceeded
EF 7475	VIS- lamp ignition cycles exceeded (only MWD, DAD and SPM)
EF 7550	DECS Sample Diode Exposure Counter limit exceeded (only VWD)
EF 7551	DECR Reference Diode Exposure Counter limit exceeded (only VWD)
EF 1100	u-Slit movement counter limit exceeded (only MWD and DAD)
EF 1101	Holmium-oxyde filter movement counter limit exceeded (only MWD and DAD)
EF 6750	Flash Lamp on-time mmmsince installation (only FLD)
EF 6751	Phote Multiplier on-time (only FLD)

28.9. MWD (G1365B) and DAD (G1315B) Events

28.9.1. Error Events

EE 1026 Deuterium lamp overcurrent
The measured current of the deuterium lamp was too high. As a result the deuterium lamp is switched off.

EE 1027 Tungsten lamp overcurrent
The measured current of the tungsten lamp was too high. As a result the tungsten lamp is switched off.

EE 1034 DSP error
fatal communication error between MAIN processor and DSP processor

EE 1036 setting of wavelength calibration failed
reading the spectrometer calibration data out of the spectrophotometer ROM failed or calibration data invalid

EE 1041 Diode leakage
dark current measured during instrument startup is not within specifications

EE 1050 Holmium filter test failed
Execution of the holmium filter test failed.
Moving the filter into the light path is monitored by the detector signal itself. If this signal does not show the expected response (e.g light pass blocked, lamp(s) switched off or holmium filter drive defective), this error message is generated.

EE 1070 Module type changed
During power ON, the detector recognized a new configuration.
Only G1315B and G1365B have built-in hardware for temperature control. The type of the module changed from A to B or vice versa because the hardware for the temperature control was detected / not detected.
The parameter of the event determines the new moduletype:
0 means G1315 no temperature hardware detected
1 means G1315 temperature hardware detected
2 means G1365 no temperature hardware detected
3 means G1365 temperature hardware detected

EE 1071 Illegal temperature value fom sensor at fan assembly.
This temperature sensor delivered a value outside the allowed range. The parameter of this event equals the measured temperature in 1/100 centigrades. As a result the temperature control is switched off.

EE 1072 Illegal temperature value fom sensor at air inlet.
This temperature sensor delivered a value outside the allowed range. The parameter of this event equals the measured temperature in 1/100 centigrades. As a result the temperature control is switched off.

EE 1073 Heater at fan assembly failed.
Every time the deuterium lamp or the tungsten lamp is switched ON or OFF, a heater selftest is performed. If the test fails an error event is gerated. As a result the temperature control is switched off..

EE 1900 DSP not running.
no communication between MAIN processor and DSP processor.

EE 7461 Cover open violation
Lamp(s) turned off because top foam is removed or because lamp(s) cannot be turned on lamp(s) while top foam is removed.
The processor periodically monitors if the top foam is in place. If removed, the fan and the lamp(s) are switched off for security reasons and this error message is generated. The message is also generated, if the lamp(s) cannot be turned on automatically (e.g. during startup or by Auto-On method) while the top foam is removed.

28.9.2. State Change Events

ES 1012	Detector : Reset begin of detector reset phase (initiated by RSET instruction)
ES 7200	Detector: PREPARE start of balance cycle: meas dark current, calculate gain, store background scan.
ES 7090	Detector : IDLE device specific state changed to idle (no prepare,no test,no reset)

28.9.3. Info Events

EV 1043	Data invalid at least for some diodes of the array the data produced are invalid due to too much light at a diode or a gain setting, which is too high. May affect signal and spectra data
EV 1044	Prepare aborted a previously started prepare cycle as been aborted.
EV 0211	Signal runbuffer full detector runbuffer for signal data is full, storing of signal data stopped
EV 1039	Spectra runbuffer overflow loss of one or more spectra because detector runbuffer was full
EV 1037	Wavelength recalibration lost spectrometer not REcalibrated (no data available, e.g. coldstart)
EV 1074	Heater power at limit The available power of the heater reached either the upper or lower limit. This event is sent only once per run. The parameter determines which limit has been hit: 0 means upper power limit hit (excessive ambient temperature drop). 1 means lower power limit hit (excessive ambient temperature increase).
EV 1160	Intensity test started
EV 1161	Dark current test started
EV 1162	Wavelength calibration test started
EV 1163	Test scan started
EV 1164	Holmium-oxyde filter test started
EV 1165	Micro-slit test started

28.10. VWD Events

28.10.1. VWD State Change Events

Device Statemachine: DeviceSTM

ES 7020 Detector Reset
begin of detector reset phase (initiated by RSET instruction)

ES 7050 Scan
detector starts scan (sample, blank, holmium, lamp-charact (sample, ref)

ES 7060 TEST:WCAL started
wavelength calibration algorithm started at detector by TEST:WCAL

ES 7070 WHOL started
wavelength check command WHOL was started at detector

ES 7090 Device idle
device specific state changed to idle (no prepare,no test,no reset, no scan)

ES 7200 Prepare
start of balance cycle: calculate gain, zero baseline

28.10.2. VWD Error Events

Frontend

EE 7800 EE_GRAT_COARSE_CONFIRM_FAIL
before starting grating motor initialization,
grating coarse step driving pulse can't be recognized by processor TPU
unrecoverable error

EE 7801 EE_GRAT_FINE_CONFIRM_FAIL
before starting grating motor initialization,
grating fine step driving pulse can't be recognized by processor TPU
unrecoverable error

EE 7802 EE_GRAT_PRIMARY_CLEAR_FAIL
at the start point of grating motor initialization,
grating drive can't clear its position sensor
unrecoverable error

EE 7803 EE_GRAT_SENSOR_MISSING
during grating motor initialization,
grating position sensor can't be found in 360deg search
unrecoverable error

EE 7804 EE_GRAT_SENSOR_CLEAR_FAIL
on grating motor initialization,
grating position sensor once found but can't be cleared afterwards
unrecoverable error

EE 7805 EE_GRAT_SENSOR_DISAPPEARED
on grating motor initialization,
grating position sensor once found and cleared but can't be found after 360deg turn
unrecoverable error

EE 7806 EE_GRAT_SENSOR_FINE_FAIL
at the final procedure of grating motor initialization,
grating position sensor can't be cleared with fine step
unrecoverable error

EE 7808 EE_GRAT_TEST_ORIGIN_MISSING
during the self test after power on,
grating reference position can't be found in the search window
power off and on may recover the situation

EE 7809 EE_GRAT_TEST_WRONG_ORIGIN
during the self test after power on,

grating position sensor can't be found at the expected position
power off and on may recover the situation

EE 7810 EE_FILT_STEP_CONFIRM_FAIL
before starting filter motor initialization,
filter driving pulse can't be recognized by processor TPU
unrecoverable error

EE 7811 EE_FILT_SENSOR_CLEAR_FAIL
at the start point of filter motor initialization,
filter drive can't clear its position sensor
unrecoverable error

EE 7812 EE_FILT_SENSOR_MISSING
during filter motor initialization,
filter position sensor can't be found in 360deg
unrecoverable error

EE 7813 EE_FILT_TEST_FAIL
on self test after lamp ignition,
cut filter doesn't decrease the light intensity at 250 nm
unrecoverable error

EE 7814 EE_FILT_TEST_ORIGIN_MISSING
during the self test after power on,
filter reference position can't be found in the search window
power off and on may recover the situation

EE 7815 EE_FILT_TEST_WRONG_ORIGIN
during the self test after power on,
filter position sensor can't be found at the expected position
power off and on may recover the situation

EE 7830 EE_ADC_SMP_COMM_FAIL
(not used)
communication error between sample ADC and FPGA

EE 7831 EE_ADC_REF_COMM_FAIL
(not used)
communication error between reference ADC and FPGA

Others

EE 7203 Prepare Error
prepare / balance algorithm finished with error
gain cannot be proper set, for example to much light
check cell

EE 7310 Wavelength calibration setting failed
wavelength calibration setting by WCAL failed, adjust is not set
parameter 0: zero order setting failed, 1:656nm setting failed

EE 7312 Wavelength calibration algorithm failed (sample)
searching for wavelength calibration peaks by TEST:WCAL failed
light peaks was not found at sample diode
-> inspect cell, clean and align cell well
then inspect sample diode and optical unit
parameter 0: zero order calib failed, 1:656nm calib failed

EE 7315 Wavelength calibration algorithm failed (reference)
searching for wavelength calibration peaks by TEST:WCAL failed
light peaks was not found at reference diode
-> align beam splitter in optical unit, inspect reference diode
parameter 0: zero order calib failed, 1:656nm calib failed

EE 7316 Wavelength calibration algorithm failed (misalign)
searching for wavelength calibration peaks by TEST:WCAL failed
light peak was found at sample and reference diode
but both values are misaligned

-> align beam splitter in optical unit
parameter 0: zero order calib failed, 1:656nm calib failed

EE 7321 Flash writing or allocating problem after Calibration
The result of the calibration cannot be write to flash so after next power cycle the calibration will be lost

EE 7327 No Peaks find at Test Wavelength Compensation
During Calibration the detector find no reference lines at holmium filter or at deuterium lamp.

EE 7328 Find to much Deviation at Test Wavelength Compensation
During Calibration the detector find lines at holmium filter or at deuterium lamp, but to far away from expected range.

EE 7330 Test terminated with error
indicates that the last started test terminated with error after starting the Grating Motor Test os the Filter Motor Test this error can be appeared, if the test failed

28.10.3. VWD Info Events

Frontend

EV 7852 EV_GRAT_POS_BELOW_LIMIT
on grating stepping request, the grating position below the lower limit was requested then no stepping has been made

EV 7880 EV_ADC_NREADY_DETECT
on data acquisition, unexpected not ready set by ADC chip is detected

EV 7881 EV_ADC_UNDERFLOW_DETECT
on data acquisition, data underflow set by ADC chip is detected

EV 7882 EV_ADC_OVER_WITH_FIX_GAIN
on PREP with fixed gain, after setting a requested gain, data overflow

EV 7883 EV_ADC_OVER_AT_MIN_GAIN
at the beginning of PREP procedure, data overflow with setting at minimum gain

EV 7884 EV_ADC_OVER_WITH_AUTO_GAIN
on PREP with auto gain, and after setting a proper gain data overflow

EV 7890 wavelength test failed (EV_WL_TEST_FAIL)
on the self test after lamp ignition, D2 emission line can't be found at 656 nm on reference photo diode re-calibration required!

EV 7891 EV_WL_ALIGNMENT_ERROR

EV 7892 Wavelength Calibration Out of Range (EV_WL_PARAM_OUT_OF_RANGE)
wavelength calibration parameters was out of range, the parameter indicates the type (0..2) of wrong calibration parameters.

Others

EV 7011 Memory Problem
memory problem during initializing phase (switch off detector)

EV 7051 Scan Started
a scan is started at the detector, the scantype is given at the parameter of this info event (0..4)

EV7171 No Start while Reset
detector don’t accept start, because reset-state is active

EV7173 No Start while Prepare
detector don’t accept start, because prepare-state is active

EV7174 No Start while Scan
detector don’t accept start, because scan-state is active

EV 7204 Prepare Twice
prepare is executed twice times in order, one executing is omitted

EV 7311 Wavelength calibration setting terminated normal
setting of wavelength calibration by WCAL terminated ok, adjust is done
parameter 0: zero order set ok, 1:656nm set ok

EV 7313 Wavelength calibration algorithm terminated normal
searching for wavelength calibration peaks by TEST:WCAL terminated ok
parameter 0: zero order calib ok, 1:656nm calib ok

EV 7318 Wavelength holmium check failed
wavelength calibration check with holmium oxide filter failed

EV 7319 Wavelength holmium check terminated normal
the wavelength calibration check with holmium oxide filter terminated normal

EV 7331 Test terminated normal
indicates that the last started test terminated normal
after starting the Grating Motor Test os the Filter Motor Test
this event is end out if the test terminated with ok

EV 7340 Peakdetector peak detected
indicates that the peakdetector has detected a peak (only one event/peak)

EV 7350 Spectrum taken error
indicates that the last started scan to take a spectrum terminated with error

EV 7351 Spectrum taken finished
indicates that the last started scan to take a spectrum finished normal

EV 7352 Blank Spectrum not available
A sample spectrum acquisition is requested by the user,
but there is no blank spectrum available

EV 7353 Blank Spectrum mismatch
A sample spectrum acquisition is requested by the user,
but the existing blank spectrum don't match the range of the sample spectrum
example a blank spectrum from 300 to 600nm doesn't match to a sample spectrum from 200 to 400nm, the range of the blank spectrum must be greater or equal than the sample spectrum

EV 7371 ADC overflow
detector A-D-Converter overflow occured

EV 7382 Output Signal Set has changed
The Signal set that are output at digital path of the detector
has changed with the RAWS intruction

EV 7390 Test Calibration Aborted
the calibration measurement was aborted

EV 7391 Holmium Check Aborted
the holmium check measurement was aborted

EV 7392 Scan Aborted
the last started scan was aborted

EV 7401 UV lamp igniton retry
retry of lamp ignition

EV 7510 Internal Communication aborted
because of a reset the internal communication was aborted

28.11. FLD Events

28.11.1. FLD State Change Events

Lamp Statemachine: LampSTM (page 735)

ES 7403	Lamp OFF Lamp is off.
ES 7404	Lamp ON Lamp is on.

Multi Wavelength Statemachine: MultiWavelengthSTM (page 736)

ES 6650	M_WL_Mode_Init Initialization of grating motors
ES 6651	M_WL_Mode_Pos Both gratings are in position mode
ES 6652	M_WL_Mode_EX Multi excitation mode
ES 6653	M_WL_Mode_EM Multi emission mode
ES 6654	M_WL_Mode_OFF grating(s) is (are) not in a stable mode

PMT Calculation Statemachine: PMTCalcSTM (page 737)

ES 6659	pmt calculation calculating the most appropriate pmt
ES 6660	no pmt calculation

Aquisition Mode Statemachine: AcqModeSTM (page 738)

ES 6671	Acq_Mode_Fluor Fluorescence mode
ES 6672	Acq_Mode_Phos Phoshorescence mode
ES 6673	Acq_Mode_Chem Chemoluminescence mode

Lamp Economy Mode Statemachine: LecoSTM (page 739)

ES 6680	Leco_OFF Economy mode is off.
ES 6681	Leco_ON Economy mode is on.

Lamp Energy Reference Mode Statemachine: LrefSTM (page 740)

ES 6685	Lref_ON Lamp energy reference is on.
ES 6686	Lref_OFF Lamp energy reference is off.

Prepare Mode Statemachine: PrepSTM (page 741)

ES 6687	Prepare Preparing
ES 6688	No preprare Not preparing.

Calibration Statemachine: CalibSTM (page 742)

ES 6689	Calib During calibration procedure

ES 6690	Calib idle Calibration procedure not active
ES 6691	Calib lost No calibration values available

3D Scan Statemachine: Scan_3dSTM (page 743)

ES 6695	3D-scan in progress a 3D-scan has been started
ES 6696	no 3D-scan in progress

28.11.2. FLD Error Events

EE 6700	EE_EX_MOT_ERROR excitation motor has failed
EE 6701	EE_EM_MOT_ERROR emission motor has failed
EE 6702	EE_NO_CELL cell has been removed during analysis
EE 6703	EE_CALIB_FAILED calibration procedure has failed
EE 6704	EE_FLD_FLASH_OVERCURR flash lamp overcurrent detected
EE 6705	EE_FLD_EX_MOT_NO_ENCODER excitation encoder not found
EE 6706	EE_FLD_EM_MOT_NO_ENCODER emission encoder not found
EE 6707	EE_FLD_EX_MOT_NO_INDEX excitation encoder index not found
EE 6708	EE_FLD_EM_MOT_NO_INDEX emission encoder index not found
EE 6709	EE_FLD_EX_MOT_FRICTION excitation motor friction too high
EE 6710	EE_FLD_EM_MOT_FRICTION emission motor friction too high
EE 6711	EE_FLD_EX_MOT_NO_POS excitation motor did not find position (position mode)
EE 6712	EE_FLD_EM_MOT_NO_POS emission motor did not find position (position mode)
EE 6713	EE_FLD_EX_MOT_POS_LOST excitation motor lost its position (position mode)
EE 6714	EE_FLD_EM_MOT_POS_LOST emission motor lost its position (position mode)
EE 6715	EE_FLD_EX_MOT_SPEED_LOW excitation motor rotation speed is too low
EE 6716	EE_FLD_EM_MOT_SPEED_LOW emission motor rotation speed is too low
EE 6717	EE_FLD_EX_MOT_SPEED_UNSTABLE excitation motor rotation speed is unstable
EE 6718	EE_FLD_EM_MOT_SPEED_UNSTABLE emission motor rotation speed is unstable
EE 6719	EE_FLD_EX_MOT_INDEX_WRONG excitation encoder index calculation failed (init-phase)
EE 6720	EE_FLD_EM_MOT_INDEX_WRONG emission encoder index calculation failed (init-phase)
EE 6721	EE_FLD_NO_LIGHT_DETECTED no light at reference diode despite lamp is on

EE 6722	EE_FLD_FLASH_TRIGGER_LOST no trigger signal for flash lamp
EE 6730	EE_FLD_FLF_BOARD_NOT_FOUND detector frontend board not found
EE 6731	EE_FLD_LAMPCOVER_OPEN lamp cover is open
EE 6732	EE_FLD_FLF_ADC_NOT_CALIB ADC on frontend board lost calibration

28.11.3. FLD Info Events

EV 6600	EV_FLD_SCAN_WHILE_SCAN a scan could not be started because a scan is already taken
EV 6601	EV_FLD_NO_SCAN_NO_MEM a scan could not be started because no memory for rawd buffers is available
EV 6602	EV_FLD_SCAN_ANGLE_STARTED a angle scan was started
EV 6603	EV_FLD_SCAN_ANGLE_END an angle scan finished
EV 6604	EV_FLD_SCAN_FAILED a scan failed due any error
EV 6605, type	EV_CELL_TYPE a new cell type has been detected
EV 6610, step	EV_FLD_CALIB shows actual step of calibration procedure
EV 6611	EV_FLD_CALIB_READY calibration procedure finished normally
EV 6612	EV_FLD_CALIB_ABORT calibration procedure was aborted
EV 6613, pmt	EV_PMT_AVAILABLE the pmt calculation has terminated normally, the new value is available
EV 6614	EV_NO_STRT_WHILE_CALIB the STRT instruction has been rejected due to the calib state
EV 6615	EV_NO_STRT_WHILE_PMT_CALC the STRT instruction has been rejected due to the pmt calculation state
EV 6616	EV_NO_STRT_WHILE_NO_CELL the STRT instruction has been rejected due to a missing cell
EV 6617, chn	EV_FLD_CHANNEL_CHANGED wavelength of channel <chn> changed
EV 6618	EV_FLD_ADC_OVERFLOW the ADC enters the overflow state
EV 6619	EV_FLD_ADC_NO_OVERFLOW the ADC leave the overflow state
EV 6620	EV_NO_STRT_WHILE_MISSING_FLF the STRT instruction has been rejected due to a missing frontend board
EV 6621	EV_NO_STRT_WHILE_ADC_CALIB_FAILED the STRT instruction has been rejected due to calibration of ADC failed
EV 6622	EV_NO_STRT_WHILE_LAMP_COVER_OPEN the STRT instruction has been rejected due to lamp cover is open
EV 6649	EV_FLD_CALIB_SET_PMT calib proc: set pmt

28.11.4. FLD Feedback Events

EF 6750	EF_FLD_LAMP_ON Flash Lamp on-time since installation exceeded
EF 6751	EF_FLD_PMT_ON PhotoMultiplier on-time exceeded
EF 6752	EF_FLD_LAMP_LIFETIME relative Flash Lamp lifetime exceeded
EF 6760	EF_CELL_NO_CELL cell not installed
EF 6761	EF_CELL_STANDARD standard cell installed
EF 6762	EF_CELL_CUVETTE cuvette installed
EF 6763	EF_CELL_3 cell type 3 installed (not used so far)

28.12. RI Detector Events

28.12.1. RID Error Events

EE 2680 thermal fuse blown
The fuse at the wire to the optical unit heater has blown.
EE 2681 heater resistance high
The heater foil resistance is to high, probably not connected or broken cable.
EE 2682 heater fuse blown
The electronic fuse for the heater circuitry has turned off the heater power.
EE 2683 wrong temperature profile
The heat up monitoring of the optical unit temperature indicates a defective heater circuitry
EE 2684 wrong temperature signal
The ADC signal for the optical unit temperature indicates a wrong tempsignal
EE 2685 maximum allowed temperature exceeded
Turned off heater voltage for savety reasons.
EE 2686 purge valve fuse blown
The electronic fuse of the purge valve turn off the valve power.
EE 2687 recycling valve fuse blown
The electronic fuse of the recycling valve turn off the valve power.
EE 2688 purge valve not connected

EE 2689 recycle valve not connected

EE 2690 lamp voltage too high

EE 2691 lamp current too high

EE 2692 a wait function timed out

EE 2693 lamp voltage too low

EE 2694 lamp current too low

28.12.2. RID State Change Events

Purge Statemachine:

ES 2600 purge on
ES 2601 purge off

Heater Statemachine:

ES 2602 heater turned on
ES 2603 heater turned off

Recycle Statemachine:

ES 2604 recycle on
ES 2605 recycle off

28.12.3. RID Info Events

EV 7211 the instruction SIMU:STRT has been received and a run has been started

EV 7212 run after SIMU:STRT has terminated

28.12.4. RID Counter Limit Events

Counter limit events are related to a Diagnosis Counter (see seperate section). A counter consists of mainly three parts: the total counter value, a presettable counter value and a limit. Once the presettable counter value exceeds the limit the associated event is generated with the parameter 1. If the limit is set to zero the limit verification is off. The event is also generated if either limit or preset value are modified. A parameter of 0 means always that the presettable value of the counter is below the limit, a parameter of 1 means that this value is equal or above the limit. Changes from above to above or from below to below are not reflected as an event.
The name after the event number is the name of the counter the event belongs to.

EF 2670 CREF, Reference valve counter

EF 2671 CREC, Recycle valve counter

EF 2672 HTON, Heater on time

EF 2673 LPOW, Lamp power

EF 2674 HPOW, Heater power

EF 2675 CSRF, Purge valve on time

EF 2676 CSRC, Recycle valve on time

EF 2677 CORF, Purge valve off time

EF 2678 CORC, Recycle valve off time

EF 2679 CALT,Lamp time

28.13. HPLC-Chip Cube Events

28.13.1. HPLC-Chip Cube Error Events

EE 5500 Move inhibit while moving
Cover was detected open, or MS was disconnected while mechanical part was moving. (check lock mechanism, check cables and connectors of SLS-Board)

EE 5501 Chip Tag not found
While processing a LOAD instruction, no valid Chip-Tag was detected. (Make sure a Chip is inserted when loading, check validity of Chip with respect to Hardware and FW. Check cables and connectors)

EE 5502 Operating lever encountered problem while moving.
Parameter:
20: chip load movement failed (reasons: Chip was NOT properly inserted into operating lever causing blockage, electrical short circuit caused by a liquid falsifying the operating lever LOAD-sensor signal, defective sensor or defective motor)
21: chip unload movement failed (reasons: Chip was NOT properly inserted into operating lever causing blockage, electrical short circuit caused by a liquid falsifying the operating lever UNLOAD-sensor signal, defective sensor or defective motor)

EE 5503 Chip withdraw drive (vertical drive) encountered problem while moving.
Parameter:
42/43: Upper/Lower bedstop instable (reasons: misaligned operating lever or other obstacle, check motor and encoder)
44/46: Expected full stroke or operating position was not reached (check motor and encoder, mechanical obstacle?)
41,47,48: A bedstop or target position was not reached, but a blockage was not detected (gearwheel loose ?)
A typical problem is a ribbon cable inside the instrument, interfering with the drive.

EE 5504 Multi-tip drive (horizontal drive) encountered problem while moving.
Parameter:
60: Init failed: drive blocked outside of center position due to mechanical obstacle or sensor defective
61: Init failed: blockage occured while adjusting to center position
62: Init failed: drive blocked at center position or sensor cable disconnected/ sensor defective.

EE 5505 Clamp drive encountered problem while moving.
Parameter:
81,87: A bedstop or target position was not reached, but a blockage was not detected (gearwheel loose ?)
83: Upon init towards sealing position, a blocking was expexted, but not detected (friction to high, motor force to low?)
84: Mismatch of physical clamp stroke and expected stroke (obstacle, chip not correctly loaded or missing chip?)
86,88: Blocking detected while moving or at release from bedstop (obstacle, too high friction, motor defective, encoder defective, cables defective, clamp slider missing?)
90: Sealing failed due to immediate or premature blocking (motor defective, friction too high, seal current too low?)
92: Sealing move: chip rotor touch detection failed (mismatch of parameter and mechanical parts?)

EE 5506 Inner rotor switch to requested target position failed
The requested switch operation of the valve was not successfull.
Parameter:

	1: the valve did not reach the target position within the expected time. The move is aborted, the current state of the valve is undefined. 2: the valve did not switch to the requested target position 3: the FW could not process the actual switch request
EE 5507	Outer rotor switch to requested target position failed The requested switch operation of the valve was not successfull. Parameter: 1: the valve did not reach the target position within the expected time. The move is aborted, the current state of the valve is undefined. 2: the valve did not switch to the requested target position 3: the FW could not process the actual switch request
EE 5508	Trap Pump pressure-wait timed out During moving out of the spray, the time waiting for the pressure of the trap pump to drop below 10 bar exceeded time out value. (Check for blocked capillaries)
EE 5509	Separation Pump ressure-wait timed out During moving out of the spray, the time waiting for the pressure of the separation pump to drop below 10 bar exceeded time out value. (Check for blocked capillaries)
EE 5510	Valve position mismatch A valve position was defined (part of the method), that doesn't match the installed chip's description
EE 5511	Chip type unknown The tag of the chip carries an unknown application key. Either tag corrupted or FW update needed.

28.13.2. HPLC-Chip Cube State Change Events

Chip Position	
ES 5455	Unload position Chip may be inserted into or removed from slot.
ES 5456	Standby position The chip is inside the cube, all hw is initialized, no spray at this position
ES 5457	Operate position The chip's tip is in the position for the spray
Trapping	
ES 5459	No trapping in progress
ES 5460	Trapping in progressf

28.13.3. HPLC-Chip Cube Info Events

EV5403	Configuration of Trap Pump has changed
EV5404	Configuration of Separation Pump has changed
EV5405	New Chip was loaded
EV5406	New Comment entry to Chip Tag
EV5407	Movement for X Adjustment done
EV5408	Movement for Y Adjustment done
EV5409	Start rejected due to missing MS
EV5410	Inject Number for Chip has been incremented
EV5411	Valve calibration run of inner rotor finished
EV5412	No start while not in operate mode
EV5416	Waiting for pressure drop of trapping pump
EV5417	Waiting for pressure drop of separation pump
EV5418	Valve calibration run of outer rotor finished

28.13.4. HPLC-Chip Cube Feedback Events

Counter limit events are related to a Diagnosis Counter (see seperate section). A counter consists of mainly three parts: the total counter value, a presettable counter value and a limit. Once the presettable counter value exceeds the limit the associated event is generated with the parameter 1. If the limit is set to zero the limit verification is off. The event is also generated if either limit or preset value are modified. A parameter of 0 means always that the presettable value of the counter is below the limit, a parameter of 1 means that this value is equal or above the limit. Changes from above to above or from below to below are not reflected as an event.
The name after the event number is the name of the counter the event belongs to.

EF 5480 LOAD, accumulated load time of the chip

EF 5482 OVLV, switches of outer valve

EF 5483 IVLV, switches of inner valve

EF 5485 LDC, number of load cycles

29. Agilent 1200 Series Reply Codes

29.1. CommonReply Codes

29.1.1. Miscellaneous Errors

RA 0000	no error, instruction has been accepted
RA xxxx	no error, <xxxx> = number of listed items
RE 0100	listing active Not all messages of the list CU are retrieved. List CU doesn't generate new listings if old once are pending
RE 0101	not allowed execution of instruction not allowed
RE 0102	busy execution of instruction nott allowed, because being busy
RE 0103	wrong parameter instruction has semantically wrong parameter
RE 0104	full execution not possible because buffer is full (no or not enough resources)
RE 0105	socket not open execution not possible because socket isn't opened yet (Diagnosis-CU and List-CU only)

29.1.2. Method Handling Errors

RE 0200	not existing method doesn't exist
RE 0201	can't overwrite method already exists
RE 0202	class protected method class is protected
RE 0203	internal error method handler internal error
RE 0204	mismatch method components not available or not acceptable
RE 0205	no such entry specified entry doesn't exist

29.1.3. Parameter Handling Errors

RE 0300	end of list end of timetable
RE 0301	begin of list start of timetable
RE 0302	timetable full no more memory for timetable
RE 0304	no timetable timetable is empty
RE 0305	not allowed during run

29.1.4. Parser Errors

RE 0501	syntax syntax error

RE 0502 out of range
paramter out of range
RE 0503 unknown keyword

RE 0504 access for instruction denied

RE 0505 class of instructions already occupied

RE 0506 parameter is locked

RE 0507 class already locked

29.1.5. Monitor Errors

RE 0600 monitor not open
monitor is not open
RE 0601 monitor wrong
monitor wrong confirmed

29.1.6. Rejected Start Commands

RE 0700 run already in progress
run already in progress
RE 0701 no start while leak
no start during leak
RE 0702 no start while shutdown
no start during remote shutdown
RE 0703 no start while startup
no start during startup initialization
RA 0704 analysis already in progress
analysis already in progress
RE 0708 no start while aborting
no start while aborting

29.1.7. Other Errors

RE 0750 xfer failed
xfer failed
RE 0751 prog failed
prog failed
RE 0752 option not installed
option not installed
RE 0770 no load in progress
no load in progress
RE 0771 load in progress
load in progress
RE 0772 not enough memory
not enough memory
RE 0803 cover open violation
attempt to use instruction which is not allowed while cover open
RE 0805 MIO card failed executing command
MIO card failed executing command

29.1.8. Cosy Errors

RE 0900 memory allocation failed

RE 0901	no such COSY-List
RE 0902	COSY-List not running
RE 0903	maximum number of lists exceeded
RE 0904	COSY-List already running
RE 0905	no sync number given
RE 0906	invalid sync number
RE 0907	invalid COSY-List name
RE 0908	unknown identifier
RE 0909	invalid cosy command
RE 0910	list is running
RE 0911	cosy init failed

29.2. Pump Reply Codes

Pumps (binary-/isocratic-/quaternary-/capillary-/nano-):

RE 2001	flow too high for high pressure limit
RE 2030	start rejected while pump off
RE 2031	start rejected while purge in progress
RE 2032	pump on rejected while leak or shutdown or permanent erro or license code
RE 2034	flash allocation failed
RE 2037	STRT while purging
RE 2038	STRT while fast composition change
RE 2039	No calibration table with specified index existing
RE 2040	Not an allowed index for a user-defined table
RE 2041	Specified command allowed in test mode only
RE 2042	Calibration table cannot be deleted while in use
RE 2043	FCC not possible, no partner module connected
RE 2044	FCC active, configuration change not allowed
RE 2045	FCC not allowed in this mode
RE 2046	FCC not possible, no column flow
RE 2047	FSAC base points not specified in descending order
RE 2048	Attempt to specify more than 11 points in FSAC table

Preparative Pump:

RE 2300	no start while pump off
RE 2307	no start while cover open
RE 2308	no start while permanent encoder error
RE 2309	no start while motor over temperature
RE 2310	no start while permanent pressure read-out error
RE 2311	no start while bottle error

29.3. Column Compartment Reply Codes

RE 2801	start not allowed while off
RE 2802	turn on not allowed during startup
RE 2803	turn on not allowed during leak
RE 2804	turn on not allowed while shutdown
RE 2805	calibration setpoint not reached
RE 2806	calibration value not accepted
RE 2807	column tag not present
RE 2808	turn on not allowed without fan

29.4. Autosampler Reply Codes

RE 3001	TRAY command running a previously issued TRAY command is still running and not yet finished
RE 3002	invalid vial number valid vial numbers are in the range [1, 120]
RE 3003	undefined vial position there are no physical coordinates defined for the specified vial number
RE 3004	invalid tray encoding
RE 3005	given coordinates cannot be transformed to a vial position
RE 3006	unknown error
RE 3007	invalid draw/eject speed
RE 3008	invalid injection volume
RE 3009	couldn't move needle
RE 3010	could not move plunger (metering device)
RE 3011	could not move valve
RE 3012	could not move mixer
RE 3013	error during initialization
RA 3014	instruction RESU or SUSP while no injection
RE 3015	instruction NDLE:UP <mm> while needle is in upper position
RE 3016	instruction NDLE:DOWN <mm> while needle is in lowest position
RE 3017	no wash vial
RE 3020	could not move X-axis
RE 3021	could not move Z-axis
RE 3022	could not move Theta
RE 3023	could not move Gripper
RE 3024	one of the four transport motors is too hot
RE 3025	X-sensor defect
RE 3026	Z-sensor defect
RE 3027	T-sensor defect
RE 3028	G-sensor defect
RE 3030	tray decoding sensor defect
RE 3031	needle sensor sensor defect
RE 3032	valve sensor defect
RE 3033	metering sensor defect
RE 3036	start of an analysis during initialization
RE 3037	front door is open
RE 3038	safety flap not in position
RE 3039	vial in gripper (couldn't put vial to tray)
RE 3040	no vial found on tray
RE 3042	allocation of flash memory failed
RE 3043	vial in gripper was lost
RE 3044	a previously issued command is still running and not yet finished
RE 3045	start analysis while module not ready
RE 3046	start analysis while module defect
RE 3047	start analysis while module in service mode
RE 3048	module already in service mode
RE 3049	module not in service mode

29.5. MWD (G1365B) and DAD (G1315B) Reply Codes

RE 6043	No start while prepare
RE 6061	Test not allowed during startup
RE 6062	Test not allowed during reset
RE 6063	Test not allowed while other test is active
RE 6064	Test not allowed during prepare
RE 6071	Test not allowed while leak
RE 6072	Test not allowed while shutdown
RE 1020	Wrong wavelength range Definition of wavelength range exceeds limits (SIGn) or number of data points greater 1024 (WLRG,MWRG)
RE 1021	Prepare while DSP not ok a balance request is rejected while the digital signal processor is in an error state

29.6. VWD Reply Codes

(related files see detector/src/detector.cfg, vwd/src/vwd.cfg, daphne/src/daphne.cfg)

29.6.1. VWD State Reply Errors

RE 6001	prep while startup
RE 6002	prep while reset
RE 6004	prep while prep
RE 6005	prep while scan
RE 6006	prep while calibration test
RE 6007	prep while holmium check
RE 6011	prep while leak
RE 6012	prep while shutdown
RE 6021	prep while ignition
RE 6022	prep while lampoff
RE 6023	prep while lamp warmup
RE 6030	prep not allowed
RE 6041	start while reset
RE 6043	start while prep
RE 6044	start while scan
RE 6061	test while startup
RE 6062	test while reset
RE 6063	test while test
RE 6064	test while prep
RE 6065	test while scan
RE 6066	test while calibrate
RE 6067	test while holmium check
RE 6071	test while leak
RE 6072	test while shutdown
RE 6074	test while run

RE 6075	not allowed during test
RE 6076	test not allowed now
RE 6077	test while abort
RE 6081	this test is not running
RE 6091	scan not allowed while startup
RE 6092	scan not allowed while reset
RE 6094	scan not allowed while prepare
RE 6095	scan not allowed while scan
RE 6096	scan not allowed while calibration test
RE 6097	scan not allowed while holmium check
RE 6101	scan not allowed while lamp ignition
RE 6102	scan not allowed while lamp off
RE 6110	calibration test not allowed while lamp off
RE 6111	calibration test not allowed while startup
RE 6112	calibration test not allowed while reset
RE 6113	calibration test not allowed while test
RE 6114	calibration test not allowed while prepare
RE 6115	calibration test not allowed while scan
RE 6116	calibration test not allowed while calibration
RE 6117	calibration test not allowed while holmium check
RE 6118	calibration test not allowed while lamp stabilize
RE 6120	holmium check not allowed while lamp off
RE 6121	holmium check not allowed while startup
RE 6122	holmium check not allowed while reset
RE 6123	holmium check not allowed while test
RE 6124	holmium check not allowed while prepare
RE 6125	holmium check not allowed while scan
RE 6126	holmium check not allowed while calibration

RE 6127	holmium check not allowed while holmium check

29.6.2. VWD Calibration Reply Errors

RE 6310	calibration set without calibration test
RE 6311	compensation set without compensation test

29.6.3. VWD Spectra Reply Errors

RE 6501	spectrum does not exist
RE 6502	spectra buffer full
RE 6503	spectra buffer empty
RE 6504	take spectrum not allowed
RE 6505	spectrum min/max search inconsistent
RE 6506	blank spectrum not available
RE 6507	blank spectrum range mismatch

29.6.4. VWD Other Reply Errors

RE 6811	unknown signal type

29.7. FLD Reply Codes

RE 5602	C_ER_NOT_IN_M_WL_POS_MODE could not start scan because not being in Pos-Pos (position) mode	
RE 5603	C_ER_NOT_IN_M_WL_EX_MODE could not start scan because not being in Rot-Pos (excitation rotation) mode	
RE 5604	C_ER_NOT_IN_M_WL_EM_MODE could not start scan because not being in Pos-Rot (emission rotation) mode	
RE 5606	C_ER_LAMPNOTONLEAK could not switch on lamp because of leak	
RE 5607	C_ER_LAMPNOTONCHEM could not switch on lamp because of chemoluminescence mode	
RE 5608	C_ER_LAMPNOTONCELL could not switch on lamp because of missing cell	
RE 5609	C_ER_STARTWHILENOCELL could not start analysis because of missing cell	
RE 5610	C_ER_STARTWHILEPMTCALC could not start analysis because Pmt gain is being calculated	
RE 5611	C_ER_STARTWHILECALIB could not start analysis because wavelength calibration is running	
RE 5612	C_ER_CALC_PMT_WHILE_NO_LAMP could not calculate Pmt gain because lamp is off	
RE 5613	C_ER_DIAG_CMD_REJECTED command rejected because not called from cosy list	
RE 5614	C_ER_STARTWHILE_MISSING_FLF could not start analysis because of missing frontend board	
RE 5615	C_ER_STARTWHILE_ADC_CALIB_FAILED could not start analysis because calibration of ADC failed	
RE 5616	C_ER_MISSING_FLF_BOARD could not switch on lamp because missing frontend board	
RE 5617	C_ER_ADC_CALIB_FAILED could not switch on lamp because calibration of ADC failed	
RE 5618	C_ER_LAMP_COVER_OPEN could not switch on lamp because lamp cover is open	
RE 5619	C_ER_STARTWHILE_LAMP_COVER_OPEN could not start analysis because lamp cover is open	
RE 6043	C_ER_STARTWHILEPREP could not start analysis during Prepare	

29.8. RI Detector Reply Codes

`RE 2200`	start while purge
`RE 2201`	RI too low for calib

29.9. HPLC-Chip Cube reply codes

RE 3800	Flash access failed.
RE 3816	Setup read-only.
RE 3823	Trapping not possible while permanent error.
RE 3844	Operation or move is inhibited. The cover is open or MS was not detected.
RE 3855	Instrument detected defective sensor. STRT not possible with defective sensor.
RE 3856	Valve position mismatch. Valve position not supported by current chip.
RE 3857	Instrument not in OPERATE state. STRT not possible while not in OPERATE state.

30. Alphabetic Instruction List

31. Index

Symbols

D

E

F

G

Comments: