ODTC

ODTC Firmware Command Set

Version: 1.09

INHECO GmbH
Fraunhoferstr. 11
82152 Martinsried

INHECO Industrial Heating and Cooling GmbH reserves the right to modify their products for quality improvement. Please note that such modifications may not be documented in this Firmware Command Set.

This Firmware Command Set and the information herein have been assembled with due diligence. **INHECO GmbH** does not assume liability for any misprints or cases of damage resulting from misprints in this manual. If there are any uncertainties, please feel free to contact us.

Contact Data:

Address	Fraunhoferstr. 11 82152 Martinsried Germany
Telephone - Sales	+49 89 899593 101
Fax	+49 89 899593 149
E-Mail - Sales	sales@inheco.com
E-Mail - Technical -Hotline	techhotline@inheco.com
Website	www.inheco.com

History

Datum	Version	Description	Author
2013-03-06	0.00	Initial draft	FOR
2013-03-21	0.01	Basic command set finished	FOR
2013-03-22	0.02	General edit	AWA
2013-03-22	0.03	Scripts	TLA, FOR
2013-08-30	0.04	ODTC related corrections	FOR
2013-10-25	0.05	Corrections: SetParameters Updates for XSD schemas (Appendix)	FOR
2014-01-10	0.06	Updates	FOR, UWI
2014-02-12	0.07	Updates	FOR, MRO
2014-03-28	0.08	Updates for CommandSet and XSDs	FOR
2014-07-21	0.09	Added missing command documentation Added SiLA Return Value chapter	FOR
2014-07-23	0.10	Corrections	FOR
2014-08-22	0.11	Corrections (StopMethod – specific Command)	VLO
2014-10-06	0.12	Added Methods chapter ExecuteMethod DataEvent added Updated MethodSet.xsd v2.0 Corrections	FOR
2014-10-15	0.13	Added SiLA command GetLastData Corrections for SiLA command GetActualTemperature Added SensorValues XSD	FOR
2014-11-03	1.00	Correction SiLA command GetLastData	FOR/VLO
2014-11-25	1.01	Command processing behavior added Changes to Get-/SetConfiguration Changes to Get-/SetParameters	FOR
2014-11-27	1.02	Update MethodSet.xsd	FOR
2014-12-01	1.03	Update MethodSet.xsd Fixed GetParameters: LimitsTempControl Fixed estimated durations and simulation mode infos	FOR
2015-02-04	1.04	Update device specific ErrorCodes Added Internal Warning Code appendix chapter Added SiLA Command Processing States chapter Updated DataEvent structure (GetLastData, ExcecuteMethod)	FOR
2015-08-03	1.05	Update MethodSet.xml Added Get-/SetConfiguration parameter LogLevel	FOR
2016-07-21	1.06	Dynamic Premethod Duration Match Lid Temperature Processing Device Command Behavior	FOR/VLO
2019-10-21	1.07	Build 7235.23748 Added Appendix ODTC Finder	FOR
2020-06-05	1.08	Build 7459.26264 Updated: Appendix Internal Warning Codes	FOR
2021-04-27	1.09	Build 7783.22980 SetParameters minor fixes Added StatusEvent Extensions description	FOR

Table of contents

1 Introduction 4
2 Common Parameter Documentation 4
3 Processing Device Command Behavior 5
4 SiLA Command Processing States 5
5 Return Values 6
5.1 Common Return Codes 6
5.2 Error Classification 6
5.3 Device specific Return Codes 7
5.4 SiLA StatusEvent Extensions 7
6 Methods 8
6.1 MethodSet XML 8
6.2 PreMethods 10
6.3 Methods 11
6.4 Method and PreMethod Deletion 14
7 Device Management 15
7.1 SetConfiguration 15
7.2 GetConfiguration 21
7.3 SetParameters 23
7.4 GetParameters 26
8 Drawer Control 28
8.1 OpenDoor 28
8.2 CloseDoor 29
8.3 PrepareForInput / PrepareForOutput 30
9 Method Control 31
9.1 ExecuteMethod 31
9.2 StopMethod 34
10 Device Sensors 35
10.1 ReadActualTemperature 35
10.2 GetLastData 37
11 Appendix 39
11.1 Internal Warning Codes 39
11.2 ODTC Finder 40

1 Introduction

The INHECO ODTC device implements a SiLA Service Provider according to the SiLA Device Control & Data Interface Specification[1] version 1.2. Beside the Mandatory Commands the ODTC provides a set of Common Commands for the SiLA Device PCR Class (ID 30) and ODTC Device Specific Commands.

Command ID	Mandatory Commands	Required	Optional	Specific
	Device Commands			
Abort	×			
CloseDoor				×
DoContinue	×			
ExecuteMethod			×	
GetConfiguration			×	
GetParameters		×		
GetDeviceIdentification	×			
GetLastData			×	
GetStatus	×			
Initialize	×			
LockDevice	×			
OpenDoor				×
Pause	×			
PrepareForInput		×		
PrepareForOutput		×		
ReadActualTemperature				×
Reset	×			
SetConfiguration			×	
SetParameters		×		
StopMethod				×
UnlockDevice	×			

Every SiLA Service Provider must provide the SiLA Mandatory Command Set. For documentation about SiLA Mandatory Commands the “SiLA Device Control & Data Interface Specification” document is available.

2 Common Parameter Documentation

Every SiLA command specifies at least two parameters: requestId and lockId, expect for the GetStatus command.

requestId
Uniquely generated sequence number by the SiLA Service Consumer for identifying asynchronous answers by a SiLA Service Provider.

lockId
Used for device reservation.

[1] http://www.sila-standard.org/

3 Processing Device Command Behavior

The INHECO ODTC device is able to process some device commands in parallel. Command queuing is not supported. All other commands must be executed in a sequential manor.

The following table shows the possibilities of parallel (P) or sequential (S) processing:

		SP	GP	OD	CD	RAT	EM	SM	GLD
SetParameters	SP	S	S	P	P	P	S	P	S
GetParameters	GP		S	P	P	P	S	P	S
OpenDoor	OD			S	S	P	S	P	P
CloseDoor	CD				S	P	S	P	P
ReadActualTemperature	RAT					S	P	P	P
ExecuteMethod	EM						S	P	S
StopMethod	SM							S	P
GetLastData	GLD								S

4 SiLA Command Processing States

The table shows the ability if a command is executable in the respective SiLA state.

Command ID	SiLA States			
	Startup	Standby	Idle	Busy[1]
Abort			×	×
CloseDoor			×	×
DoContinue			×	×
ExecuteMethod			×	×
GetConfiguration		×		
GetParameters			×	×
GetDeviceIdentification			×	×
GetLastData			×	×
GetStatus			×	×
Initialize		×		
LockDevice		×		
OpenDoor			×	×
Pause			×	×
PrepareForInput			×	×
PrepareForOutput			×	×
ReadActualTemperature			×	×
Reset	×	×	×	×
SetConfiguration		×		
SetParameters			×	×
StopMethod			×	×
UnlockDevice		×		

[1] Command processing ability in state Busy may differ. See → 3 Processing Device Command Behavior

5 Return Values

The SiLA Return Value data type is part of every response sent by the INHECO ODTC device.
Kind of responses supported by the device:

- Synchronous Response
- ResponseEvent, StatusEvent, DataEvent

5.1 Common Return Codes

The following Common Return Codes specified by “SiLA Device Control & Data Interface Specification” are used:

returnCode	Description
1	Success
2	Asynchronous command accepted
3	Asynchronous command has finished
4	Device is busy due to other command execution
5	Error on lockId
6	Error on requestId
9	Command not allowed in this state
11	Invalid command parameter
12	Finished with warning
13	Command unexecuted de-queued due to error

5.2 Error Classification

Type	Description
Warning	Warnings should inform the user about minor device issues while runtime and after processing. Warnings could be ignored by the SiLA PMS.
Error	An error stops current command processing. The device will stay in an operable state.
DeviceError	A device error will result in a non-operable device state. The device will switch into SiLA InError state. Only in case of a communication interrupt it is possible to get into an operable state again by using the Reset command. In all other cases power cycle is needed to get back into an operable state.

5.3 Device specific Return Codes

returnCode	Status	Classification	Correction Hints
1000	InternalSoftwareError	DeviceError	Contact service
2000	Hardware error while operation	DeviceError	Contact service
2001	Connection failure P&CU-ODTC	DeviceError	Check cable connection between P&CU and ODTC
2002	Door error	Error	Check drawer for external blocking
2003	MethodSet error	Error	Correct MethodSet-XML
2004	Command not accepted	Error	Power cycle device
2005	Temperature out of specification	Warning	User should be informed that there is a temperature deviation between set temperature and real temperature
2006	General warning	Warning	Contact service
2007	Main voltage lost	DeviceError	No reaction necessary
2008	Drawer warning	Warning	Check if a disposable is inserted and positioned correctly on the mount.
2009	Ambient temperature is too high	Warning	Check whether air inlet or outlet is blocked.
2010	SDCard	Warning	Insert SDCard correctly and power cycle the device

Further details about Warning information, returnCodes 2005, 2006, 2008, 2009 and 2010 → see chapter 11.1 Internal Warning Codes

5.4 SiLA StatusEvent Extensions

The INHECO ODTC device uses SiLA StatusEvent <Extension> Tags to report internal error code and classification. Generic device driver implementations should not rely on internal codes as these codes may change over device firmware versions. It is recommended to use returnCode value of SiLA ReturnValue from StatusEvent.

Pos	Description	Example Value
1	Error classification	Warning
2	Internal error code (Decimal)	201
3	Internal error code (Hex)	0xC9
4	Internal error code name	OBD_INT_FAN_1
5	Internal error code description	PCU fan 2 (middle) does not reach set speed

6 Methods

Methods are temperature cycle scripts used by the INHECO ODTC device. Methods are defined in a XML structure called MethodSet and can be transmitted from or to the device by using the SiLA commands GetParameters and SetParameters.

6.1 MethodSet XML

A MethodSet is used to manage one or more Methods (scripts) and PreMethods (prerequisite scripts) on the INHECO ODTC device. When using the SetParameters command it is possible to transfer new methods, update existing and delete existing methods. Once a Method or PreMethod is transferred to the device it will be stored at internal flash disc memory. When transferring a new MethodSet its Methods and PreMethods will be merged with already existing Methods and PreMethods stored at the device. It is possible to delete a selection of Methods and PreMethods stored at the device but also it's possible to delete the whole set at once. The MethodSet XML can contain multiple PreMethods, Methods and DeleteMethod/DeleteAllMethods elements at the same time. If you want to read out stored PreMethods and Methods use the command GetParameters (see chapter 7.4)

The INHECO ODTC device is able to store:

- 255 methods at total, with each 100 possible cycle steps and 9 individual PIDSets
- 255 pre-methods at total

Basic MethodSet XML example with a PreMethod and a Method:

```
<MethodSet>
        <PreMethod methodName="method 1" creator="inheco" description="sample pre method"
dateTime="2014-10-06T12:19:26.2418772+02:00">
                <TargetBlockTemperature>20</TargetBlockTemperature>
                <TargetLidTemp>110</TargetLidTemp>
        </PreMethod>
        <Method methodName="method 2" creator="inheco" description="sample method"
dateTime="2014-10-06T12:47:47.4140391+02:00">
                <Variant>960000</Variant>
                <PlateType>0</PlateType>
                <FluidQuantity>0</FluidQuantity>
                <PostHeating>false</PostHeating>
                <StartBlockTemperature>20</StartBlockTemperature>
                <StartLidTemperature>110</StartLidTemperature>
                <Step>
                        <Number>1</Number>
                        <Slope>4.4</Slope>
                        <PlateauTemperature>95</PlateauTemperature>
                        <PlateauTime>60</PlateauTime>
                        <OverShootSlope1>4.4</OverShootSlope1>
                        <OverShootTemperature>5</OverShootTemperature>
                        <OverShootTime>0</OverShootTime>
                        <OverShootSlope2>2.2</OverShootSlope2>
                        <GotoNumber>0</GotoNumber>
                        <LoopNumber>0</LoopNumber>
                        <PIDNumber>1</PIDNumber>
                        <LidTemp>110</LidTemp>
                </Step>
                <Step>
                        <Number>2</Number>
                        <Slope>4.4</Slope>
                        <PlateauTemperature>95</PlateauTemperature>
                        <PlateauTime>10</PlateauTime>
                        <OverShootSlope1>4.4</OverShootSlope1>
                        <OverShootTemperature>5</OverShootTemperature>
                        <OverShootTime>0</OverShootTime>
                        <OverShootSlope2>2.2</OverShootSlope2>
                        <GotoNumber>0</GotoNumber>
                        <LoopNumber>0</LoopNumber>
```

```
                        <PIDNumber>1</PIDNumber>
                        <LidTemp>110</LidTemp>
                </Step>
                <Step>
                        <Number>3</Number>
                        <Slope>2.2</Slope>
                        <PlateauTemperature>65</PlateauTemperature>
                        <PlateauTime>5</PlateauTime>
                        <OverShootSlope1>2.2</OverShootSlope1>
                        <OverShootTemperature>5</OverShootTemperature>
                        <OverShootTime>0</OverShootTime>
                        <OverShootSlope2>2.2</OverShootSlope2>
                        <GotoNumber>0</GotoNumber>
                        <LoopNumber>0</LoopNumber>
                        <PIDNumber>1</PIDNumber>
                        <LidTemp>110</LidTemp>
                </Step>
                <Step>
                        <Number>4</Number>
                        <Slope>4.4</Slope>
                        <PlateauTemperature>72</PlateauTemperature>
                        <PlateauTime>5</PlateauTime>
                        <OverShootSlope1>4.4</OverShootSlope1>
                        <OverShootTemperature>5</OverShootTemperature>
                        <OverShootTime>0</OverShootTime>
                        <OverShootSlope2>2.2</OverShootSlope2>
                        <GotoNumber>2</GotoNumber>
                        <LoopNumber>5</LoopNumber>
                        <PIDNumber>1</PIDNumber>
                        <LidTemp>110</LidTemp>
                </Step>
                <Step>
                        <Number>5</Number>
                        <Slope>4.4</Slope>
                        <PlateauTemperature>95</PlateauTemperature>
                        <PlateauTime>60</PlateauTime>
                        <OverShootSlope1>4.4</OverShootSlope1>
                        <OverShootTemperature>5</OverShootTemperature>
                        <OverShootTime>0</OverShootTime>
                        <OverShootSlope2>2.2</OverShootSlope2>
                        <GotoNumber>0</GotoNumber>
                        <LoopNumber>0</LoopNumber>
                        <PIDNumber>1</PIDNumber>
                        <LidTemp>110</LidTemp>
                </Step>
                <PIDSet>
                        <PID number="1">
                                <PHeating>60</PHeating>
                                <PCooling>80</PCooling>
                                <IHeating>250</IHeating>
                                <ICooling>100</ICooling>
                                <DHeating>10</DHeating>
                                <DCooling>10</DCooling>
                                <PLid>100</PLid>
                                <ILid>70</ILid>
                        </PID>
                </PIDSet>
        </Method>
</MethodSet>
```

6.2 PreMethods

A PreMethod is a prerequisite Method to lift the mount- and the LID-temperature of the device to a defined level. It is necessary to run a PreMethod prior running a Method at the device. Mount- and LID-temperature will stay stable after PreMethod has finished. The device will check the actual target temperatures of the mount before starting a Method. If Method start temperature of the mount does not match, the device will refuse executing. For executing a PreMethod the SiLA command ExecuteMethod is available.
The duration of a PreMethod is calculated by the adjusted LID-temperature. By deactivating the functionality DynamicPreMethodDuration (see chapter 7.3) the runtime of every PreMethod is fixed to 10 minutes. When Premethod is finished the device is ready for executing further commands (e.g. OpenDoor, CloseDoor, ExcecuteMethod).
Several PreMethods/Methods are managed within a MethodSet.

Example XML:

```
<MethodSet>
        <PreMethod methodName="premethod 1" creator="inheco" description="sample pre
method" dateTime="2014-10-06T12:19:26.2418772+02:00">
                <TargetBlockTemperature>20</TargetBlockTemperature>
                <TargetLidTemp>110</TargetLidTemp>
        </PreMethod>
        <PreMethod methodName="premethod 2" creator="inheco" description="sample pre method
2" dateTime="2014-10-06T12:19:26.2418772+02:00">
                <TargetBlockTemperature>40</TargetBlockTemperature>
                <TargetLidTemp>90</TargetLidTemp>
        </PreMethod>
        ...
</MethodSet>
```

6.2.1 Attributes

PreMethod Attributes	
methodName	Unique name of the PreMethod Note: methodName attribute is unique for Methods and PreMethods!
Creator	Name of the staff (optional)
description	Description of the PreMethod (optional)
dateTime	Creation/change date and time in ISO 8601 dateTime format (optional)

6.2.2 Elements

PreMethod Elements	
TargetBlockTemperature	Target temperature of the mount in °C
TargetLidTemp	Target temperature of the LID in °C

6.3 Methods

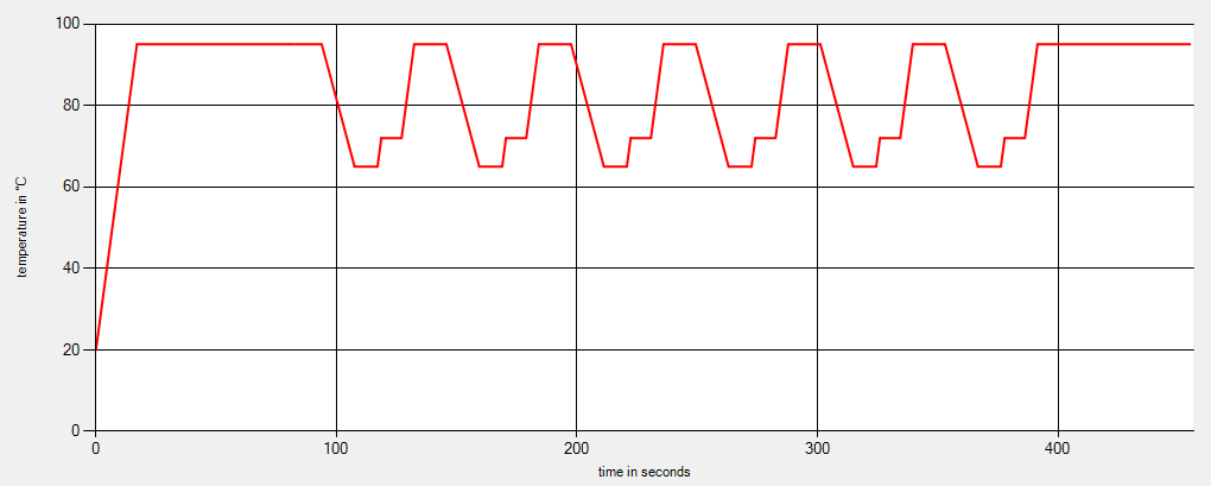
Methods are temperature cycle scripts formatted as XML. Basically a Method defines a temperature profile by defining several temperature plateaus and holding times between each plateau. Each temperature plateau can be reached by different slopes and optionally looped multiple times. For further details how to create a script, please see SkriptEditor Manual.
Methods execution begins at a predefined start mount temperature which must be reached prior by executing a PreMethod. The duration of a Method is dependent of its parameters and can be calculated by adding slope time, overshoot time (not visible in the following Sample Method) and plateau time of each step in consideration of the loops.
Optionally the device can hold the plateau temperature of the last step similar to a PreMethod. When PostTemper is enabled (true). Methods can be combined together without the need of executing a corresponding PreMethod before. The PlateauTemperature of the last step need to be the StartBlockTemperature of the new Method. After Method execution is finished and PostTemper option is enabled the device is ready for executing further commands (e.g. OpenDoor, CloseDoor).
For executing a Method the SiLA command ExecuteMethod is available.

Sample Method defining 5 steps and looping:

Step number	Slope in °C/s	Plateau temp. in °C	Plateau time in s	Goto number	Loop number
1	4.4	95	60	0	0
2	4.4	95	10	0	0
3	2.2	65	5	0	0
4	4.4	72	5	2	5
5	4.4	95	60	0	0

(Start temperature is 20°C)

Resultant temperature curve diagram:



Method rules:

- Max individual Steps: 100
- Max Loops per Step: 100
- Max total Steps (including loops): 100.000
- Max Method duration: 16h
- GoTo number: only step backward is allowed

Example XML:

```
<MethodSet>
        <Method methodName="method 1" creator="inheco" description="sample method"
dateTime="2014-10-06T12:47:47.4140391+02:00">
                <Variant>960000</Variant>
                <PlateType>0</PlateType>
                <FluidQuantity>0</FluidQuantity>
                <PostHeating>false</PostHeating>
                <StartBlockTemperature>20</StartBlockTemperature>
                <StartLidTemperature>110</StartLidTemperature>
                <Step>
                        <Number>1</Number>
                        <Slope>4.4</Slope>
                        <PlateauTemperature>95</PlateauTemperature>
                        <PlateauTime>60</PlateauTime>
                        <OverShootSlope1>4.4</OverShootSlope1>
                        <OverShootTemperature>5</OverShootTemperature>
                        <OverShootTime>0</OverShootTime>
                        <OverShootSlope2>2.2</OverShootSlope2>
                        <GotoNumber>0</GotoNumber>
                        <LoopNumber>0</LoopNumber>
                        <PIDNumber>1</PIDNumber>
                        <LidTemp>110</LidTemp>
                </Step>
                <PIDSet>
                        <PID number="1">
                                <PHeating>60</PHeating>
                                <PCooling>80</PCooling>
                                <IHeating>250</IHeating>
                                <ICooling>100</ICooling>
                                <DHeating>10</DHeating>
                                <DCooling>10</DCooling>
                                <PLid>100</PLid>
                                <ILid>70</ILid>
                        </PID>
                </PIDSet>
        </Method>
        <Method methodName="method 2" creator="inheco" description="sample method 2"
dateTime="2014-10-06T12:47:47.4140391+02:00">
                ...
        </Method>
        ...
</MethodSet>
```

6.3.1 Attributes

Method Attributes	
methodName	Unique name of the Method Note: methodName attribute is unique for Methods and PreMethods!
Creator	Name of staff (optional)
description	Description of the PreMethod (optional)
dateTime	Creation date and time in ISO 8601 dateTime format (optional)

6.3.2 Elements

Method Elements	
Variant	The specified variant (e.g. 960000 for 96 ODTC) must match the hardware variant of the INHECO ODTC device.
PlateType	Manufacturer and/or Type of the Plate
FluidQuantity	Fluid Quantity (default values)
PostHeating	Keeps on regulating the last specified block and lid temperature after method
StartBlockTemperature	Must match TargetBlockTemperature of the prior executed PreMethod or must match the last PlateauTemperature of the prior Method when PostHeating is activated.
StartLidTemperature	If functionality MatchLidTemperatures (see chapter 7.3) is activated, StartLidTemperature Must match TargetLidTemp of the prior executed PreMethod or must match the last LidTemp of the prior Method when PostHeating is activated.
Step	Defines a temperature step.
Step / Number	Order steps to a sequence and mark steps for specifying loops
Step / Slope	Specifies the ramping rate in °C/s
Step / PlateauTemperature	Mount temperature in °C
Step / PlateauTime	Plateau time in s.
Step / OverShootSlope1	Specifies the first overshoot ramping rate in °C/s
Step / OverShootTemperature	Overshoot temperature in °C
Step / OverShootTime	Plateau time of overshoot in seconds
Step / OverShootSlope2	Specifies the second overshoot ramping rate in °C/s
Step / GotoNumber	Starting point of a new loop (must be smaller than actual step number)
Step / LoopNumber	Number of loops
Step / PIDNumber	References a PIDSet PID number. If the number is zero default PID values of the micro controller will be used.
Step / LidTemp	Set LID temperature in °C
PIDSet	Set of PID values used by the controller (defined by INHECO)

6.4 Method and PreMethod Deletion

To delete all Methods and PreMethods stored at device the XML Element <DeleteAllMethods> with value true must be transmitted.

Example XML:

```
<MethodSet>
        <DeleteAllMethods>true</DeleteAllMethods>
</MethodSet>
```

To delete individual Methods and PreMethods stored at device the XML element <DeleteMethods> followed by element <MethodName> must be transmitted.

Example XML:

```
<MethodSet>
        <DeleteMethods>
                <MethodName>method 1</MethodName>
                <MethodName>method 2</MethodName>
                ...
        </DeleteMethods>
</MethodSet>
```

Elements	
DeleteMethods / MethodName	Can be used to choose methods by their unique identifiers for deletion. The DeleteMethods element is optional, it cannot be combined with DeleteAllMethods element at same time.
DeleteAllMethods	When true, deletes all stored methods on the device. The DeleteAllMethods element is optional, it cannot be combined with DeleteMethods element at same time.

7 Device Management

For device management the following commands are available:

- SetConfiguration
  - SysDateTime
  - NetworkConfig
  - SoapCompression
  - UseDeviceClassDateTime
  - LogLevel
- GetConfiguration
  - SysDateTime
  - NetworkConfig
  - SoapCompression
  - UseDeviceClassDateTime
  - LogLevel
- SetParameters
  - MethodsXML
  - ExecuteMethodDataEvent
  - DynamicPreMethodDuration
  - MatchLidTemperatures
  - CSVSeparatorCharacter
- GetParameters
  - MethodsXML
  - ExecuteMethodDataEvent
  - DynamicPreMethodDuration
  - MatchLidTemperatures
  - CSVSeparatorCharacter

7.1 SetConfiguration

The SetConfiguration command is used to set the configuration of a device. This command can only be executed in the SiLA state Standby.

Estimated Duration	5s
Simulation Mode	Not supported Changes made by SetConfiguration are applied persistently.

7.1.1 Parameters

Name	Data type	Restrictions	Default	Nullable
requestId	Int32	[1; 2.147.483.647]		False
lockId	String	XML-characters		True
configLevel	Int32	[1; 2.147.483.647]		True
password	String			True
configXML	String (XMLDocument)			False

7.1.2 ConfigLevel

Parameter configLevel is not used.

7.1.3 Password

Parameter password is not used.

7.1.4 ConfigXML

Parameter configXML contains configuration parameters formatted as XML. It has to be composed according to SiLA → ResponseType_1.2.xsd XML schema file. The possible XML root elements are <ParameterSet> for transmitting at least one or multiple configuration parameters or <Value> for a single parameter.

```
<ParameterSet>
        <Parameter parameterType="xxx" name="yyy">
                <xxx>zzz</xxx>
        </Parameter>
        ...
</ParameterSet>
```

```
<Value parameterType="xxx" name="yyy">
        <xxx>zzz</xxx>
</Value>
```

7.1.4.1 Writable Parameters

Parameter name	Type
SysDateTime	DateTime
NetworkConfig	String (XMLDocument)
SoapCompression	Boolean
UseDeviceClassDateTime	Boolean
LogLevel	String

7.1.4.2 SysDateTime

Setting the system date, time and time zone of the INHECO ODTC device in XML DateTime format.

For further information about the XML type DateTime see:
→ http://www.w3.org/TR/xmlschema-2/#dateTime

7.1.4.3 NetworkConfig

It's possible to switch between the two Ipv4 network operation modes automatic and static IP configuration. By default the INHECO ODTC device is configured to automatic IP configuration by a DHCP server. A device power cycle is required to apply new network settings.

7.1.4.3.1 Static IP Configuration

Static (manual) IP configuration requires IP- and Subnet-Address. The default Gateway- and DNS-Address are optional.

Example XML:

```
<Network>
	<StaticIPv4>
		<IP>10.0.1.129</IP>
		<Subnet>255.255.255.128</Subnet>
		<Gateway>10.0.1.254</Gateway>
		<DNS>10.0.0.1</DNS>
		<DNS>10.0.0.2</DNS>
	</StaticIPv4>
</Network>
```

7.1.4.3.2 Automatic IP configuration

A Dynamic Host Control Protocol (DHCP) server is required for automatic (dynamic) IP assignment.

Example XML:

```
<Network>
	<DHCP/>
</Network>
```

7.1.4.3.3 NetworkConfig XML schema definition

Network.xsd:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="Network" xmlns:xs="http://www.w3.org/2001/XMLSchema" version="1.0">
	<xs:element name="Network" type="NetworkType" />
	<xs:complexType name="NetworkType">
		<xs:sequence>
			<xs:choice>
				<xs:element name="DHCP" type="DHCPIPv4Type" minOccurs="1"/>
				<xs:element name="StaticIPv4" type="StaticIPv4Type"
minOccurs="1" />
			</xs:choice>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="StaticIPv4Type">
		<xs:sequence>
			<xs:element name="IP" type="IPtype" minOccurs="1" maxOccurs="1"/>
			<xs:element name="Subnet" type="IPtype" minOccurs="1"
maxOccurs="1"/>
			<xs:element name="Gateway" type="IPtype" minOccurs="0"
maxOccurs="1"/>
			<xs:element name="DNS" type="IPtype" minOccurs="0"
maxOccurs="unbounded"/>
		</xs:sequence>
	</xs:complexType>
		<xs:complexType name="DHCPIPv4Type">
		<xs:sequence>
			<xs:element name="IP" type="IPtype" minOccurs="0" maxOccurs="1"/>
			<xs:element name="Subnet" type="IPtype" minOccurs="0"
maxOccurs="1"/>
			<xs:element name="Gateway" type="IPtype" minOccurs="0"
maxOccurs="1"/>
			<xs:element name="DNS" type="IPtype" minOccurs="0"
maxOccurs="unbounded"/>
		</xs:sequence>
	</xs:complexType>
	<xs:simpleType name="IPtype">
		<xs:restriction base="xs:string">
			<xs:pattern value="(([2][0-5][0-5])|([0-1][0-9][0-
9])|\d{2}|\d{1})\.(([2][0-5][0-5])|([0-1][0-9][0-9])|\d{2}|\d{1})\.(([2][0-5][0-5])|([0-
1][0-9][0-9])|\d{2}|\d{1})\.(([2][0-5][0-5])|([0-1][0-9][0-9])|\d{2}|\d{1})"/>
		</xs:restriction>
  </xs:simpleType>
</xs:schema>
```

7.1.4.4 SoapCompression

Parameter name	Type	Restrictions	Default
SoapCompression	Boolean		False

A device power cycle is required to apply SoapCompression configuration.

Enables or disables HTTP GZIP compression support of INHECO ODTC device.

True: Compression support enabled
False: Compression support disabled

GZIP compression must also be supported by the SiLA PMS to work. A request of the SiLA Service Consumer must indicate GZIP support in the HTTP Header. This is done by “Accept-Encoding” header with GZIP flag. Only if both the SoapCompression support of INHECO ODTC device is enabled and the client sends the GZIP supported flag, GZIP compressed synchronous response (HTTP response) will be sent.

Receiving compressed HTTP requests is not supported by INHECO ODTC device.
HTTP GZIP compression of SiLA Events is not supported by INHECO ODTC device.

Example HTTP request:

```
POST / HTTP/1.1
Content-Type: text/xml; charset=utf-8
SOAPAction: "http://sila.coop/Reset"
Host: 10.1.22.4:8080
Content-Length: 476
Expect: 100-continue
Accept-Encoding: gzip, deflate

<s:Envelope              xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"><s:Body><Reset
xmlns="http://sila.coop"                          xmlns:i="http://www.w3.org/2001/XMLSchema-
instance"><requestId>519394860</requestId><lockId    i:nil="true"/><deviceId>56aeb91e-68ae-
4659-a688-
5ff9f1667516</deviceId><eventReceiverURI>http://10.1.22.100:7071/ihc</eventReceiverURI><PM
SId>http://10.1.22.100:7071/ihc</PMSId><errorHandlingTimeout
i:nil="true"/><simulationMode>false</simulationMode></Reset></s:Body></s:Envelope>
```

Example compressed HTTP response:

```
HTTP/1.1 200 OK
Server: gSOAP/2.8
X-Frame-Options: SAMEORIGIN
Content-Type: text/xml; charset=utf-8
Content-Encoding: gzip
Transfer-Encoding: chunked
Connection: close

128
.............n.0.E.H...{b....#....Th..e...d&.8....m..E..<.......P...*v.31.G".4d..f.}.8....
....z.z.HV...J....t.Ld....l2(4..f.eL.N...?.....?.w.3z`w......S.d4......i..C..
.......
.f`(6Tt...]..C.P..=.=...7`.a(....:.mY..+\..5I.U.........h...j.B.B...1Pz.q.u...u.}xI...7.w.
Z....2..jz2/....'o. ..P.|
u7..
```

(Content: 523 Bytes)

Example uncompressed HTTP response:

```
HTTP/1.1 200 OK
Server: gSOAP/2.8
X-Frame-Options: SAMEORIGIN
Content-Type: text/xml; charset=utf-8
Content-Encoding: gzip
Transfer-Encoding: chunked
Connection: close

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"  xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"                    xmlns:i="http://inheco.com"
xmlns:s="http://sila.coop"><SOAP-
ENV:Body><s:ResetResponse><s:ResetResult><s:returnCode>2</s:returnCode><s:message>Asynchro
nous                                                                                command
accepted.</s:message><s:duration>PT1S</s:duration><s:deviceClass>8</s:deviceClass></s:Rese
tResult></s:ResetResponse></SOAP-ENV:Body></SOAP-ENV:Envelope>
```

(Content: 795 Bytes)

7.1.4.5 UseDeviceClassDateTime

Parameter name	Type	Restrictions	Default
UseDeviceClassDateTime	Boolean		False

Enables or disables usage of parameter “deviceClass” of SiLA data type “SiLAReturnValue” as Date and Time parameter. If enabled the Device Class of type Int32 will be used as a Unix-Time value of the current device time. Unix time is defined as the number of seconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970.

True: Enabled
False: Disabled (Use device specific device class)

Important Note

Enabling this option may cause incompatibility to SiLA drivers. Check if your SiLA driver relies on correct SiLA Device Class values of SiLAReturnValue.

7.1.4.6 LogLevel

Sets the log level of internal logging system for diagnostic purposes of device software. Do not change this value until advised by INHECO support personal.

Valid LogLevel values (case-insensitive):

- DEBUG
- INFO
- WARN
- ERROR
- FATAL

Default value: INFO

7.1.4.7 Examples

- setting systems date and time
- setting static IP configuration
- setting 'DEBUG' log level

(Parameter configXML formatted as ParameterSet)

```
<ParameterSet>
        <Parameter parameterType="DateTime" name="SysDateTime">
                <DateTime>2014-01-01T15:20:00Z</DateTime>
        </Parameter>
        <Parameter name="NetworkConfig">
                <String>&lt;Network&gt;&lt;StaticIPv4&gt;&lt;IP&gt;10.0.1.129&lt;/IP&gt;&lt;
                Subnet&gt;255.255.255.128&lt;/Subnet&gt;&lt;Gateway&gt;10.0.1.254&lt;/Gatewa
                y&gt;&lt;DNS&gt;10.0.0.1&lt;/DNS&gt;&lt;DNS&gt;10.0.0.2&lt;/DNS&gt;&lt;/Stat
                icIPv4&gt;&lt;/Network&gt;</String>
        </Parameter>
         <Parameter name="SoapCompression">
             <Boolean>false</Boolean>
         </Parameter>
         <Parameter name="UseDeviceClassDateTime">
             <Boolean>false</Boolean>
         </Parameter>
        <Parameter name="LogLevel">
                <String>DEBUG</String>
        </Parameter>
</ParameterSet>
```

- setting automatic IP configuration

(Parameter configXML formatted as Value)

```
<Value name="NetworkConfig">
        <String>&lt;Network&gt;&lt;DHCP/&gt;&lt;/Network&gt;</String>
</Value>
```

7.1.5 ResponseData

No response will be expected, therefore only the XML frame will be send:

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData                          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd"/>
```

7.2 GetConfiguration

The GetConfiguration command is used to receive the configuration of a device. It can only be invoked in the standby state.

Estimated Duration	5s
Simulation Mode	Not supported GetConfiguration delivers applied configuration.

7.2.1 Parameters

Name	Data type	Restrictions	Default	Nullable
requestId	Int32	[1; 2.147.483.647]		False
lockId	String	(XML-characters)		True
configLevel	Int32	[1; 2.147.483.647]		True
password	String			True

7.2.2 ConfigLevel

Parameter configLevel is not used.

7.2.3 Password

Parameter password is not used.

7.2.4 ResponseData

The response contains configuration parameters formatted as XML. It is composed according to SiLA → ResponseType_1.2.xsd XML schema file. The XML root element is <ResponseData>. Followed by the element <ParameterSet> which contains the element <Parameter> for each configuration parameter.

```
<?xml version="1.0"?>
<ResponseData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd">
        <ParameterSet>
                <Parameter name="xxx">
                        <yyy>zzz</yyy>
                </Parameter>
                ...
        </ParameterSet>
</ResponseData>
```

7.2.4.1 Readable Parameters

Parameter name	Type
SysDateTime	DateTime
NetworkConfig	String (XMLDocument)
SoapCompression	Boolean
UseDeviceClassDateTime	Boolean
LogLevel	String

7.2.4.2 SysDateTime

The INHECO ODTC device system date and time.
→ 7.1.4.2 SysDateTime

7.2.4.3 NetworkConfig

The INHECO ODTC device network configuration.
→ 7.1.4.3 NetworkConfig

7.2.4.4 SoapCompression

→ 7.1.4.4 SoapCompression

7.2.4.5 UseDeviceClassDateTime

→ 7.1.4.5 UseDeviceClassDateTime

7.2.4.6 LogLevel

The INHECO ODTC device detail level of internal software logging system.
→ 7.1.4.6 LogLevel

7.2.4.7 Example

Response contains ResponseData XML and parameters formatted as ParameterSet:

- Device system date and time
- Device network configuration
  - Network mode: automatic IP
  - Current IP settings assigned by DHCP server

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd">
        <ParameterSet>
                <Parameter name="SysDateTime">
                        <DateTime>2014-01-01T00:00:00Z</DateTime>
                </Parameter>
                <Parameter name="NetworkConfig">
                        <String>&lt;Network&gt;&lt;DHCP&gt;&lt;IP&gt;10.0.0.65&lt;/IP&gt;&lt
                        ;Subnet&gt;255.255.255.192&lt;/Subnet&gt;&lt;Gateway&gt;10.0.0.126&l
                        t;/Gateway&gt;&lt;DNS&gt;10.0.0.1&lt;/DNS&gt;&lt;DNS&gt;10.0.0.2&lt;
                        /DNS&gt;&lt;/DHCP&gt;&lt;/Network&gt;</String>
                </Parameter>
                <Parameter name="SoapCompression">
                     <Boolean>false</Boolean>
                </Parameter>
                <Parameter name="UseDeviceClassDateTime">
                     <Boolean>false</Boolean>
                </Parameter>
                <Parameter name="LogLevel">
                        <String>INFO</String>
                </Parameter>
        </ParameterSet>
</ResponseData>
```

7.3 SetParameters

The SetParameters command is used to set device specific parameters within the state idle so that the new values are used for the next command invocation. Only the parameters that need to be changed have to be transmitted to the device.

Estimated Duration	60s
Simulation Mode	Partially supported Parameters transmitted aren't verified and won't be applied.

7.3.1 Parameters

Name	Data type	Restrictions	Default	Nullable
requestId	Int32	[1; 2.147.483.647]		False
lockId	String	(XML-characters)		True
paramsXML	String (XMLDocument)			False

7.3.2 ParamsXML

Parameter paramsXML contains configuration parameters formatted as XML. It has to be composed according to SiLA → ResponseType_1.2.xsd XML schema file. The possible XML root elements are <ParameterSet> for transmitting at least one or multiple configuration parameters or <Value> for a single parameter.

```
<ParameterSet>
        <Parameter parameterType="xxx" name="yyy">
                <xxx>zzz</xxx>
        </Parameter>
        ...
</ParameterSet>
```

```
<Value parameterType="xxx" name="yyy">
        <xxx>zzz</xxx>
</Value>
```

7.3.2.1 Writable Parameters

Parameter name	Type
MethodsXML	String (XMLdocument)
ExecuteMethodDataEvent	Boolean
DynamicPreMethodDuration	Boolean
MatchLidTemperatures	Boolena
CSVSeparatorCharacter	String (Character)

7.3.2.2 MethodsXML

Setting cycle scripts formatted as XML.
See → 6 Methods

7.3.2.3 ExecuteMethodDataEvent

Sending SiLA DataEvents while processing device command ExecuteMethod can be enabled or disabled. By default this setting is enabled (true).

- True: enabled DataEvents
- False: disabled DataEvents

7.3.2.4 DynamicPreMethodDuration

The duration of the PreMethod will be calculated by the adjusted Lid-temperature. By default this setting is enabled (true).

- True: Dynamic calculation of PreMethod duration is enabled
- False: PreMethod duration is set to 10 minutes

7.3.2.5 MatchLidTemperatures

To execute a new Method the actual mount temperature need to match the StartBlockTemperature of the new Method (see chapter 6.3). Compliance of Lid temperatures can be set by using the MatchLidTemperatures. By default this setting is disabled (false).

- True: If Lid temperature of the Method do not match current temperature, execution is denied.
- False: no verification of Lid temperatures

7.3.2.6 CSVSeparatorCharacter

Setting a single separator character symbol used for internal generated CSV data. By default the character is set to semicolon ";". Changing this symbol has an impact to device command GetLastData.

7.3.2.7 Examples

- setting MethodsXML

```
<ParameterSet>
	<Parameter parameterType="String" name="MethodsXML">
		<String>&lt;MethodSet&gt;&lt;Method methodName=&quot;method 1&quot;
		creator=&quot;inheco&quot; description=&quot;sample method&quot;
		dateTime=&quot;2014-11-
		25T15:19:01.747465+01:00&quot;&gt;&lt;Variant&gt;960000&lt;/Variant&gt;&lt;P
		lateType&gt;0&lt;/PlateType&gt;&lt;FluidQuantity&gt;0&lt;/FluidQuantity&gt;&
		lt;PostHeating&gt;false&lt;/PostHeating&gt;&lt;StartBlockTemperature&gt;20&l
		t;/StartBlockTemperature&gt;&lt;StartLidTemperature&gt;110&lt;/StartLidTempe
		rature&gt;&lt;Step&gt;&lt;Number&gt;1&lt;/Number&gt;&lt;Slope&gt;4&lt;/Slope
		&gt;&lt;PlateauTemperature&gt;44&lt;/PlateauTemperature&gt;&lt;PlateauTime&g
		t;4&lt;/PlateauTime&gt;&lt;OverShootSlope1&gt;4&lt;/OverShootSlope1&gt;&lt;O
		verShootTemperature&gt;5&lt;/OverShootTemperature&gt;&lt;OverShootTime&gt;0&
		lt;/OverShootTime&gt;&lt;OverShootSlope2&gt;2.2&lt;/OverShootSlope2&gt;&lt;G
		otoNumber&gt;0&lt;/GotoNumber&gt;&lt;LoopNumber&gt;0&lt;/LoopNumber&gt;&lt;P
		IDNumber&gt;1&lt;/PIDNumber&gt;&lt;LidTemp&gt;110&lt;/LidTemp&gt;&lt;/Step&g
		t;&lt;Step&gt;&lt;Number&gt;2&lt;/Number&gt;&lt;Slope&gt;2&lt;/Slope&gt;&lt;
		PlateauTemperature&gt;22&lt;/PlateauTemperature&gt;&lt;PlateauTime&gt;2&lt;/
		PlateauTime&gt;&lt;OverShootSlope1&gt;2&lt;/OverShootSlope1&gt;&lt;OverShoot
		Temperature&gt;5&lt;/OverShootTemperature&gt;&lt;OverShootTime&gt;0&lt;/Over
		ShootTime&gt;&lt;OverShootSlope2&gt;2.2&lt;/OverShootSlope2&gt;&lt;GotoNumbe
		r&gt;1&lt;/GotoNumber&gt;&lt;LoopNumber&gt;10&lt;/LoopNumber&gt;&lt;PIDNumbe
		r&gt;1&lt;/PIDNumber&gt;&lt;LidTemp&gt;110&lt;/LidTemp&gt;&lt;/Step&gt;&lt;P
		IDSet&gt;&lt;PID
		number=&quot;1&quot;&gt;&lt;PHeating&gt;60&lt;/PHeating&gt;&lt;PCooling&gt;8
		0&lt;/PCooling&gt;&lt;IHeating&gt;250&lt;/IHeating&gt;&lt;ICooling&gt;100&lt
		;/ICooling&gt;&lt;DHeating&gt;10&lt;/DHeating&gt;&lt;DCooling&gt;10&lt;/DCoo
		ling&gt;&lt;PLid&gt;100&lt;/PLid&gt;&lt;ILid&gt;70&lt;/ILid&gt;&lt;/PID&gt;&
		lt;/PIDSet&gt;&lt;/Method&gt;&lt;/MethodSet&gt;</String>
	</Parameter>
</ParameterSet>
```

- setting new CSV separator character ","

- disable DataEvents while ExecuteMethod

```
<ParameterSet>
	<Parameter name="CSVSeparatorCharacter">
		<String>,</String>
	</Parameter>
	<Parameter name="ExecuteMethodDataEvent">
		<Boolean>false</Boolean>
	</Parameter>
</ParameterSet>
```

7.3.3 ResponseData

No response will be expected, therefore only the XML frame will be send:

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData                          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd"/>
```

7.4 GetParameters

The GetParameters command is used to report actual parameter names and values to the PMS.

Estimated Duration	60s The duration depends on the amount of Methods and PreMethods stored on the device.
Simulation Mode	Partially supported GetParameters delivers an empty ParameterSet.

7.4.1 Parameters

Name	Data type	[Min; Max]	Default	Nullable
requestId	Int32	1; 2.147.483.647]	-	False
lockId	String	(XML-characters)	-	True

7.4.2 ResponseData

The response contains parameters formatted as XML. It is composed according to SiLA → ResponseType_1.2.xsd XML schema file. The XML root element is <ResponseData>. Followed by the element <ParameterSet> which contains the element <Parameter> for each configuration parameter.

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd">
        <ParameterSet>
                <Parameter name="xxx">
                        <yyy>zzz</yyy>
                </Parameter>
                ...
        </ParameterSet>
</ResponseData>
```

7.4.2.1 Readable Parameters

Parameter name	Type
MethodsXML	String (XMLdocument)
ExecuteMethodDataEvent	Boolean
DynamicPreMethodDuration	Boolean
MatchLidTemperatures	Boolean
CSVSeparatorCharacter	String (Character)

7.4.2.2 MethodsXML

Getting cycle scripts formatted as XML.
See → 6 Methods

7.4.2.3 ExecuteMethodDataEvent

Enable or disable sending SiLA DataEvents while processing device command ExecuteMethod.
See → 7.3.2.3 ExecuteMethodDataEvent

7.4.2.4 DynamicPreMethodDuration

Enable or disable DynamicPreMethodDuration.
See → 7.3.2.4 DynamicPreMethodDuration

7.4.2.5 MatchLidTemperature

Enable or disable MatchLidTemperature.
See → 7.3.2.5 MatchLidTemperature

7.4.2.6 CSVSeparatorCharacter

Getting the CSV separator symbol.
See → 7.3.2.6 CSVSeparatorCharacter

7.4.2.7 Example

Response contains ResponseData XML and parameters formatted as ParameterSet:

- Cycle scripts MethodsXML
- Sending ExecuteMethodDataEvent
- Setting DynamicPreMethodDuration
- Setting MatchLidTemperatures
- CSV separator character

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd">
        <ParameterSet>
                <Parameter name="MethodsXML">
                        <String>&lt;?xml version="1.0" encoding="utf-
                        8"?&gt;&lt;MethodSet&gt;&lt;DeleteAllMethods&gt;false&lt;/DeleteAllM
                        ethods&gt;&lt;PreMethod methodName="pre 1" creator="inheco"
                        dateTime="2014-01-
                        01T00:00:00"&gt;&lt;TargetBlockTemperature&gt;40&lt;/TargetBlockTemp
                        erature&gt;&lt;TargetLidTemp&gt;110&lt;/TargetLidTemp&gt;&lt;/PreMet
                        hod&gt;&lt;/MethodSet&gt;</String>
                </Parameter>
                <Parameter name="ExecuteMethodDataEvent">
                        <Boolean>true</Boolean>
                </Parameter>
                <Parameter name="DynamicPreMethodDuration">
                        <Boolean>true</Boolean>
                </Parameter>
                <Parameter name="MatchLidTemperatures">
                        <Boolean>false</Boolean>
                </Parameter>
                <Parameter name="CSVSeparatorCharacter">
                        <String>;</String>
                </Parameter>
        </ParameterSet>
</ResponseData>
```

8 Drawer Control

For drawer control the following commands are available:

- OpenDoor
- CloseDoor
- PrepareForOutput
- PrepareForInput

Note:
SiLA PrepareForOutput and PrepareForInput both commands closes the drawer of the INHECO ODTC device.

8.1 OpenDoor

The OpenDoor command opens the drawer of the INHECO ODTC device where it can accept the next labware item. The physical opening drawer process is not interruptible or stoppable by any SiLA Command. When SiLA command Pause is invoked the SiLA SubState will change to state pauseRequested and remain until processing is finished and then change to state asynchPaused.

Estimated Duration	14s
Simulation Mode	Supported

8.1.1 Parameters

Name	Data type	[Min; Max]	Default	Nullable
requestId	Int32	1; 2.147.483.647]	-	False
lockId	String	(XML-characters)	-	True

8.1.2 ResponseData

No response will be expected, therefore only the XML frame will be send:

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData                          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd"/>
```

8.2 CloseDoor

The CloseDoor command closes the drawer of the INHECO ODTC device. The physical closing drawer process is not interruptible or stoppable by any SiLA Command. When SiLA command Pause is invoked the SiLA SubState will change to state pauseRequested and remain until processing is finished and then change to state asynchPaused.

Estimated Duration	14s
Simulation Mode	Supported

8.2.1 Parameters

Name	Data type	[Min; Max]	Default	Nullable
requestId	Int32	1; 2.147.483.647]	-	False
lockId	String	(XML-characters)	-	True

8.2.2 ResponseData

No response will be expected, therefore only the XML frame will be send:

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData                          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd"/>
```

8.3 PrepareForInput / PrepareForOutput

The PrepareForInput / PrepareForOutput commands closes the drawer of the INHECO ODTC device. The physical closing drawer process is not interruptible or stoppable by any SiLA Command. When SiLA command Pause is invoked the SiLA SubState will change to state pauseRequested and remain until processing is finished and then change to state asynchPaused.

Estimated Duration	14s
Simulation Mode	Supported

8.3.1 Parameters

Name	Data type	[Min; Max]	Default	Nullable
requestId	Int32	[1; 2.147.483.647]		False
lockId	String	XML-characters		True
position	Int32	[1; 2.147.483.647]		False

8.3.2 Position

Parameter position value is ignored.

8.3.3 ResponseData

No response will be expected, therefore only the XML frame will be send:

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData                          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd"/>
```

9 Method Control

To start or stop a temperature script (= Method) the following commands are available:

- ExecuteMethod
- StopMethod

9.1 ExecuteMethod

The ExecuteMethod command is used to start a previously defined method with the device control software. A method is uniquely determined by its identifier “methodName”. A physical method process is not interruptible by the SiLA Mandatory Command Pause. When SiLA command Pause is invoked the SiLA SubState will change to state PauseRequested and remain until process is finished and then change to state AsynchPaused.
Methods can be transferred to the ODTC device by using the SetParameters command, available methods can be looked up with the GetParameters command. For further details about methods see → 6 Methods.

Estimated Duration	Variable PreMethod: 10min Method: depends on cycle script
Simulation Mode	Partially supported PreMethod and Method have fixed duration of 10s.

9.1.1 Parameters

Name	Data type	[Min; Max]	Default	Nullable
requestId	Int32	[1; 2.147.483.647]		False
lockId	String	(XML-characters)		True
methodName	String			False
priority	Int32	[1; 2.147.483.647]		True

9.1.2 MethodName

The parameter specifies the unique name of the method to execute. The name can either specify a PreMethod or a Method.

9.1.3 Priority

Parameter priority is not used.

9.1.4 ResponseData

No response will be expected, therefore only the XML frame will be send:

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData                          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd"/>
```

9.1.5 Data Event

ExecuteMethod sends continuously SiLA DataEvents while executing. These Events includes temperature sensor values for possible live monitoring by a PMS system. The data is formatted according to the "SiLA Data Capture Specification"[2] version 1.0 which is also known as a SiLA Complex Response data package.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd">
    <AnyData>
        &lt;?xml version=&quot;1.0&quot; encoding=&quot;utf-8&quot;?&gt;
        &lt;containerData xmlns:xsi=&quot;http://www.w3.org/2001/XMLSchema-instance&quot;
xsi:noNamespaceSchemaLocation=&quot;DataCapture_1.0.xsd&quot; version=&quot;1.0&quot;&gt;
            &lt;experimentStep name=&quot;Method&quot; sequence=&quot;1&quot;
type=&quot;Measurement&quot;&gt;
                &lt;dataSeries nameId=&quot;Elapsed time&quot; description=&quot;&quot;
unit=&quot;ms&quot; repeat=&quot;0&quot;&gt;
                    &lt;seriesValueSet&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;190&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;1286&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;2314&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;3335&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;4355&lt;/integerValue&gt;
                    &lt;/seriesValueSet&gt;
                &lt;/dataSeries&gt;
                &lt;dataSeries nameId=&quot;Target temperature&quot;
description=&quot;&quot; unit=&quot;1/100°C&quot; repeat=&quot;0&quot;&gt;
                    &lt;seriesValueSet&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;2142&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;2000&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;1780&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;1580&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;1566&lt;/integerValue&gt;
                    &lt;/seriesValueSet&gt;
                &lt;/dataSeries&gt;
                &lt;dataSeries nameId=&quot;Current temperature&quot;
description=&quot;&quot; unit=&quot;1/100°C&quot; repeat=&quot;0&quot;&gt;
                    &lt;seriesValueSet&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;2139&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;2064&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;1931&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;1791&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;1656&lt;/integerValue&gt;
                    &lt;/seriesValueSet&gt;
                &lt;/dataSeries&gt;
                &lt;dataSeries nameId=&quot;LID temperature&quot; description=&quot;&quot;
unit=&quot;1/100°C&quot; repeat=&quot;0&quot;&gt;
                    &lt;seriesValueSet&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;2368&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;2374&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;2397&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;2423&lt;/integerValue&gt;
                        &lt;integerValue mapId=&quot;&quot;&gt;2463&lt;/integerValue&gt;
                    &lt;/seriesValueSet&gt;
                &lt;/dataSeries&gt;
            &lt;/experimentStep&gt;
            &lt;labwareInfo&gt;
                &lt;standardPlate name=&quot;&quot; identification=&quot;&quot;
labwareTypeId=&quot;&quot; labwareTypeName=&quot;&quot;
labwareDefinitionSource=&quot;Other&quot; /&gt;
            &lt;/labwareInfo&gt;
            &lt;deviceMetadata&gt;
                &lt;software name=&quot;&quot; manufacturer=&quot;&quot;
version=&quot;&quot; /&gt;
                &lt;identification productName=&quot;&quot; productNumber=&quot;&quot;
manufacturer=&quot;&quot; firmwareVersion=&quot;&quot; serialNumber=&quot;&quot; /&gt;
            &lt;/deviceMetadata&gt;
            &lt;methodMetadata&gt;
                &lt;commonProperties&gt;
```

[2] http://www.sila-standard.org

```
                    &lt;MethodStartTime&gt;2014-10-02T19:32:02Z&lt;/MethodStartTime&gt;
                    &lt;MethodName&gt;method 2&lt;/MethodName&gt;
                &lt;/commonProperties&gt;
            &lt;/methodMetadata&gt;
        &lt;/containerData&gt;
    </AnyData>
</ResponseData>
```

9.2 StopMethod

The StopMethod command will stop a processing method. The command can be executed even if the command “ExecuteMethod” is not processing at the moment to stop temperature controlling of a post temper enabled Method process or a PreMethod.

Estimated Duration	1s
Simulation Mode	Supported

9.2.1 Parameters

Name	Data type	[Min; Max]	Default	Nullable
requestId	Int32	[1; 2.147.483.647]		False
lockId	String	(XML-characters)		True

9.2.2 ResponseData

No response will be expected, therefore only the XML frame will be send:

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData                          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd"/>
```

10 Device Sensors

Beside of the DataEvent of SiLA command ExecuteMethod there are two additional ways getting sensor data values of the INHECO ODTC device in an explicit manner.

- ReadActualTemperature
- GetLastData

10.1 ReadActualTemperature

The ReadActualTemperature command is used to get temperature values of the available temperature sensors. This command can be executed while the device is in SiLA state “Idle” but also while in state “Busy” e.g. while executing the SiLA command ExecuteMethod.

Estimated Duration	1s
Simulation Mode	Supported

10.1.1 Parameters

Name	Data type	[Min; Max]	Default	Nullable	Dir
requestId	Int32	[1; 2.147.483.647]	-	False	In
lockId	String	(XML-characters)	-	True	In

10.1.2 ResponseData

The response contains one single parameter formatted as XML. It is composed according to SiLA → ResponseType_1.2.xsd XML schema file. The XML root element is <ResponseData>. Followed by the element <Parameter>.

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
        <Parameter name="xxx">
                <yyy>zzz</yyy>
        </Parameter>
</ResponseData>
```

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
        <Parameter name="SensorValues">
                <String>&lt;SensorValues timestamp=&quot;2014-01-
                01T12:00:00Z&quot;&gt;&lt;Mount&gt;2463&lt;/Mount&gt;&lt;Mount_Monitor&gt;26
                42&lt;/Mount_Monitor&gt;&lt;Lid&gt;2575&lt;/Lid&gt;&lt;Lid_Monitor&gt;2627&l
                t;/Lid_Monitor&gt;&lt;Ambient&gt;2450&lt;/Ambient&gt;&lt;PCB&gt;3308&lt;/PCB
                &gt;&lt;Heatsink&gt;2596&lt;/Heatsink&gt;&lt;Heatsink_TEC&gt;2487&lt;/Heatsi
                nk_TEC&gt;&lt;/SensorValues&gt;</String>
        </Parameter>
</ResponseData>
```

10.1.3 ResponseData parameter

Parameter name	Type
SensorValues	String (XMLdocument)

10.1.3.1 SensorValues

The SensorValues XML document contains temperature sensor values.

Example:

```
<SensorValues timestamp="2014-07-17T23:38:36Z">
        <Mount>2463</Mount>
        <Mount_Monitor>2642</Mount_Monitor>
        <Lid>2575</Lid>
        <Lid_Monitor>2627</Lid_Monitor>
        <Ambient>2450</Ambient>
        <PCB>3308</PCB>
        <Heatsink>2596</Heatsink>
        <Heatsink_TEC>2487</Heatsink_TEC>
</SensorValues>
```

SensorValues XSD schema:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" id="SensorValues" version="1.0">
        <xs:element name="SensorValues">
                <xs:complexType>
                        <xs:sequence>
                                <xs:element name="Mount" type="xs:short"/>
                                <xs:element name="Mount_Monitor" type="xs:short"/>
                                <xs:element name="Lid" type="xs:short"/>
                                <xs:element name="Lid_Monitor" type="xs:short"/>
                                <xs:element name="Ambient" type="xs:short"/>
                                <xs:element name="PCB" type="xs:short"/>
                                <xs:element name="Heatsink" type="xs:short"/>
                                <xs:element name="Heatsink_TEC" type="xs:short"/>
                        </xs:sequence>
                        <xs:attribute name="timestamp" type="xs:dateTime" use="required"/>
                </xs:complexType>
        </xs:element>
</xs:schema>
```

10.2 GetLastData

The GetLastData command is used to get a temperature trace of the last executed PreMethod or Method.

Estimated Duration	0s (not specified) Duration depends of last executed cycle script length.
Simulation Mode	Supported

10.2.1 Parameters

Name	Data type	[Min; Max]	Default	Nullable	Dir
requestId	Int32	[1; 2.147.483.647]	-	False	In
lockId	String	(XML-characters)	-	True	In

10.2.2 ResponseData

The response contains sensor data formatted as a CSV document. The response XML is formatted according to the “SiLA Data Capture Specification”[3] version 1.0 which is also known as a SiLA Complex Response data package.

```
<?xml version="1.0" encoding="utf-8"?>
<ResponseData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ResponseType_1.2.xsd">
    <AnyData>
        &lt;?xml version=&quot;1.0&quot; encoding=&quot;utf-8&quot;?&gt;
        &lt;containerData xmlns:xsi=&quot;http://www.w3.org/2001/XMLSchema-instance&quot;
xsi:noNamespaceSchemaLocation=&quot;DataCapture_1.0.xsd&quot; version=&quot;1.0&quot;&gt;
            &lt;experimentStep name=&quot;ExecuteMethod&quot; sequence=&quot;0&quot;
type=&quot;Measurement&quot;&gt;
                &lt;dataSeries nameId=&quot;Temperature Trace&quot;
description=&quot;&quot; unit=&quot;CSV&quot; repeat=&quot;0&quot;&gt;
                    &lt;seriesValueSet&gt;
                        &lt;stringValue mapId=&quot;&quot;&gt;Elapsed time [ms];Target
temperature [1/100°C];Current temperature [1/100°C];Lid temperature [1/100°C]
115;2132;2195;2501
1132;2000;2041;2501
2267;2000;2011;2500
3296;2000;2007;2501
4631;2000;2006;2503
5668;2000;2004;2500
6700;2000;2004;2501
7729;2000;2004;2500
8909;2000;2005;2500
9890;2000;2003;2500
10919;2000;2004;2504
11951;2000;2005;2502
13081;2000;2004;2501
14110;2000;2004;2500
15247;2000;2003;2500
16280;2000;2003;2501
17310;2000;2003;2501
18337;2000;2004;2500
19365;2000;2003;2500
20408;2000;2004;2500
21425;2000;2004;2501
22454;2000;2003;2501
23483;2000;2003;2501
24511;2000;2003;2501
25575;2000;2003;2500
26600;2000;2002;2500
27732;2000;2002;2500
28761;2000;2003;2501
```

[3] http://www.sila-standard.org

```
29788;2000;2003;2501
30819;2000;2003;2501
31864;2000;2002;2501
32873;2000;2002;2500
33897;2000;2001;2501
34928;2000;2001;2500
35963;2000;2001;2500
36990;2000;2002;2500
38019;2000;2002;2500
39049;2000;2002;2501
                        &lt;/stringValue&gt;
                    &lt;/seriesValueSet&gt;
                &lt;/dataSeries&gt;
            &lt;/experimentStep&gt;
            &lt;labwareInfo&gt;
                &lt;standardPlate name=&quot;&quot; identification=&quot;&quot;
labwareTypeId=&quot;&quot; labwareTypeName=&quot;&quot;
labwareDefinitionSource=&quot;Other&quot; /&gt;
            &lt;/labwareInfo&gt;
            &lt;deviceMetadata&gt;
                &lt;software name=&quot;&quot; manufacturer=&quot;&quot;
version=&quot;&quot; /&gt;
                &lt;identification productName=&quot;&quot; productNumber=&quot;&quot;
manufacturer=&quot;&quot; firmwareVersion=&quot;&quot; serialNumber=&quot;&quot; /&gt;
            &lt;/deviceMetadata&gt;
        &lt;/containerData&gt;
    </AnyData>
</ResponseData>
```

Note:
In the example above the CSV separator character ";" is being used.
See → 7.3.2.6 CSVSeparatorCharacter

11 Appendix

11.1 Internal Warning Codes

Note: internal codes are not unique but the combination of returnCode and internal Code. Internal Code values may change on ODTC Firmware updates. Do not consider internal codes for driver development!

returnCode	Internal Code	Name	Description
2006	101	ERR_WRONG_EEPROMADDRESS	Unknown EEPROM is addressed (not ODTC, PCU or Calibrator)
2008	118	ERR_LID_CLOSURE	Drawer Lid is not closed correctly. Check whether a disposable is inserted and positioned correctly on the mount.
2006	143	ERR_U_SWITCH	Not yet defined. Reserved for later use. (Output voltage of switching regulator is wrong, e.g. shortcut)
2006	150	OBD_IDRAW_HIGH	Maximum drawer current exceeded
2006	155	OBD_STATUS_HEATFOIL	Drawer-LID-IC: short cut, output disruption, over temperature-IC, power supply to Heated Lid is disabled. Thermal performance of ODTC might be decreased.
2006	156	OBD_STATUS_FAN	Fan-IC: short cut, output disruption, over temperature-IC. Power supply to fan is disabled. Thermal performance of ODTC might be decreased.
2006	157	OBD_STATUS_MOTOR	Drawer motor-IC: short cut, over temperature-IC. Power supply to drawer is disabled.
2006	162	OBD_TEMPSENSOR_2_SHORT	Short cut or break of ambient temperature sensor (Sensor 2). Thermal performance might be decreased.
2006	167	OBD_TEMPSENSOR_7_SHORT	Short cut or break of ODTC board temperature sensor (Sensor 7). Thermal performance of ODTC might be decreased.
2006	171	ERR_TEMP_SENSOR_FILTE2	Too many temperature shifts detected – Area2
2009	188	OBD_MAX_TEMP_AMBIENT	Ambient temperature too high. Check whether air inlet or outlet is blocked. Thermal performance of ODTC might be decreased.
2006	200	OBD_INT_FAN_0	PCU fan 1 (corner) does not reach set speed
2006	201	OBD_INT_FAN_1	PCU fan 2 (middle) does not reach set speed
2005	209	OBD_TEMP_REGULATION1	Deviation between set temperature and measured mount temperature (Sensor 8 / high temperature area D1) is too large during the heating/ cooling ramps. Either the ramp speed was not programmed correctly or the initial configuration (filling level, selection 96 or 384 well version) fits not to the processed plate.
2005	210	OBD_TEMP_REGULATION2	Deviation between set temperature and measured mount temperature (Sensor 8 / low temperature area D2) is too large during the heating/ cooling ramps. Either the ramp speed was not programmed correctly or the initial configuration (filling level, selection 96 or 384 well version) might not fit to the processed plate.
2005	211	OBD_TEMP_REGULATION3	Deviation between set temperature and measured mount temperature (Sensor 8 / high temperature area) is too large on plateau. The initial configuration (filling level, selection 96 or 384 well

			version) might not fit to the processed plate. Thermal performance might be decreased
2005	212	OBD_TEMP_REGULATION4	Vapor Chamber Mount might be damaged. Thermal performance might be decreased.
2005	213	OBD_TEMP_REGULATION5	Not yet defined. Reserved for later use.
2005	217	OBD_LID_CNTRL	Deviation between set temperature and measured LID temperature (Sensor 6) is too large during heating (during cooling this temperature is not checked). Thermal performance might be decreased.
2006	248	ERR_RESERVED_WARNING_1	Reserved for internal INHECO use only
2006	249	ERR_RESERVED_WARNING_2	Reserved for internal INHECO use only
2006	250	ERR_RESERVED_WARNING_3	Reserved for internal INHECO use only
2006	254	ERR_ERRBUFFER_OVERFLOW1	Error buffer overflow - errors may be lost
2010	267	NO_SDCARD	No SD-Card available. Log and Trace files won't be written
2010	268	SDCARD_ERROR	Defective SDCard detected. Log and Trace files won't be written.

For all returnCodes (Errors/DeviceErrors/Warnings) → see chapter 5.3 Device specific Return Codes

11.2 ODTC Finder

For identifying a INHECO ODTC device on network, the ODTC implements a UDP based request response protocol over IPv4. Because the protocol is based on broadcasts the built-in finder is limited to non-routed local networks.

11.2.1 Request

There are two different kind of request patterns:
Variant 1 is the recommended method finding ODTCs on network. Only those ODTCs can be found which have suitable subnet configuration to send unicast response.
Variant 2 sends out broadcast for finding all ODTCs on network. A ODTC will send a broadcast response datagram. Any client can find any ODTC in broadcast domain even a unicast connection isn't possible.

11.2.1.1 Variant 1

Client UDP Port: any (>=1024)
Server UPD Port: 4711
Request Method: Subnet-Broadcast (recommended) or Broadcast
Response Method: Unicast
Response Port: Client Port

Datagram:

0x53 0x69 0x4c 0x41 0x5f 0x46 0x69 0x6e 0x64 0x65 0x72 0x5f 0x52 0x65 0x71 0x75 0x65 0x73 0x74

11.2.1.2 Variant 2

Client UDP Port: any (>=1024)
Server UPD Port: 4711
Request Method: Broadcast
Response Method: Broadcast
Response Port: Client Port

Datagram:

0x53 0x69 0x4c 0x41 0x5f 0x46 0x69 0x6e 0x64 0x65 0x72 0x5f 0x52 0x65 0x71 0x75 0x65 0x73 0x74 0x5f 0x42 0x63

11.2.2 Response

When the ODTC device receives a request pattern of variant 1 or 2 it will send a fixed length response datagram which contains information about the device.

Datagram (total length 134 bytes):

Preamble, 16 bytes: 0x01 0x4f 0x44 0x54 0x43 0x02 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00	
Device Name, 16 bytes, ASCII Null-terminated	
Build Version, 16 bytes, ASCII Null-terminated	
Firmware Version, 32 bytes, ASCII Null-terminated	
Article Number, 4 bytes UInt32 (little endian)	SOAP Service Port, 2 bytes UInt16 (little endian)
Other (Future Use), 25 bytes	DHCP Enabled, 1 byte (1: true / 0: false)
IPv4 Mask, 4 bytes	Network Gateway Address, 4 bytes
Network DNS1 Address, 4 bytes	Network DNS2 Address, 4 bytes
MAC address, 6 bytes	