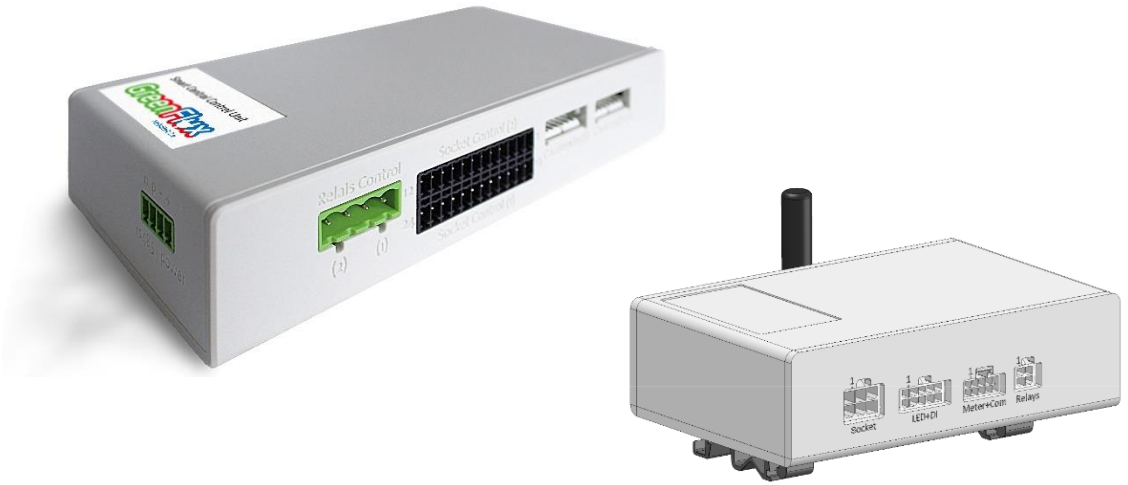
GREENFLUX SMART CHARGING CONTROLLERS



CONFIGURATION MANUAL

DSCU, P/N: T2235-02

SSCU, P/N: 1040300

2019-11-05; version 2.3

The award winning GreenFlux smart charging controller turns every Charging Station into a connected device.

www.greenflux.com

GreenFlux Assets BV

Mauritskade 64

1092AD Amsterdam

The Netherlands

Email: info@greenflux.com

Web: www.greenflux.com

Tel: +31 88 605 0705

CONTENT

1 ABOUT

1.1 GREENFLUX

GreenFlux operates as a technology provider, offering smart charging controllers and a cloud platform (Platform-as-a-Service) for remote management, billing, roaming, customer support and smart charging.

For more information on GreenFlux, visit www.greenflux.com.

1.2 THIS DOCUMENT

This document lists configuration settings and their definitions for the GreenFlux smart charging controllers and may be subject to change after initial publishing.

1.2.1 DOCUMENT VERSION HISTORY

Document version	Date	Controller software version
1.0	November 2017	
1.1	December 2018	4.1.2
2.0	March 2019	4.2.1
2.1	May 2019	4.3.0
2.2	October 2019	4.3.1
2.3	November 2019	4.3.2

1.2.2 PROTOCOLS

This document may at some point refer to the OCPP 1.6 protocol. This protocol can be found and downloaded free of charge at the Open Charge Alliance website, www.openchargealliance.org.

1.3 CONTACT

Internet: www.greenflux.com
Office: Mauritskade 64, 1092AD, Amsterdam, The Netherlands
Phone: +3188 605 0705 (09.00 – 18.00 CET)
Email: info@greenflux.com

Figure 1 DSCU

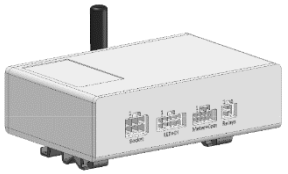


Figure 2 SSCU

2 INTRODUCTION

This manual helps you through the process of configuration and troubleshooting of the GreenFlux smart charging controllers. This document can be used for both the Dual Socket Control Unit (DSCU) and Single Socket Control Unit (SSCU) since they both use the same firmware.

The GreenFlux controllers connect to any Central System via OCPP 1.6.

2.1 SAFETY INSTRUCTIONS

When installing a GreenFlux smart charging controller, always read the safety instructions in the installation manual that comes with the hardware.

2.2 TERMS AND ABBREVIATIONS USED IN THE MANUAL AND THE PORTAL

BES	Back-end server
CSL	Comma separated list
DSCU	Dual Socket Control Unit
EV	Electric vehicle
EVSE	Electric vehicle supply equipment (socket or cable)
FTP(S)	File Transfer Protocol (Secure)
HTTP(S)	Hypertext Transfer Protocol (Secure)
ICCID	Integrated Circuit Card Identifier
IMSI	International Mobile Subscription Identity
IOCU	Input Output Control Unit
MCU	Microcontroller unit
MODE 3	Part of IEC-61851 and IEC-62196 conductive charging protocols
OCPP	Open Charge Point Protocol
SSCU	Single Socket Control Unit

2.3 ORDERING THE GREENFLUX SMART CHARGING CONTROLLER

To order (additional) GreenFlux smart charging controllers, please contact sales@greenflux.com.

2.4 BASIC CONTROLLER CONFIGURATION

Initial configuration of the charging controller will be done by GreenFlux ahead of shipping. This entails all default values, where factory-installed customer specific settings can be agreed upon beforehand.

3 CENTRAL SYSTEM OCPP

This chapter lists the OCPP 1.6 remote actions that the GreenFlux Smart Charging Controller has implemented. Execution of those actions depends on the Central System with which the controller is managed.

3.1 CANCEL RESERVATION

A matching pending reservation on the Charging Station will now be cancelled. The reserved Charge Point becomes available again.

3.2 CHANGE AVAILABILITY

To make a connector available from an unavailable status, change the status from *inoperative* to *operative*. The same can be done vice versa.

This command can for instance be used to prevent Electric Vehicles connecting to the charger when maintenance of a firmware update is planned.

3.3 CHANGE CONFIGURATION

Central System can request a Charge Point to change configuration parameters.

3.4 CLEAR CACHE

The Charging Station will clear its authorization cache. The authorization cache holds previously accepted IDs. After this command, the controller will authenticate all IDs via the Central System.

3.5 CLEAR CHARGING PROFILES

The Central System can clear some or all of the charging profiles that were previously sent the Charging Station.

Please refer to Section 6.3 for more information about how smart charging is handled within the controller.

3.6 DATA TRANSFER

If the Central System needs to send information to a Charge Point it can use this function.

Note : Not implemented.

3.7 GET COMPOSITE SCHEDULE

The Central System may request a Charge Point to report the Composite Charging Schedule. The reported schedule is the result of the calculation of all active schedules and possible local limits present in the Charge Point.

Note : Not implemented.

3.8 GET CONFIGURATION

This command will let the controller return the controller's configuration settings. A successful attempt will be answered by a message containing all configurable charging controller settings.

3.9 GET DIAGNOSTICS

The Charging Controller will send a diagnostics file to the FTP server. The Diagnostics file contains logging and system information that can be used for problem analysis etc.

The FTP server is not part of the GreenFlux environment and must be set up by the customer.

3.10 GET LOCAL LIST VERSION

In order to support synchronization of the Local Authorization List, Central System can request a Charging Station for the version number of the Local Authorization List.

3.11 HARD RESET

The Charging Station will reboot. Running charging transactions will be stopped first as if a Stop Transaction command was given.

3.12 REMOTE START TRANSACTION

This requires a token (tag) ID and a connector number. This tag ID is usually the chip ID (RFID) of a user's charge card but can be any identifier accepted by the Central System (depending on the Central System implementation). Please note that when RFID charge cards are used, the tag ID is usually is not equal to the visual ID printed on such a charge card, it is the *internal* ID of charge card. A transaction will start if the Remote Start Transaction message sent by the Central System is accepted by the Charging Station.

A situation where this command can be very useful, is when a maintenance engineer needs to interrupt a running session on a Charging Station. As he has no card to swipe against the RFID reader to stop the session, he can ask the support desk to do this remotely with the REMOTE STOP TRANSACTION command (see Section 3.13). When the maintenance is done, the session can be continued without having to swipe a charge card. The support desk can simply send the REMOTE START TRANSACTION command to continue the charging session, after looking up the ID used in the previous start transaction message in the Central System.

3.13 REMOTE STOP TRANSACTION

To remotely stop a transaction at a Charging Station, a transaction ID of the running transaction that is to be stopped must be passed alongside the command. This ID can be found in the 'Start transaction' message that was sent by the Charging Station to the Central System when the transaction was initiated.

3.14 RESERVE NOW

The Central System can reserve a Charge Point for a specific customer for a specific time / duration. The expected Idtag of the card will be send to the Charging Station and the Charging station will make sure that a Charge Point is available during the time the reservation is valid. After expiration of the reservation time the Charge Socket will become available for any user.

3.15 SEND LOCAL LIST

The Central System can send a Local Authorization List that a Charge Point can use for authorization idTags. This list contains the IdTags that can start a charge

session. It is also possible to add entries to a local list using the Local Management facility of the controller itself (no dependency with a Central System needed in that case). Please refer to the "*Installation manual*" of the controller how to get access to this local management and edit the local list.

3.16 SET CHARGING PROFILE

The Central System can set a charging profile for the Charging Station or its individual Charge Points.

Please refer to Section 6.3 for more information about how smart charging is handled within the controller.

3.17 SOFT RESET

Terminates any running transaction, as if a Stop Transaction command was given. The charging controller will then return to its state as if it had just been booted.

3.18 TRIGGER MESSAGE

The Central System can send a trigger message to have the controller send an update of its operational status. Mostly used when the Central System is not sure what the actual state of a Charge Point / Charging Station is. Status information can be retrieved regarding E.g. the following states: Boot information, Charge socket information, Meter values, Firmware update Status, Heartbeat, Diagnostics. All OCPP define trigger messages are supported.

3.19 UNLOCK CONNECTOR

The specified connector will be unlocked. Only use this command when a *RemoteStopTransaction* (ref 3.13) fails.

3.20 UPDATE FIRMWARE

The GreenFlux smart charging controller firmware is remotely updatable.

New firmware versions can and will only be released by GreenFlux. New firmware versions need to be placed on a FTP server and the Update Firmware command translates the address of the FTP server to the Charging Station. The FTP server is not part of the GreenFlux environment and must be set up by the customer.

Note : As from version 4.2.0 and higher, retrieving the controller's configuration settings will always reset the local management password (please refer to the configuration parameter *LocalManagementPassword* in Section 4.4).

Not part of the GreenFlux environment and must be set up by the customer.

4 CHARGE POINT CONFIGURATION

This Section explains the usage of the configuration items available in the GreenFlux Smart Charging Controller.

4.1 GENERAL

Be aware that improper connection configurations can abruptly stop the remote accessibility of the controller, making it impossible to undo any previous steps.

It is very important to carefully check this manual before making any changes to the configuration settings of a Charging Station.

Not all configuration changes become active immediately after changing them, some changes will require a reset of the controller by activating a remote reset or performing a hard reset (power-cycle).

Many settings listed in this Section are related to / defined by the OCPP 1.6 protocol. Section 1.2.2 shows how you can obtain a protocol description.

The success or failure of a configuration change is published by the controller with a response to the Change Configuration message. The change result is therefore always visible in the Central system.

The Charge Controller also has a Local Management functionality available (as from firmware version 4.2.0 and higher). With this Local Management function, the controller hosts a local website that can also be used for changing the working configuration of the controller.

Please refer to the "*Installation manual*" of the controller how to get access to this local management. The Local Management functionality enables making configuration changes without using a Central System. This can be very useful for a field engineer.

Please take note of the following. Several parameters have a high impact on the behavior of the controller. Changing them will e.g. influence connectivity and actual charge rate. In case of doubt, please contact GreenFlux before making any changes.

4.2 HOW TO USE THIS SECTION

For describing the functionality of a specific configuration parameter, the following layout format is used. The template describes the general functionality the parameter affects/controls and shows all relevant setting values.

Controller Support	This field indicates which controller supports the given configuration parameter:

DSCU : Dual Socket Control unit (prod# : T2235-02)

SSC : Single Socket Control unit (prod# : 1040300)

Description

Shortly describes the purpose of the configuration and what the effects changing it will have on the controller's behavior.

Type

Indicates the type of parameter:

Integer : A whole number (e.g. 3, -5, 500 etc.).

Float : A floating decimal number (e.g. 17.3).

Boolean : Logical state (e.g. True or False)

String : Any combination of alphanumeric characters (e.g. "Hello World 123").

CSL : Comma Separate List (e.g. "actor", "listener77", "zone1")

List : Function list. (e.g. [[0, "option1"],[5, "option2"],[2, "option3"]])

Value(s)

Lists the possible values this configuration parameter may contain, e.g. values for an Integer and Boolean typed parameter:

0 .. 100 : An integer value between 0 and 100 (including both lower and upper value).

True, False : Only both given values can be used (Boolean).

Default value

The default (factory) value. Please, also refer to Section 2.4.

Reboot required

Whether a reboot is required

Yes : The changed configuration parameter will only become active after resetting the controller (by issuing a hard-reset command or conducting a power cycle locally).

No : After changing, the new value will become active immediately.

Field validation

Yes : The entered value is validated by software. Only valid values are accepted. Erroneous values are denied and reported to the

requestor using OCPP in case of a change issued by the Central System, or directly when using the Local Management Portal.

No : Entered value is not validated but accepted as is.

4.3 ALL CONFIGURATION PARAMATERS

4.4 CONFIGURATION PARAMETER DEFINITIONS

This Section describes the available configuration parameters. Parameters are ordered alphabetically.

AllowOfflineTxForUnknownId

Controller Support	DSCU, SSCU
Description	Configuration used when controller is offline, or no authorization response is received in time from the Central System. If the presented *IdTag* is not known, not valid or has expired in the LocalList and AutohorizationCache, the next action is performed: **True**: Accept tag and start charge session **False**: Reject tag.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	No
Field validation	Yes

AuthorisationCacheEnabled

Controller Support	DSCU, SSCU
Description	If the authorization cache is enabled, the accepted RFID tags will be stored in the authorization cache. When the same RFID tag is trying to authenticate within 7 days, the tag will be accepted by the cache instead of the back-end server. Final authentication is always done by a *StartTransaction*. If the cache is disabled, the tag is always authenticated by the Central System.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	No
Field validation	Yes

AuthorizeRemoteTxRequests

Controller Support	DSCU, SSCU
Description	Sets whether a remote request for starting a transaction needs to validate the *IdTag* given in the *RemoteStartTransaction.req*.

True : The Charge Point SHALL behave as if in response to a local action at the Charge Point (scanning a RFId card) to start a transaction with the idTag given in the *RemoteStartTransaction.req* message. This means that the Charge Point will first try to authorize the idTag, using the Local Authorization List, Authorization Cache and/or an *Authorize.req* request.

A transaction will only be started after authorization was obtained.

False : The Charge Point SHALL immediately try to start a transaction for the idTag given in the *RemoteStartTransaction.req* message. Note that after the transaction has been started, the Charge Point will send a StartTransaction request to the Central System, and the Central System will check the authorization status of the idTag when processing this StartTransaction request.

Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	No
Field validation	Yes

BESPath

Controller Support	DSCU, SSCU
Description	WebSocket communication path for connecting to the Central System (aka Back-End Server).
Type	String
Value(s)	*<BESPath>*
Default value	/
Reboot required	Yes
Field validation	No

BESPortnumber

Controller Support	DSCU, SSCU
Description	WebSocket communication port number for connecting to the Central System (aka Back-End Server).
Type	Integer
Value(s)	0 .. 65535
Default value	80
Reboot required	Yes

Field validation	Yes

BESURL

Controller Support	DSCU, SSCU
Description	WebSocket communication URL for connecting to the Central System (aka Back-End Server).
Type	String
Value(s)	*<BESURL>*
Default value	chargepointservicej.greenflux.nl
Reboot required	Yes
Field validation	No

BlinkingInterval

Controller Support	DSCU, SSCU
Description	Sets the interval between two successive blinks of the LED in milliseconds.
Type	Integer
Value(s)	50 .. 10000
Default value	500

Reboot required	No
Field validation	Yes

BuzzerActive

Controller Support	DSCU
Description	Sets buzzer usage during operation.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	No
Field validation	Yes

BuzzerInterval

Controller Support	DSCU
Description	Sets the interval between two successive buzzes in milliseconds.
Type	Integer
Value(s)	0 .. 2000
Default value	250

Reboot required	No
Field validation	Yes

BuzzerPeriod

Controller Support	DSCU
Description	Duration of the buzzer ‘beep’ (milliseconds).
Type	Integer
Value(s)	0 .. 2000
Default value	250
Reboot required	No
Field validation	Yes

ChargePointCurrentLimit

Controller Support	DSCU, SSCU
Description	The maximum current that the Charging Station can draw from the grid. This value defines the maximum current for the whole Charging Station, thus all enabled Charge Points (sockets) combined.
Type	Integer
Value(s)	0 .. 80
Default value	32
Reboot required	No
Field validation	Yes

ChargePointVendor

Controller Support	DSCU, SSCU
Description	The name of the Charging Station vendor.
Type	String
Value(s)	*<VendorName>*
Default value	Greenflux
Reboot required	Yes
Field validation	No

ChargingProfileMaxStackLevel

Controller Support	DSCU, SSCU
Description	Max stack level of a charging profile. The number defined also indicates the max allowed number of installed charging

	schedules per charging profile purposes. See the OCPP documentation for more information on the use of stack levels.
Type	Integer (read-only)
Value(s)	N/A
Default value	6
Reboot required	N/A
Field validation	N/A

ChargingScheduleAllowedChargingRateUnit

Controller Support	DSCU, SSCU
Description	A list of supported quantities for use in a Charging Schedule ([A] / [W]).
Type	CSL (read-only)
Value(s)	N/A
Default value	Current [A]
Reboot required	N/A
Field validation	N/A

ChargingSchedulesMaxPeriods

Controller Support	DSCU, SSCU
Description	Maximum number of periods that may be defined per charging schedule. **Note:** This setting is inactive and intended for future use.
Type	Integer (read-only)
Default value	255
Reboot required	N/A
Field validation	N/A

ClientCertificate

Controller Support	DSCU, SSCU
Description	Client certificate to be used for setting up a secured WebSocket connection with the Central System. **Note:** This setting is inactive and intended for future use.
Type	String
Value(s)	*<certificate name>*
Default value	test.cert

Reboot required	Yes
Field validation	No

ClientKey

Controller Support	DSCU, SSCU
Description	Client key to be used for setting up a secured WebSocket connection with the Central System. **Note:** This setting is currently inactive and intended for future use.
Type	String
Value(s)	*<client key value>*
Default value	key.txt
Reboot required	Yes
Field validation	No

ClockAlignedDataInterval

Controller Support	DSCU, SSCU
Description	Interval in seconds of the clock-aligned data interval. Intervals start from 00.00 (midnight). A value of "0" (numeric zero), by convention, is to be interpreted to mean that no clock-aligned data should be transmitted. **Note** : Mind the difference with the *MeterValueSampleInterval* setting, the start of these intervals always start at the beginning of a charging transaction. and relates to meter values that are only sent during a transaction, whereas *ClockAlignedDataInterval* relates to those meter values that are sent at each data interval, regardless of whether there is an active charging session or not.
Type	Integer
Value(s)	0 .. 86400
Default value	900
Reboot required	No
Field validation	Yes

ConnectionTimeOut

Controller Support	DSCU, SSCU
Description	The maximum allowed time (in seconds) between a successful authorization (a valid IdTag or remotely started transaction) and the

	actual start of a charging session (EV has been plugged in correctly). **Note** : Also refer to *PlugPresentTimeout*.
Type	Integer
Value(s)	10 .. 900
Default value	30
Reboot required	False
Field validation	Yes

Connector1Enable

Controller Support	DSCU, SSCU
Description	**True**: Charge Point 1 is Available / enabled. **False**: Charge Point 1 is Unavailable.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	Yes
Field validation	Yes

Connector1FlexCable

Controller Support	DSCU, SSCU
Description	**True**: Charge Point 1 is equipped with a socket. **False**: Charge Point 1 is equipped with a tethered cable.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	Yes
Field validation	Yes

Connector2Enabled

Controller Support	DSCU
Description	**True**: Charge Point 2 is Available / enabled. **False**: Charge Point 2 is Unavailable.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	Yes

Field validation	Yes

Connector2FlexCable

Controller Support	DSCU, SSCU
Description	**True**: Charge Point 2 is equipped with a socket. **False**: Charge Point 2 is equipped with a tethered cable.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	Yes
Field validation	Yes

ConnectorPhaseRotation

Controller Support	DSCU, SSCU
Description	Gives the phase rotation per connector in respect to the connector's energy meter (or if absent, the grid connection). Possible values per connector are: *Not Applicable* (for Single phase or DC Charge Points) *Unknown*: not (yet) known *RST* : Standard Reference Phasing *RTS* : Reversed Reference Phasing *SRT* : Reversed 240-degree rotation *STR* : Standard 120-degree rotation *TRS* : Standard 240-degree rotation *TSR* : Reversed 120-degree rotation R can be identified as phase 1 (L1), S as phase 2 (L2), T as phase 3 (L3). If known, the Charge Point MAY also report the phase rotation between the grid connection and the main energy meter by using index number Zero (0). Values are reported in CSL, formatted: 0.RST, 1.RST, 2.RTS **Note:** This setting is currently inactive and intended for future use.
Type	CSL (read-only, currently inactive)
Value(s)	*<Phase rotation info>*
Default value	1.Unknown,2.Unknown
Reboot required	N/A
Field validation	N/A

ConnectorSwitch3to1PhaseSupported

Controller Support	DSCU, SSCU
Description	If defined and true, this Charging Station supports switching from 3 to 1 phase during a charging session. Note that simply changing this setting will not automatically enable this function, the Charging Station hardware needs to support this functionality as well.
Type	Boolean (read-only)
Value(s)	True, False
Default value	False
Reboot required	N/A
Field validation	N/A

DetailedWarningMessages

Controller Support	DSCU, SSCU
Description	If set to True, the controller will send more detailed messages in case an abnormally is detected. For example, when the controller receives a smart charging profile that cannot be delivered by the Charging Station, a warning message will then be sent to indicate this problem.
Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	No
Field validation	Yes

ExternalSensor1CoupledConnector

Controller Support	SSCU
Description	The Charging Station that is affected by the actions defined for this sensor (0=all connectors). Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	Integer
Value(s)	0 .. 1
Default value	0
Reboot required	Yes
Field validation	Yes

ExternalSensor1Name

Controller Support	SSCU
Description	Sensor name (will be ignored unless *ExternalSensor1Type* is of type “99-OTHER”. Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	String
Value(s)	*<Sensor name>* (in case type = “99-OTHER”.)
Default value	Disabled
Reboot required	Yes
Field validation	Yes

ExternalSensor1NormallyOpen

Controller Support	SSCU
Description	Normal (untriggered) state of the sensor Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	Boolean
Value(s)	True (input contact = High/Open), False (input contact = Low/Closed)
Default value	True
Reboot required	Yes
Field validation	Yes

ExternalSensor1RecoverActions

Controller Support	SSCU
Description	Actions to perform when sensor enters the recovered state. A *0*-value means: ignore the specified action. Values other than *0* indicates the priority/order of execution. Priority is defined in ascending order. Beware, these value musts not exceed the number of actions available! By default, no actions are defined. **Note** : The list is fixed regarding to the possible actions – no action can be added manually. Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	List
Value(s)	*<List with action definitions>*

Default value	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]
Reboot required	Yes
Field validation	Yes

ExternalSensor1TriggerActions

Controller Support	SSCU
Description	Actions to perform when the sensor is triggered (activated). A *0*-value means: ignore the specified action. Values other than *0* indicates the priority/order of execution. Priority is defined in ascending order. Beware, these value musts not exceed the number of actions available! By default, no actions are defined. **Note** : The list is fixed regarding to the possible actions – no action can be added manually. Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	List
Value(s)	*<List with action definitions>*
Default value	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]
Reboot required	Yes
Field validation	Yes

ExternalSensor1Type

Controller Support	SSCU
Description	The type of the sensor. Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	Integer
Value(s)	1 = TILT 2 = WATER 3 = TEMPERATURE 4 = LIGHT 5 = OPENDOOR 6 = BUTTON

	7 = GROUND_FAILURE 8 = CCID (refer to Section 7) 99 = OTHER (user defined)
Default value	0 (not used)
Reboot required	Yes
Field validation	Yes

ExternalSensor2CoupledConnector

Controller Support	SSCU
Description	The actual connector that is affected by the actions defined for this sensor (0=all connectors). Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	Integer
Value(s)	0 .. 1
Default value	0
Reboot required	Yes
Field validation	Yes

ExternalSensor2Name

Controller Support	SSCU
Description	Sensor name (will be ignored unless *ExternalSensor1Type* is of type "99-OTHER". Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	String
Value(s)	*<Sensor name>* (in case type = "99-OTHER".)
Default value	Disabled
Reboot required	Yes
Field validation	Yes

ExternalSensor2NormallyOpen

Controller Support	SSCU
Description	Normal (untriggered) state of the sensor Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	Boolean

Value(s)	True (contact=High/Open), False (contact= Low/Closed)
Default value	True
Reboot required	Yes
Field validation	Yes

ExternalSensor2RecoverActions

Controller Support	SSCU
Description	Action to perform when sensor enters the recovered state. A *0*-value means: ignore the specified action. Values other than *0* indicates the priority/order of execution. Priority is defined in ascending order. Beware, these value musts not exceed the number of actions available! By default, no actions are defined. Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	List
Value(s)	*<List with action definitions>*
Default value	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]
Reboot required	Yes
Field validation	Yes

ExternalSensor2TriggerActions

Controller Support	SSCU
Description	Action to perform when the sensor is triggered (activated). A *0*-value means: ignore the specified action. Values other than *0* indicates the priority/order of execution. Priority is defined in ascending order. Beware, these value musts not exceed the number of actions available! By default, no actions are defined. Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	List
Value(s)	*<List with action definitions>*
Default value	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]

Reboot required	Yes
Field validation	Yes

ExternalSensor2Type

Controller Support	SSCU
Description	The type of the sensor. Please refer to Section 5 for a detailed description about how to use the (external) sensor events.
Type	Integer
Value(s)	1 = TILT 2 = WATER 3 = TEMPERATURE 4 = LIGHT 5 = OPENDOOR 6 = BUTTON 7 = GROUND_FAILURE 8 = CCID (refer to Section 7) 99 = OTHER (user defined)
Default value	0 (not used)
Reboot required	Yes
Field validation	Yes

FirmwareVersionIOCU

Controller Support	DSCU, SSCU
Description	The actual version of the firmware running on the IOCU processor. The version information is used for referencing purpose. Section 8.1 gives a description of the IOCU and is functionality.
Type	String (read-only)
Value(s)	N/A
Default value	*<firmware version info>*
Reboot required	N/A
Field validation	N/A

FirmwareVersionMCU

Controller Support	DSCU, SSCU

Description	The actual version of the firmware running on the MCU processor. The version information is used for referencing purpose. Section 8.1 gives a description of the MCU and is functionality.
Type	String (read-only)
Value(s)	N/A
Default value	*<firmware version info*
Reboot required	N/A
Field validation	N/A

FlexConnectorWithActuator

Controller Support	DSCU, SSCU
Description	Indicates the presence of a servo lock actuator (servo) fitted on to the (flex) socket of a Charge Point. In combination with parameter *Connector1FlexCable* and *Connector2FlexCable* the desired configuration can be set up. **Note** : This setting is active for all available flex Charge Points of a Charging Station, meaning, Flex Charge Points are both equipped with an actuator or not. Mixing this flex options is not possible. **Mind** : Make sure to set configuration *StopTransactionOnEVSideDisconnect* to *True*, otherwise a once started charge session cannot be stopped normally anymore.
Type	Boolean
Value(s)	True (Socket has actuator), False (Socket has **no** actuator)
Default value	True
Reboot required	Yes
Field validation	Yes

FTPPassword

Controller Support	DSCU, SSCU
Description	Password for the FTP server when no password is given in the URL calling the FTP server. An FTP server connection is used for retrieving system diagnostics or performing a remote firmware update. **Note :** Intended for future use.
Type	String
Value(s)	*<FTP password>*

Default value	FTPPASS
Reboot required	No
Field validation	No

FTPUserName

Controller Support	DSCU, SSCU
Description	Username for the FTP server when a username is not given in the URL calling the FTP server. An FTP server connection is used for retrieving system diagnostics or performing a remote firmware update. **Note :** Intended for future use.
Type	String
Value(s)	*<FTP user name>*
Default value	FTPUSER
Reboot required	No
Field validation	No

GetConfigurationMaxKeys

Controller Support	DSCU, SSCU
Description	Maximum number of requested configuration keys in a *GetConfiguration.req* PDU. This number specifies how many configuration items the controller can send at a time.
Type	Integer (read-only)
Value(s)	N/A
Default value	255
Reboot required	N/A
Field validation	N/A

gprsAPN

Controller Support	DSCU, SSCU
Description	The APN to be used for setting up the mobile connection.
Type	String
Value(s)	*<APN>*
Default value	greenflux.tele2.m2m
Reboot required	No
Field validation	No

gprsPASSWORD

Controller Support	DSCU, SSCU
Description	The password to be used for setting up the mobile connection with the provider.
Type	String
Value(s)	*<password>*
Default value	*<blank>*
Reboot required	No
Field validation	No

gprsQcfgBand

Controller Support	SSCU
Description	Defines the 4G LTE bands used on the LTE M1 network. Depending on location (continent) the desired bands can be selected. **Note :** setting to *ALL* will scan all available bands. Connecting to the network in that case may take considerable time. Note: In Value(s) table between brackets the actual bands are given that will be used when selecting its location. E.g *JAPAN* wil use bands 1,8,18,19.
Type	String
Value(s)	ALL *(1,2,3,4,5,8,12/17,13,18,19,20,26,28,39)* EUROPE *(3,8,20)* US *(2,4,12/17,13)* JAPAN *(1,8,18,19)* AUSTRALIA *(3,28)* CHINA *(1,3,5,8,26,39)* MIDDLE_EAST *(3,8,28)* KOREA *(3,5)*
Default value	ALL
Reboot required	No
Field validation	No

gprsUSER

Controller Support	DSCU, SSCU
Description	The user name to be used for setting up the mobile connection with the provider.
Type	String
Value(s)	*<User>*

Default value	*<blank>*
Reboot required	No
Field validation	No

HeartBeatInterval

Controller Support	DSCU, SSCU
Description	Interval of inactivity (no OCPP data exchange) with the Central System after which the Charging Station should send a *Heartbeat.req PDU*. If set to 0, no heartbeats are sent.
Type	Integer
Value(s)	0 .. 65234
Default value	900
Reboot required	No
Field validation	Yes

IdleLedActive

Controller Support	DSCU, SSCU
Description	Defines whether the LED is on (True) or off (False) when the corresponding Charge Point is Idle (available). **Note** : This setting is inactive and intended for future use.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	No
Field validation	Yes

InversedServo

Controller Support	DSCU, SSCU
Description	Servo lock position information is by default based on the Mennekes socket specification. Some other servo lock sockets use an inversed signaling. With this configuration the servo status signal reading can be inversed to support these kinds of locks.
Type	Boolean
Value(s)	True, False
Default value	False

Reboot required	Yes
Field validation	Yes

LEDcolor_Authorized

Controller Support	DSCU, SSCU
Description	Defines the color of the LED indicator of a charge Point when a valid **Authorisation** is found. Also, the blinking of the LED can be set.
Type	String
Value(s)	RED, RED_BLINKING, GREEN, GREEN_BLINKING, BLUE, BLUE_BLINKING, YELLOW, YELLOW_BLINKING, MAGENTA, MAGENTA_BLINKING, CYAN, CYAN_BLINKING
Default value	GREEN_BLINKING
Reboot required	False
Field validation	False

LEDcolor_Available

Controller Support	DSCU, SSCU
Description	Defines the color of the LED indicator of a charge Point when in **Available** state. Also, the blinking of the LED can be set. Refer to Section 8.3 for more information about Charge Point states descriptions.
Type	String
Value(s)	RED, RED_BLINKING, GREEN, GREEN_BLINKING, BLUE, BLUE_BLINKING, YELLOW, YELLOW_BLINKING, MAGENTA, MAGENTA_BLINKING, CYAN, CYAN_BLINKING
Default value	GREEN
Reboot required	False
Field validation	False

LEDcolor_Charging

Controller Support	DSCU, SSCU
Description	Defines the color of the LED indicator of a charge Point when in **Charging** state. Also, the blinking of the LED can be set. Refer to Section 8.3 for more information about Charge Point states descriptions.

Type	String
Value(s)	RED, RED_BLINKING, GREEN, GREEN_BLINKING, BLUE, BLUE_BLINKING, YELLOW, YELLOW_BLINKING, MAGENTA, MAGENTA_BLINKING, CYAN, CYAN_BLINKING
Default value	BLUE
Reboot required	False
Field validation	False

LEDcolor_Preparing

Controller Support	DSCU, SSCU
Description	Defines the color of the LED indicator of a charge Point when in **Preparing** state. Also, the blinking of the LED can be set. Refer to Section 8.3 for more information about Charge Point states descriptions.
Type	String
Value(s)	RED, RED_BLINKING, GREEN, GREEN_BLINKING, BLUE, BLUE_BLINKING, YELLOW, YELLOW_BLINKING, MAGENTA, MAGENTA_BLINKING, CYAN, CYAN_BLINKING
Default value	YELLOW
Reboot required	False
Field validation	False

LEDcolor_Reserved

Controller Support	DSCU, SSCU
Description	Defines the color of the LED indicator of a charge Point when in **Reserved** state. Also, the blinking of the LED can be set. Refer to Section 8.3 for more information about Charge Point states descriptions.
Type	String
Value(s)	RED, RED_BLINKING, GREEN, GREEN_BLINKING, BLUE, BLUE_BLINKING, YELLOW, YELLOW_BLINKING, MAGENTA, MAGENTA_BLINKING, CYAN, CYAN_BLINKING
Default value	MAGENTA
Reboot required	False
Field validation	False

LEDcolor_Reserved_Authorized

Controller Support	DSCU, SSCU
Description	Defines the color of the LED indicator of a charge Point when Authorized in **Reserved** state. Also, the blinking of the LED can be set. Refer to Section 8.3 for more information about Charge Point states descriptions.
Type	String
Value(s)	RED, RED_BLINKING, GREEN, GREEN_BLINKING, BLUE, BLUE_BLINKING, YELLOW, YELLOW_BLINKING, MAGENTA, MAGENTA_BLINKING, CYAN, CYAN_BLINKING
Default value	MAGENTA_BLINKING
Reboot required	False
Field validation	False

LEDcolor_Suspended

Controller Support	DSCU, SSCU
Description	Defines the color of the LED indicator of a charge Point when in **Suspended** state. Also, the blinking of the LED can be set. Refer to Section 8.3 for more information about Charge Point states descriptions.
Type	String
Value(s)	RED, RED_BLINKING, GREEN, GREEN_BLINKING, BLUE, BLUE_BLINKING, YELLOW, YELLOW_BLINKING, MAGENTA, MAGENTA_BLINKING, CYAN, CYAN_BLINKING
Default value	YELLOW
Reboot required	False
Field validation	False

LEDcolor_Unavailable

Controller Support	DSCU, SSCU
Description	Defines the color of the LED indicator of a charge Point when in **Unavailable** state. Also, the blinking of the LED can be set. Refer to Section 8.3 for more information about Charge Point states descriptions.

Type	String
Value(s)	RED, RED_BLINKING, GREEN, GREEN_BLINKING, BLUE, BLUE_BLINKING, YELLOW, YELLOW_BLINKING, MAGENTA, MAGENTA_BLINKING, CYAN, CYAN_BLINKING
Default value	RED
Reboot required	False
Field validation	False

L1Active

Controller Support	DSCU, SSCU
Description	Sets whether the Charge Point phase L1 connection is installed / available and can be used for charging the EV. If this is set to true, the under and over voltage measurements will be monitored on this phase and in case of an error reported to the Central System. Normally, phase L1 must be set to true (since, at least one phase should be used). Also refer to *L2Active* and *L3Active*).
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	True
Field validation	True

L2Active

Controller Support	DSCU, SSCU
Description	Sets whether the Charge Point phase L2 connection is installed / available and can be used for charging the EV. If this is set to true, the under and over voltage measurements will be monitored on this phase and in case of an error reported to the Central System. This setting must be set to true in case of a 2-phase (L1+L2) or 3-phase (L1+L2+L3) Charge Point configuration. Also refer to *L1Active* and *L3Active*).
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	True
Field validation	True

L3Active

Controller Support	DSCU, SSCU
Description	Sets whether the Charge Point phase L3 connection is installed / available and can be used for charging the EV. If this is set to true, the under and over voltage measurements will be monitored on this phase and in case of an error reported to the Central System. This setting must be set to true in case of a 3-phase (L1+L2+L3) Charge Point configuration. Also refer to *L1Active* and *L2Active*).
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	True
Field validation	True

LedActive

Controller Support	DSCU, SSCU
Description	Sets whether the LED is actively used by the controller software. **True**: LED gives visual indications. **False**: LED is always off. **Note**: This setting is inactive and intended for future use.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	No
Field validation	Yes

LEDIntensityLow

Controller Support	DSCU, SSCU
Description	Minimum LED intensity value. Used with E.g. the external sensor scenario handling. Please refer to Section 5.6 and 5.7 for more information about using this setting in combination with (external) sensor events.
Type	Integer
Value(s)	1 (dimmed) .. 5 (bright)
Default value	5
Reboot required	No

Field validation	Yes

LEDIntensityHigh

Controller Support	DSCU, SSCU
Description	Maximum LED intensity value. Used with E.g. the external sensor scenario handling. Please refer to Section 5.6 and 5.7 for more information about using this setting in combination with (external) sensor events.
Type	Integer
Value(s)	1 (dimmed) .. 5 (bright)
Default value	5
Reboot required	No
Field validation	Yes

LEDIntensityDefault

Controller Support	DSCU, SSCU
Description	Default LED intensity after start-up.
Type	Integer
Value(s)	1 (dimmed) .. 5 (bright)
Default value	5
Reboot required	No
Field validation	Yes

LocalAuthListEnabled

Controller Support	DSCU, SSCU
Description	Enables use of the local authorization list, which overrules he authorization via the connected Central System.
Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	No
Field validation	Yes

LocalAuthListMaxLength

Controller Support	DSCU, SSCU

Description	Maximum number of identifications that can be stored in the Local Authorization List.
Type	Integer (read-only)
Value(s)	N/A
Default value	255
Reboot required	N/A
Field validation	N/A

LocalAuthorizeOffline

Controller Support	DSCU, SSCU
Description	The controller supports an authorization cache that autonomously maintains a record of previously presented identifiers that have been successfully authorized by the Central System. **True**: The controller will try to accept an IdTag using the authorization cache. **False**: The controller will skip the authorization cache to authorize the IdTag. **Note**: The actual acceptance of the IdTag is also controlled in combination with *AllowOfflineTxForUnknownId*.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	No
Field validation	Yes

LocalManagementPassword

Controller Support	DSCU, SSCU
Description	The currently active password to access the local management website of the controller  **Important:** The password will be changed every time a “GetConfiguration” command is issued from the connected Central System. This automatically means that the currently active password can only be obtained using the Central System. The default password remains active until the Central System issues “GetConfiguration” command for the first time.

	Please refer to the *Smart Charging Controller Installation Manual* for more information regarding the usage of the Local Management website.
Type	String (read-only)
Value(s)	*<Management password>*
Default value	$r33nFl^x
Reboot required	N/A
Field validation	N/A

LocalPreAuthorize

Controller Support	DSCU, SSCU
Description	**True**: Whether the Charge Point, when online, will start a transaction for locally authorized identifiers (IdTag) without waiting for an authorization from the Central System. **False**: Always authenticate via Central System. Does not use any locally authorized identifiers (IdTag) such as an available local or white list. **Notes** : The *Local List* is a locally or centrally maintained list of IdTags that can start a charge session. The list itself is stored inside the controller. The *White List* is a list with TagIds that has been approved on by the Central System to be valid for starting a charge session. This list is also stored inside the controller.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	No
Field validation	Yes

MACAddress

Controller Support	DSCU, SSCU
Description	MAC address of the ethernet connection.
Type	String (read-only)
Value(s)	N/A
Default value	*<MAC adress>*
Reboot required	N/A
Field validation	N/A

MainsVoltage

Controller Support	DSCU, SSCU
Description	The normally expected mains voltage value (Volt). This value is used for calculation purposes in case a S0-pulse kWh meter is used as energy meter. The value is e.g. used for overload protection calculations. Please note that S0-pulse meters should only be used in single phase connected chargers and are generally not recommended since the charging current is not measured but calculated.
Type	Integer
Value(s)	100 .. 250
Default value	230
Reboot required	No
Field validation	Yes

MaxChargingProfilesInstalled

Controller Support	DSCU, SSCU
Description	Maximum number of charging profiles installed at a time. The controller cannot handle more charging profiles at one time.
Type	Integer (read-only)
Value(s)	N/A
Default value	8
Reboot required	N/A
Field validation	N/A

MaxCurrentConnection

Controller Support	DSCU, SSCU
Description	The maximum allowed current on each Charge Point. Value is given in Ampere [A]. In case of using a DSCU, each of the two Charge Points will have the same maximum allowed current. **Note**: Please ref to Section **Error! Reference source not found.** regarding the Local Load Balancing functionality of the controller.
Type	Integer
Value(s)	0 .. 80
Default value	16
Reboot required	Yes

Field validation	Yes

MaxEnergyOnInvalidId

Controller Support	DSCU, SSCU
Description	Maximum energy in [Wh] delivered when an identifier (e.g. IdTag) is invalidated by the Central System after start of a transaction when *StopTransactionOnInvalidId* is set to **False**. The given value will allow the customer to drive away from the charger with a (small) amount of added energy. When *StopTransactionOnInvalidId* is **True**, *MaxEnergyOnInvalidId* is not used. **Note :** See also *StopTransactionOnInvalidId*.
Type	Integer
Value(s)	0 .. 65535
Default value	0 (no energy)
Reboot required	No
Field validation	Yes

MessageResponseTimeOut

Controller Support	DSCU, SSCU
Description	The maximum accepted delay time for receiving a response from the Central System after sending a request. After this time-out the recovery function belonging to the request will be triggered when available, e.g. retrying to resend the specific message. In combination with the *TransactionMessageRetryInterval* the message response handling with the Central System is configured.
Type	Integer
Value(s)	3 .. 300
Default value	30
Reboot required	No
Field validation	Yes

MeterPulseCount

Controller Support	DSCU, SSCU
Description	In case of using a pulse kWh meter for registering the amount of delivered energy, this number must be equal to the number of pulses that matches with 1 kWh energy flowing through the pulse meter. This number

	must therefore match the settings on the meter.
Type	Integer
Value(s)	1 .. 10000
Default value	100
Reboot required	Yes
Field validation	Yes

MeterType

Controller Support	DSCU, SSCU
Description	The type of the power meter(s) of the Charging Station. Currently, the below mentioned types of meters are supported. Drivers for other meter types can be added, please contact sales@greenflux.com for more information about this.
Type	Integer
Value(s)	1 (ABB B23) 2 (Carlo Gavazzi EM24DIN) 3 (Pulse meter) 4 (Eastron SDM630) 5 (Carlo Gavazzi EM340) 6 (Phoenix Contacts EEM350 7 (Tele S6 XM50A 1000VM) 8 (Tele S9 XM300A 1000VM) 9 (EKM Omnimeter Pulse UL v.4 Modbus) 10 (EKM Metering Omnimeter Pulse UL v.4) 11 (Konect 1.4) 12 (CET PMC-340) 13 (Carlo Gavazzi EM111) 14 (Eastron SDM230) 15 (Iskra WM3-6) 16 (Schrack MGRZK465)
Default value	1 (ABB B23)
Reboot required	Yes
Field validation	Yes

MeterValueSampleInterval

Controller Support	DSCU, SSCU
Description	Interval (in seconds) between sampling of metering data, intended to be transmitted to the Central System. For charging session data, samples are acquired and transmitted

periodically at this interval from the start of the charging transaction.

A value of "0" (numeric zero), by convention, is to be interpreted to mean that no sampled data should be transmitted.

Only one interval can be chosen for the intervals at which all sampled data (e.g. currents, voltages, etc.) is sent. Please refer to *MeterValuesSampledData.*

Note : Mind the difference with the *ClockAlignedDataInterval* setting, the start of these intervals always start from 0:00 hours (midnight) and are sent regardless of whether or not an active session is taking place, whereas the *MeterValueSampleInterval* relates to meter values that are only sent from the Charging Station to the Central System during a transaction.

Type	Integer
Value(s)	0 .. 86400
Default value	180
Reboot required	No
Field validation	Yes

MeterValuesAlignedData

Controller Support	DSCU, SSCU
Description	Clock-aligned measurand(s) to be included in a MeterValues.req PDU message, every *ClockAlignedDataInterval* seconds.
Type	CSL
Value(s)	Please refer to the OCPP 1.6 documentation for a list of enumeration values for measurands and their meaning. Depending on whether the connected meter can measure these quantities, the following fields are supported: • Current.Import • Energy.Active.Export.Register • Energy.Active.Import.Register • Energy.Reactive.Export.Register • Energy.Reactive.Import.Register • Power.Active.Import • Voltage • Frequency
Default value	Energy.Active.Import.Register
Reboot required	No
Field validation	No

MeterValuesAlignedDataMaxLength

Controller Support	DSCU, SSCU
Description	Maximum number of items in the *MeterValuesAlignedData* configuration setting. This sets the maximum measurands to be sent to the Central System.
Type	Integer (read-only)
Value(s)	N/A
Default value	8
Reboot required	N/A
Field validation	N/A

MeterValuesSampledData

Controller Support	DSCU, SSCU
Description	Sampled measurands to be included in a MeterValues.req PDU, every *MeterValueSampleInterval* seconds. Where applicable, the Measurand is combined with the optional phase; for instance: *Voltage.L1.*
Type	CSL
Value(s)	*Please refer to the OCPP 1.6 documentation for a list of enumeration values for measurands.*
Default value	Energy.Active.Import.Register, Current.Import,Current.Offered
Reboot required	No
Field validation	No

MeterValuesSampledDataMaxLength

Controller Support	DSCU, SSCU
Description	Maximum number of items in the *MeterValuesSampledData* configuration setting. This sets the maximum measurands to be send to the Central System.
Type	Integer (read-only)
Value(s)	N/A
Default value	8
Reboot required	N/A
Field validation	N/A

MinCurrentConnection

Controller Support	DSCU, SSCU
Description	The result of the local load balancing calculations will use this setting as lower bound value for setting the allowed charge rate. The value must be smaller or equal to MaxCurrentConnection. **Note**: Please ref to Section 6 regarding the Local Load Balancing functionality of the controller.
Type	Float
Value(s)	0.0 .. 40.0
Default value	12.5
Reboot required	Yes
Field validation	Yes

Model

Controller Support	DSCU, SSCU
Description	The name of the charge station model.
Type	String (read-only)
Value(s)	N/A
Default value	Home
Reboot required	N/A
Field validation	N/A

ModemICCID

Controller Support	DSCU, SSCU
Description	The *ICCID* of the SIM-card placed in the controller.
Type	String (read-only)
Value(s)	N/A
Default value	*<ICCID value>*
Reboot required	N/A
Field validation	N/A

ModemIMSI

Controller Support	DSCU, SSCU
Description	The *IMSI* of the SIM-card placed in the controller.
Type	String (read-only)
Value(s)	N/A
Default value	*<IMSI value>*

Reboot required	N/A
Field validation	N/A

ModemInitString

Controller Support	DSCU, SSCU
Description	This parameter is intended for future use.
Type	String
Value(s)	*<blank>*
Default value	*<blank>*
Reboot required	No
Field validation	No

ModemSignalStrengthInterval

Controller Support	DSCU, SSCU
Description	Sets the interval to report the measured modem RF signal strength (dBm) to the Central System. Value is given in seconds. **Note :** This parameter is intended for future use.
Type	Integer
Value(s)	0 .. 10080
Default value	1440
Reboot required	No
Field validation	Yes

ModemWeakSignalStrengthLimit

Controller Support	DSCU, SSCU
Description	Whenever the modem RF signal strength becomes lower (weaker) than the given limit value (in dBm), a warning status notification is sent to the Central System. **Note** : Intended for future use.
Type	Integer
Value(s)	-113 .. -53
Default value	-90
Reboot required	No
Field validation	Yes

NodeId

Controller Support	DSCU, SSCU
Description	The node ID (external ID) of the Charging Station with which it identifies itself to the Central System.
Type	String
Value(s)	*<nodeId>*
Default value	NL_GFX_TEMP
Reboot required	Yes
Field validation	No

NominalAmperage

Controller Support	DSCU, SSCU
Description	Sets the default charge rate (standard situation, no smart charging). In case smart charging is active, the charging profile will be ignored the first *NominalPeriod* seconds of a charging session. During this time the EV will be charged with a *NominalAmperage* value.
Type	Float
Value(s)	6.0 .. 80.0
Default value	16.0
Reboot required	No
Field validation	Yes

NominalPeriod

Controller Support	DSCU, SSCU
Description	In case smart charging is active this value will set the time in seconds that the *NominalAmperage* value is active during the start of a charge session.
Type	Integer
Value(s)	30 .. 300
Default value	30
Reboot required	No
Field validation	Yes

NumberOfConnectors

Controller Support	DSCU, SSCU

Description	The number of physical charging connectors of this Charging Station.
Type	Integer (Read-only)
Value(s)	1 .. 2
Default value	1 (SSCU) 2 (DSCU)
Reboot required	No
Field validation	N/A

OfflineMode

Controller Support	DSCU, SSCU
Description	When set to True, the controller will not attempt to connect to the back-end system (anymore), nor will the controller perform reboot cycles when no connection to a Central System could be established. **Note**: making changes to a configuration parameter in offline mode is only possible using the Local Management website (please refer to the *Smart Charging Controller Installation Manual*). **Note** : available in version 4.3.x
Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	Yes
Field validation	Yes

OCPPProtocolUsed

Controller Support	DSCU, SSCU
Description	Version of the OCPP protocol that is used in communicating with the Central System service.
Type	String (read-only)
Value(s)	N/A
Default value	*<OCPP version>*
Reboot required	N/A
Field validation	N/A

OCPPProtocolsSupported

Controller Support	DSCU, SSCU

Description	List of OCPP protocol versions that are supported by the controller.
Type	CSL
Value(s)	["ocpp1.6", "ocpp1.5"]
Default value	[“ocpp1.6”]
Reboot required	Yes
Field validation	No

OverVoltageLimit

Controller Support	DSCU, SSCU
Description	When the grid power supply voltage exceeds the value in this configuration, a warning status notification is sent to the Central System. See also *UnderVoltageLimit.*
Type	Integer
Value(s)	20 .. 512
Default value	253
Reboot required	No
Field validation	Yes

PINCode

Controller Support	DSCU, SSCU
Description	This parameter is intended for future use.
Type	Integer
Value(s)	N/A
Default value	*<blank>*
Reboot required	No
Field validation	No

PlugPresentTimeout

Controller Support	DSCU, SSCU
Description	When connecting the EV to the Charge Point, the socket should be authenticated within this time out, given in seconds. Otherwise, no charging session will be activated. Authentication can be offering a valid IdTag or a remote start transaction initiated by the Central System. Note : See also *ConnectionTimeOut.*
Type	Integer
Value(s)	60 .. 300

Default value	90
Reboot required	No
Field validation	Yes

PrimaryConnection

Controller Support	DSCU, SSCU
Description	Primary connection type the controller will attempt to use. If this connection is lost and cannot be restored, the controller will attempt to use the secondary connection. Switching between both connections normally takes a few minutes. When connected to the secondary connection the controller will attempt every 4 hours to reconnect to the primary connection or sooner after connection failure of the secondary connection. Currently supported connection types are mobile (GPRS/LTE) and ethernet (wired, RJ45). Please refer to *SecondaryConnection.*
Type	Integer
Value(s)	0 (mobile), 1 (ethernet)

Default value	0
Reboot required	Yes
Field validation	Yes

ReserveConnectorZeroSupported

Controller Support	DSCU, SSCU
Description	**True :** The Charging Station supports reservations on Charge Point 0. (Either Charge Point 1 or 2 will be reserved for the customer). **False :** The Charging Station does not support reservations on Charge Point 0. Only reservation for a specific Charge Point (1 or 2) will be processed. This is the default setting for the controller
Type	Boolean (read-only)
Value(s)	True, False
Default value	False
Reboot required	No
Field validation	No

RFIDReader1Enabled

Controller Support	DSCU, SSCU
Description	Enables RFID reader 1 if set to true. **Note**: If *TaglessCharging* is used, this setting must be set to False. If two connectors are used and RFID reader 2 is disabled, RFID reader 1 is used for both connectors.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	Yes
Field validation	Yes

RFIDReader2Enabled

Controller Support	DSCU
Description	Enables RFID reader 2 if set to true. **Note**: If *TaglessCharging* is used, this setting must be set to false. If two connectors are used and RFID reader 2 is disabled, RFID reader 1 is used for both connectors.
Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	Yes
Field validation	Yes

Registered

Controller Support	DSCU, SSCU
Description	Indicates whether the controller was registered in the Central System. **Note :** This setting is inactive and intended for future use.
Type	Boolean (read-only)
Value(s)	True, False
Default value	True
Reboot required	N/A
Field validation	N/A

RelayHelperContactNormallyOpen

Controller Support	SSCU
Description	For supporting an additional safety detection for a correct functioning of the main contactor relay, a helper relay can be installed to monitor the main contactor / relay. This setting defines the default connection state of the helper relay contacts. By default, a Normally Open (NO) contact is expected (True). Using the feature is enabled with *SafetyRelaisEnabled*. Please refer to the *Smart Charging Controller Installation Manual* for more information.
Type	Boolean
Value(s)	True = NO (Normally Open) False = NC (Normally Closed)
Default value	True
Reboot required	No
Field validation	Yes

ResetRetries

Controller Support	DSCU, SSCU
Description	Number of times to retry an unsuccessful reset of the Charging Station. **Note :** This setting is inactive and intended for future use.
Type	Integer
Value(s)	0 .. 10
Default value	2
Reboot required	No
Field validation	Yes

RFIDMasterSlaveID

Controller Support	SSCU
Description	The master / slave functionality of the controller is configured with this parameter. By default, no RFID master/slave functionality is supported.
Type	Integer
Value(s)	0 = None (not active) 1 = Master 2 = Slave
Default value	0
Reboot required	Yes

Field validation	Yes

RFIDMasterSlaveEnabled

Controller Support	SSCU
Description	Enables the RFID reader master/slave support. The controller offers the possibility to couple two units and use only one RFId reader. The controller that has the actual reader connected is defined as master. The second controller must be the slave. Communication between both units is realized using the external RS485 bus (for connection info, please refer to the *SSCU Single Socket Smart Charging Controller Installation Manual*).
Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	Yes
Field validation	Yes

RootCertificate

Controller Support	DSCU, SSCU
Description	Root certificate for secure websocket. **Note:** This setting is inactive and is intended for future use.
Type	String
Value(s)	*<file name of root certificate>*
Default value	root.cert
Reboot required	Yes
Field validation	No

SafetyRelayEnabled

Controller Support	SSCU
Description	If set to True, additional safety detection for a correct functioning of the main contactor relay using a helper relay is activated. Also refer to *RelaisHelperContactNormallyOpen.* Please refer to the *Smart Charging Controller Installation Manual* for more information.
Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	No

Field validation	Yes

SecondaryConnection

Controller Support	DSCU, SSCU
Description	Secondary connection type the controller will attempt to use. If this connection is lost, the controller will retry to connect with the primary connection. When connected to the secondary connection the controller will attempt every 4 hours to reconnect to the primary connection or sooner after connection failure of the secondary connection. Currently supported connection types are mobile (GPRS/LTE) and ethernet (wired, RJ45). Please refer to *PrimaryConnection.*
Type	Integer
Value(s)	0 (mobile), 1 (ethernet),
Default value	1
Reboot required	Yes
Field validation	Yes

SendLocalListMaxLength

Controller Support	DSCU, SSCU
Description	Maximum number of identifications that can be send in a single *SendLocalList* request (from the Central System). **Note** : not implemented.
Type	Integer (read-only)
Value(s)	N/A
Default value	64
Reboot required	N/A
Field validation	N/A

SerialNumber

Controller Support	DSCU, SSCU
Description	The hardware serial number of the controller.
Type	String (read-only)
Value(s)	N/A
Default value	<*blank*>. On first time boot this value will be configured by the controller. The hardware serial number is retrieved and stored.

Reboot required	N/A
Field validation	N/A

SmartChargingEnabled

Controller Support	DSCU, SSCU
Description	Sets whether smart charging functionality is enabled. **True**: The actual allowed charge current is calculated/defined by a central system service. During the charge session the charge current may change based on received ChargeProfiles. **False**: All received ChargingProfiles will be ignored and charging will be done at nominal amperage (*NominalAmperage*). **Note:** Local Load Balancing will remain active in both situations, refer to Section 6 how this will work out in both situations.
Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	No
Field validation	Yes

SMSFactoryResetString

Controller Support	DSCU, SSCU
Description	The text which will prompt the controller to do a factory reset, which restores the original configuration settings. **Note:** This setting is currently inactive.
Type	String (read-only)
Value(s)	N/A
Default value	FACTRESET
Reboot required	N/A
Field validation	N/A

SMSManagementOriginator

Controller Support	DSCU, SSCU
Description	The phone number from which the controller will accept a (factory) reset command message. Include a country code if used. **Note**: This setting is currently inactive.
Type	String

Value(s)	*<tel number>*
Default value	*<blank>*
Reboot required	No
Field validation	No

SMSResetString

Controller Support	DSCU, SSCU
Description	The text which will prompt the controller to do a hard reset. **Note**: This setting is currently inactive.
Type	String (read-only)
Value(s)	N/A
Default value	RESET
Reboot required	N/A
Field validation	N/A

StartTransactionDelay

Controller Support	DSCU, SSCU
Description	The duration in seconds for which the authorization ID will be disabled after starting / stopping a transaction. This prevents the RFID reader from scanning the card twice unintendedly. **Note**: Intended for future use.
Type	Integer
Value(s)	0 .. 60
Default value	4
Reboot required	No
Field validation	Yes

StopTransactionOnEVSideDisconnect

Controller Support	DSCU, SSCU
Description	**True:** Stop transaction when cable is disconnected from EV **False:** Transaction will stay active after disconnection of EV. Reconnecting the EV will continue the charge session. Also see *UnlockConnectorOnEVSideDisconnect.*
Type	Boolean
Value(s)	True, False

Default value	False
Reboot required	No
Field validation	Yes

StopTransactionOnInvalidId

Controller Support	DSCU, SSCU
Description	Sets whether the Charge Point will stop an ongoing transaction when it receives a **non**-Accepted authorization status in a StartTransaction confirmation message from the Central System. When set to **False**, the EV is allowed to charge an amount of *MaxEnergyOnInvalidId* [Wh]. When a Charging Station was offline and a transaction has been allowed to start without authorization, this setting determines whether it may continue charging if the charger has restored connection and has found the authorization to be rejected by the service provider. **Note** : See also *MaxEnergyOnInvalidId* and *AllowOfflineTxForUnknownID*.
Type	Boolean
Value(s)	True, False
Default value	True
Reboot required	No
Field validation	Yes

SupportedFeatureProfiles

Controller Support	DSCU, SSCU
Description	A list of supported Feature Profiles. Please refer to the OCPP 1.6 for a more detailed description.
Type	CLS (read-only)
Value(s)	*<refer to the OCPP 1.6 for available profiles>*
Default value	Core, FirmwareManagement, SmartCharging
Reboot required	N/A
Field validation	N/A

SupportedFeatureProfilesMaxLength

Controller Support	DSCU, SSCU
Description	Maximum number of items in a *SupportedFeatureProfiles* Configuration Key.

Type	Integer (read-only)
Value(s)	N/A
Default value	6
Reboot required	N/A
Field validation	N/A

TaglessCharging

Controller Support	DSCU, SSCU
Description	**True**: No IdTag needs to be presented to start charging, in this case a valid ID must be configured under *TaglessIdTag1* and/or *TaglessIdTag2*. **False**: Charging will not commence without an RFID authorization. **Note** : the configurationparameters *RFIDReader1Enabled* and *RFIDReader2Enabled* need to be set matching the desired functionality. **Note**: For a proper setup of tag less charging make sure to configure *StopTransactionOnEVSideDisconnect* and *UnlockConnectorOnEVSideDisconnect.*
Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	Yes

TaglessIdTag1

Controller Support	DSCU, SSCU
Description	RFID tag number used for tag less charging on Charge Point 1.
Type	String
Value(s)	*<RFId tag identification number>*
Default value	TAGLESS1
Reboot required	No
Field validation	No

TaglessIdTag2

Controller Support	DSCU, SSCU
Description	RFID tag number used for tag less charging on Charge Point 2 (DSCU only). If this value is equal to the *TaglessIdTag1*, its configuration

	change will be rejected, each Charge Point needs to have a unique tag number.
Type	String
Value(s)	*<RFId tag identification number>*
Default value	TAGLESS2
Reboot required	No
Field validation	Yes

TempLowerOffsetDegrees

Controller Support	DSCU, SSCU
Description	Lower temperature offset (warning level) value (in Celsius) where the charger will set a reduced charge value of *TempLowerOffsetLimit* Ampere (A). To prevent a temperature overload condition of the Charging Station. See also *TempUpperOffsetDegrees*.
Type	Integer
Value(s)	40 .. 70
Default value	55
Reboot required	No
Field validation	Yes

TempLowerOffsetLimit

Controller Support	DSCU, SSCU
Description	The maximum charging value (Ampere) in case the *TempLowerOffsetDegrees* temperature limit has been reached. Value will be set for all Charge Points of the Charging Station.
Type	Integer
Value(s)	0.0 .. 80.0
Default value	16.0
Reboot required	No
Field validation	Yes

TempUpperOffsetDegrees

Controller Support	DSCU, SSCU
Description	Upper temperature offset (error level) value (in Celsius) where the charger will set a low charge value of *TempUpperrOffsetLimit* Ampere (A). To prevent a temperature overload condition of the Charging Station.
Type	Integer

Value(s)	40 .. 70
Default value	70
Reboot required	No
Field validation	Yes

TempUpperOffsetLimit

Controller Support	DSCU, SSCU
Description	The maximum charging value (Ampere) in case the *TempUpperOffsetDegrees* temperature limit has been reached. Value will be set for all Charge Points of the Charging Station.
Type	Integer
Value(s)	0.0 .. 80.0
Default value	0.0
Reboot required	No
Field validation	Yes

TransactionMessageAttempts

Controller Support	DSCU, SSCU
Description	How often the Charge Point should try to submit a transaction-related message when the Central System fails to process it.
Type	Integer
Value(s)	0 .. 10
Default value	5
Reboot required	No
Field validation	Yes

TransactionMessageRetryInterval

Controller Support	DSCU, SSCU
Description	Time (seconds) how long the Charge Point should wait before resubmitting a transaction related message that the Central System failed to process. In combination with the *MessageResponseTimeOut* the message response handling with the Central System is configured.
Type	Integer
Value(s)	0 .. 300
Default value	60

Reboot required	No
Field validation	Yes

UnderVoltageLimit

Controller Support	DSCU, SSCU
Description	When the grid power supply voltage exceeds the given value (lower), a warning status notification is sent to the Central System. See also *OverVoltageLimit.*
Type	Integer
Value(s)	20 .. 512
Default value	207
Reboot required	No
Field validation	Yes

UnlockConnectorOnEVSideDisconnect

Controller Support	DSCU, SSCU
Description	When set to True, automatically releases the connector at Charge Point side as soon as the EV side is being disconnected (unplugged). In case of a tag less charging setup be sure to set *StopTransactionOnEVSideDisconnect* also to true. This way making sure that disconnecting the cable on EV side also stops the transaction and unlocks the cable at EVSE side.
Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	No
Field validation	Yes

UseWebsocketSecure

Controller Support	DSCU, SSCU
Description	If enabled, connections can be made to a secure websocket server address (wss), by providing a secure websocket server address to the *BESPath* setting.
Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	Yes

Field validation	Yes

VoltageLimitCheckInterval

Controller Support	DSCU, SSCU
Description	Sets the interval rate in seconds between measuring and checking the *UnderVoltageLimit* and *OverVoltageLimit.*
Type	Integer
Value(s)	0 .. 86400 (= duration of 1 day)
Default value	600
Reboot required	No
Field validation	Yes

VoltageLimitMaxSequentialStatusNotification

Controller Support	DSCU, SSCU
Description	Sets the number of sequential error messages sent in case of an *UnderVoltageLimit* and *OverVoltageLimit* detection to prevent an overflow of error messages. The limiter will be reset after measuring one good value again.
Type	Integer
Value(s)	0 .. 128
Default value	5
Reboot required	No
Field validation	Yes

VoltageSensorVCC3V3IOCoupledConnector

Controller Support	SSCU
Description	The actual connector that is affected by the actions defined for guarding the 3.3V board power (0=all connectors). Please refer to Section **Error! Reference source not found.** for a detailed description about how to use the voltage sensor events.
Type	Integer
Value(s)	0 .. 1
Default value	0
Reboot required	Yes
Field validation	Yes

VoltageSensorVCC3V3IORecoverActions

Controller Support	SSCU
Description	Action to perform when the 3.3V board power reaches a safe operation condition (again). A *0*-value means: ignore the specified action. Values other than *0* indicates the priority/order of execution. Priority is defined in ascending order. Beware, these values must not exceed the number of actions available! By default, no actions are defined. Please refer to Section **Error! Reference source not found.** for a detailed description about how to use the voltage sensor events.
Type	List
Value(s)	*<List with action definitions>*
Default value	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]
Reboot required	Yes
Field validation	Yes

VoltageSensorVCC3V3IOTriggerActions

Controller Support	SSCU
Description	Action to perform when the 3.3V board power reaches a critical (low/high) value. A *0*-value means: ignore the specified action. Values other than *0* indicates the priority/order of execution. Priority is defined in ascending order. Beware, these values must not exceed the number of actions available! By default, no actions are defined. Please refer to Section **Error! Reference source not found.** for a detailed description about how to use the voltage sensor events.
Type	List
Value(s)	*<List with action definitions>*
Default value	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]
Reboot required	Yes
Field validation	Yes

VoltageSensorVCCBuffCoupledConnector

Controller Support	SSCU
Description	The actual connector that is affected by the actions defined for guarding the 12V Servo lock power buffer (0=all connectors). Please refer to Section **Error! Reference source not found.** for a detailed description about how to use the voltage sensor events.
Type	Integer
Value(s)	0 .. 1
Default value	0
Reboot required	Yes
Field validation	Yes

VoltageSensorVCCBuffTriggerActions

Controller Support	SSCU
Description	Action to perform when the 12V Servo lock power buffer reaches a save operation condition (again). A *0*-value means: ignore the specified action. Values other than *0* indicates the priority/order of execution. Priority is defined in ascending order. Beware, these values must not exceed the number of actions available! By default, no actions are defined. Please refer to Section **Error! Reference source not found.** for a detailed description about how to use the voltage sensor events.
Type	List
Value(s)	*<List with action definitions>*
Default value	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]
Reboot required	Yes
Field validation	Yes

VoltageSensorVCCBuffRecoverActions

Controller Support	SSCU
Description	Action to perform when the 12V Servo lock power buffer reaches a critical (low/high) value. A *0*-value means: ignore the specified action. Values other than *0* indicates the priority/order of execution. Priority is defined in ascending order. Beware, these values must

	not exceed the number of actions available! By default, no actions are defined. Please refer to Section **Error! Reference source not found.** for a detailed description about how to use the voltage sensor events.
Type	List
Value(s)	*<List with action definitions>*
Default value	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]
Reboot required	Yes
Field validation	Yes

VoltageSensorVCCMainCoupledConnector

Controller Support	SSCU
Description	The actual connector that is affected by the actions defined for guarding the 12V main power (0=all connectors). Please refer to Section **Error! Reference source not found.** for a detailed description about how to use the voltage sensor events.
Type	Integer
Value(s)	0 .. 1
Default value	0
Reboot required	Yes
Field validation	Yes

VoltageSensorVCCMainTriggerActions

Controller Support	SSCU
Description	Action to perform when the 12V main power reaches a save operation condition (again). A *0*-value means: ignore the specified action. Values other than *0* indicates the priority/order of execution. Priority is defined in ascending order. Beware, these values must not exceed the number of actions available! By default, no actions are defined. Please refer to Section **Error! Reference source not found.** for a detailed description about how to use the voltage sensor events.
Type	List
Value(s)	*<List with action definitions>*
Default value	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0,

	"HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]
Reboot required	Yes
Field validation	Yes

VoltageSensorVCCMainRecoverActions

Controller Support	SSCU
Description	Action to perform when the 12V main power reaches a critical (low/high) value. A *0*-value means: ignore the specified action. Values other than *0* indicates the priority/order of execution. Priority is defined in ascending order. Beware, these values must not exceed the number of actions available! By default, no actions are defined. Please refer to Section **Error! Reference source not found.** for a detailed description about how to use the voltage sensor events.
Type	List
Value(s)	*<List with action definitions>*
Default value	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0,

	"HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]
Reboot required	Yes
Field validation	Yes

WebSocketPingInterval

Controller Support	DSCU, SSCU
Description	A '0'-value disables controller side WebSocket *Ping/Pong*. In this case there either is no *Ping/Pong,* or the server initiates the *Ping* and controller responds with *Pong*. A positive value gives the number of seconds between the initiated *Ping*s.
Type	Integer
Value(s)	0 .. 255
Default value	210
Reboot required	Yes
Field validation	Yes
Field validation	Yes

3StageServo

Controller Support	DSCU, SSCU
Description	**False :** The connected servo lock actuator is a 2-stage type. Only open and closed (locked) states can be detected and processed in software. The standard Mennekens socket is an example of such an actuator: **True :** The connected servo lock actuator is a 3-stage type. Besides an open and closed (locked) state also a locked but no connector present situation can be detected and processed in software. The standard Phoenix socket is an example of such an actuator: See also *InversedServo* for setting up the desired state signaling level (Normally Open / Close contact definition). **Note** : Only controllers equipped with the correct hardware are able to support a 3-stage servo actuator. Next board hardware revisions support a 3-stage servo: **SSCU** : V3 (and higher) **DSCU** : E3 (and higher)
Type	Boolean
Value(s)	True, False
Default value	False
Reboot required	Yes
Field validation	Yes

5 CONFIGURING AND USING (EXTERNAL) SENSOR EVENTS

5.1 GENERAL

Only the SSCU has two external sensors at its disposal for connecting digital sensors. The DSCU does not support these sensors. This Section describes the usage and configuration of these sensors. The SSCU supports up to two external sensors.

For both external sensors a so-called scenario can be assigned. This means that on activation (and deactivation) of the sensor a configurable handling of pre-defined actions can be assigned.

For defining a complete sensor handling (scenario) there are six configuration parameters used for each external sensor. Names and functionality of these parameters (sensor **1**) are:

ExternalSensor1Type	Type of sensor that is connected to the external sensor input. More information is given in Section 5.2.
ExternalSensor1Name	Only active when the ExternalSensor1Type is set to "OTHER". The name entered here will be used for the *ExternalSensor1Type* definition. More information is given in Section 5.3.
ExternalSensor1CoupledConnector	Defines the Charge Point (Connector) for which the scenario should run when (de-)activated. More information is given in Section 5.4.
ExternalSensor1NormallyOpen	Defines the sensor state in Normally Open situation (the not activated state of the sensor). More information is given in Section 5.5.
ExternalSensor1TriggerActions	Defines the trigger actions of the scenario when the sensor enters the active situation (e.g. detected water). More information is given in Section 5.6.
ExternalSensor1RecoverActions	Defines the recover actions of the scenario when the sensor enters the normal situation. More information is given in Section 5.7.

for the second external sensor the corresponding configuration parameter names will have a '**2**' in their name, e.g. *ExternalSensor**2**Name*.

Only digital sensors (on/off) can be connected. Please refer to the SSCU *Smart Charging Control Installation Manual* and *SSCU wiring reference schematics* for more information about connecting the external sensors.

Next Sections explain the usage and working of the configuration parameters in more detail. All descriptions are given for external sensor '**1**' only. The other external sensor handling is identical, just use '**2**' as an identifier instead of '**1**'.

5.2 EXTERNAL SENSOR TYPE

The SSCU makes it possible to add special handling of external sensors. These sensors are connected to dedicated inputs of the SSCU and make it possible to run a script (scenario) in case the sensor is activated or deactivated. Defining the type of the sensor is merely used for sending the right sensor type status notification to the Central System.

Several predefined sensor types are available, but it is also possible to create a custom defined sensor handling (Other).

ExternalSensor1Type (integer):

TILT	1	Indicator that measures whether the Charging Station itself is tilted.
WATER	2	Sensor that measures the presence of water inside the critical zone of the Charging Station.
TEMPERATURE	3	Thermometer sensor for indicating a too high / low temperature within the measured zone of the Charging Station.
LIGHT	4	Indicator used to determine whether is day or night.
OPENDOOR	5	Signal sensor indicating the state of the access door or panel (opened / closed).
BUTTON	6	Button to start / stop a specific function.
GROUND_FAILURE	7	Sensor detecting possible Ground failure issues.
CCID	8	Special fault current detection sensor, refer to Section 7 for more info about the usage of a CCID sensor.
OTHER	99	Customizable sensor.

When choosing "**OTHER**" (99) for a sensor type, the parameter *ExternalSensor1Name* should contain the desired Type name for this sensor, please refer to Section 5.3. When using a pre-configured sensor type, all messages sent to the Central System related with activity for this sensor will be referenced using the sensor type identifier as name (e.g. "TILT"), the value of *ExternalSensor1Name* will be ignored.

5.3 EXTERNAL SENSOR NAME

In case a not pre-defined sensor type is used (ref Section 5.2) use parameter *ExternalSensor1Name* (string) to set the desired name type of the sensor, e.g. "MySensor". All messages sent to the Central System in relation to activity for this sensor will be referenced using the given name.

5.4 EXTERNAL SENSOR COUPLED CONNECTOR

This parameter defines for which Charge Point (connector) of the charge station the coupled scenarios will take place. For example, based on an "OPENDOOR" trigger action a scenario might be to stop all charging sessions currently active on the charge station. With parameter *ExternalSensor1CoupledConnector* (integer) this can be selected by setting it to "0" (all connectors).

0 : All connectors
1 : Connector 1

Note: Since the SSCU only has one connector, no difference will be noted between using a "0" or "1". When in the future the usage of external sensors will also become available for the DSCU, a specific usage of connector "1" and "2" will become possible.

5.5 EXTERNAL SENSOR NORMALLY OPEN

To make sure the scenarios will act on the correct sensor triggering the Normal Open (NO) sensor contact state must be defined. Depending on your connected sensor type, the untriggered situation (sensor is not activated) will be a closed or opened contact. With parameter *ExternalSensor1NormallyOpen* (boolean) this value can be set.

True: Input contact = High / Open.
False: Input contact = Low / Closed.

5.6 EXTERNAL SENSOR TRIGGER ACTIONS

With the trigger and recover actions (refer to Section 5.7) the actual scenario to be executed is specified. A scenario is a sequence of actions that will be executed when the associated sensor is triggered. With *ExternalSensor1TriggerActions* the desired sensor activation actions can be set. Therefore, several pre-defined actions can be selected on triggering of the sensor:

StatusNotification	The controller will send a status notification to the Central System indicating that the sensor has been triggered.
StopTransaction	A running charge session will be stopped on associated Charge Point (connector).
UnlockConnector	The Charge Point (connector) will be unlocked.
SetError	The Charge Point (connector) will be set to error state. Resulting in a red LED indication and withholding all operative actions (e.g. starting a charge session).
ClearError	The Charge Point (connector) will be set to normal operating condition. Resulting in a green LED indication and making it available for starting a charge session.
HardReset	The charge station will execute a software reset and will be initialized as being switched off and on again.
SetLedIntensityLow	Sets the LED to the low intensity mode.
SetLedIntensityHigh	Sets the LED to the high intensity mode.

Note : Setting the LED intensity as part of a scenario will use the actual values as defined with configuration parameters *LEDIntensityLow* and *LEDIntensityHigh*.

Activating actions that need to be executed is done by given them a priority indicator. Selected actions will then be executed following the given priority. This way an exact pre-defined scenario can be constructed.

ExternalSensor1TriggerActions (list) **:**

```
[
  [0, "StatusNotification"],
  [1, "StopTransaction"],
  [2, "UnlockConnector"],
  [3, "SetError"],
  [0, "ClearError"],
  [0, "HardReset"],
  [3, "SetLEDIntensityLow"],
  [0, "SetLEDIntensityHigh"]
]
```

Using the '0' as priority value means it will not be executed.

When using the same priority value for multiple actions the action will execute based on the priority value in relation to other priority action values, but within the same priority no specific order is guaranteed. It is possible to give every action its own priority, thus guaranteeing an exact specified order.

A lower priority value prevails over a higher value.

In the above given *ExternalSensor1TriggerActions* example, after the sensor is triggered next actions will be executed in following order:

- Charge session will be stopped.
- The connector will be unlocked.
- The error state will be initiated, and the LED will be set to the value of configuration item *LEDIntensityLow* (see Section 4.4). There will be no specific order for the latter two.

5.7 EXTERNAL SENSOR RECOVER ACTIONS

With *ExternalSensor1RecoverrActions* the sensor de-activation actions will be executed. Several pre-defined actions can be selected to be executed on triggering, please refer to Section 5.6 for clarification. The handling of a recovery action is the same as defined for a trigger action, also, please refer to Section 5.6.

5.8 VOLTAGE SENSOR EVENTS

Besides the handling of events of external (digital) sensor inputs, it is also possible to define (recover) actions on certain power supply events. This is handled in the same way as external (digital) sensors are handled.

Three important voltages of the controller can be monitored and used for event handling when needed:

VMAIN	Voltage monitoring of the main power supply (12 Volt).
3V3IO	Derived printed circuit board voltage (3.3 Volt).
VBUFF	Voltage of the servo lock power buffer (12 Volt).

Defining trigger and recover actions definitions are handled just like the external sensor actions.

Note : In normal operating situations, specific voltage sensor handling is not needed.

5.9 EXAMPLES

This Section gives a view examples to clarify the usage of the external sensor scenario handling.

5.9.1 TILT DETECTION

In the case of a Charging Station getting hit by E.g. a vehicle, it is likely that the Charging Station is damaged, and usage of its Charge Points might become dangerous. By installing a tilt sensor, the controller can now detect this situation and take appropriate safety measures, meaning stop al running transactions (no energy delivery anymore) and make a notification to the Central System. Also, all charging attempts should be blocked.

We will connect the tilt sensor to external sensor number 1. Let's assume we use a sensor with a *Normally Open* output contact.

Configuration definition to accomplish these requirements:

ExternalSensor1	**1**
Type	(TILT)
ExternalSensor1 NormallyOpen	**True** (Input contact = High / Open)
ExternalSensor1	**0**
CoupledConnector	(All connectors)
ExternalSensor1	**<empty>**
Name	(pre-configured sensor, uses pre-defined name "TILT")
ExternalSensor1	[
TriggerActions	**[3, "StatusNotification"],** **[1, "StopTransaction"],** [0, "UnlockConnector"], **[2, "SetError"],** [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]

ExternalSensor1 RecoverActions	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], [0, "SetLEDIntensityHigh"]]

5.9.2 LED INDICATOR DIMMER

In the case of a Charging Station being in a crowded environment (near houses) the illumination of the status LED might be annoying during nighttime. Using a light sensor now makes it possible to dim the LED's during darkness.

We will connect the light sensor to external sensor number 2. Let's assume we use a sensor with a *Normally Closed* output contact.

Configuration definition to accomplish these requirements:

ExternalSensor2	**4**
Type	(LIGHT)
ExternalSensor2 NormallyOpen	**False** Input contact = Low / Closed
ExternalSensor2	**0**
CoupledConnector	(All connectors)
ExternalSensor2	**<empty>**
Name	(pre-configured sensor; uses pre-defined name "LIGHT")
ExternalSensor2 TriggerActions	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], **[1, "SetLEDIntensityLow"],** [0, "SetLEDIntensityHigh"]]
ExternalSensor2 RecoverActions	[[0, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], **[1, "SetLEDIntensityHigh"]**]

5.9.3 USER DEFINED SENSOR

To connect a specific sensor that is not in the pre-defined list of sensor types we still can connect this sensor and give it our own name. Suppose we are using a sensor called "*MySensor*". In the case of activation of the sensor we want next actions to be processed:

- Stop a running charge session (transaction)
- Disconnect the cable
- Set the Charge Point into error mode
- Notify the Central System

After returning to the in-active (save) situation next recovery actions must be executed:

- Reset the Charge Point (clear error situation)
- Notify the Central System

We will connect our sensor to external sensor number 2. Let's assume "*MySensor*" is equipped with a *Normally Closed* output contact.

Configuration definition to accomplish these requirements:

ExternalSensor2	**99**
Type	(OTHER)
ExternalSensor2 NormallyOpen	**False** Input contact = Low / Closed
ExternalSensor2	**1**
CoupledConnector	(connector 1 only (SSCU))
ExternalSensor2 Name	**MySensor**

ExternalSensor2 TriggerActions	`[`
	`  [3, "StatusNotification"],`
	`  [1, "StopTransaction"],`
	`  [2, "UnlockConnector"],`
	`  [3, "SetError"],`
	`  [0, "ClearError"],`
	`  [0, "HardReset"],`
	`  [0, "SetLEDIntensityLow"],`
	`  [0, "SetLEDIntensityHigh"]`
	`]`

ExternalSensor2 RecoverActions	`[`
	`  [2, "StatusNotification"],`
	`  [0, "StopTransaction"],`
	`  [0, "UnlockConnector"],`
	`  [0, "SetError"],`
	`  [1, "ClearError"],`
	`  [0, "HardReset"],`
	`  [0, "SetLEDIntensityLow"],`
	`  [0, "SetLEDIntensityHigh"]`
	`]`

6 LOAD BALANCING

6.1 INTRODUCTION

Charging stations are connected to the grid and are only allowed to draw a certain amount of energy from it. It is important to configure the charging station accordingly to make sure no overload situation can occur, resulting in switching of the complete Charging Station. Thus, making sure that within this defined maximal allowed grid connection settings the actual connected EV's will get the most optimal amount of energy available from the charging point. For this the controller uses a *Local Load Balancing* algorithm. The algorithm realizes the most optimal distributed energy for both connected EV's taking in account the allowed total energy the charging station can deliver.

The controller also supports *Smart Charging* (OCPP1.6). If activated, the Central System decides what charging profile will run on a Charge Point and how the available energy will be distributed to connected EV's. Local Load balancing will in this case only check for possible overload situations. Using the Smart Charging facility makes it possible to arrange load balancing for a (large) group of Charging Stations. *Local Load Balancing,* however, will only affect the Charge Points of the Charging Station itself. It has no knowledge of other Charging Station charge conditions.

6.2 LOCAL LOAD BALANCING

With next parameters the working of the Local Load balancing algorithm is defined (in case the *Smart Charging* functionality is activated, please refer to Section 6.3).

MaxCurrentConnection	The maximum allowed current on each Charge Point. Value is given in Ampere [A]. In case of using a DSCU, each of the two Charge Points will have the same maximum allowed current.
ChargePointCurrentLimit	The maximum current that the Charging Station can draw from the grid. This value defines the maximum current for the whole Charging Station, thus all enabled Charge Points (sockets) combined.
MinCurrentConnection	The result of the local load balancing calculations will use this setting as lower bound value for setting the allowed charge rate. The value must be smaller than *ChargePointCurrentLimit* / 2.

Note : Only the DSCU supports actual load balancing.

Next are the rule definitions of the *Local Load Balancing* algorithm that will adapt the actual allowed charging rate of connected EV's if needed.

1. *Local Load balancing* will only become active when the combined max. load per active Charge Point (*MaxCurrentConnection*) exceeds the EVSE max load (*ChargePointCurrentLimit*).
2. *Local Load Balancing* reduces the maximum allowed load for all active sockets simultaneously by the same reduction factor (0.1 Ampere

usually). It does so in a loop until the combined max. load per active connector does not exceed the EVSE max load anymore.

3. Beside the max. current per Charge Point, there is also a minimum current (*MinCurrentConnection)* value per Charge Point. While reducing the maximum current it is considered that the allowed max. current must not fall below this minimum value. This is done because certain EV's do not start charging when a certain threshold of current cannot be provided (a save lower value proves to be 12.5A).
4. The reduction loop is terminated when no connector can reduce its max current anymore (reached the *MinCurrentConnection* value). This can lead to the situation that the combined max current of all connectors still exceeds the max. current for the Charging Station. To prevent damage to the grid or electrical installation, charging will now be done in turns, using the maximum allowed current.
5. The algorithm will automatically increase the available energy of a Charge Point when the EV connected to the other Charge Point indicates to be ready witch charging (no energy demand anymore for 15 minutes). All energy will become available for the remaining Charge Point.

6.3 SMART CHARGING (CONTROLLED BY THE CENTRAL SYSTEM)

When Smart Charging has been enabled (*SmartChargingEnabled*) on a Charging Station, the Central System is responsible for the desired distribution of energy over the Charging Stations taken part in this Smart Charging cluster. The Central System knows after all the specific goal(s) to achieve and will create the correct charging profiles for all individual Charge Points. The Central System is responsible for sending these profiles in time to the Charging Stations. The Charging Stations are responsible for following up the requested energy delivery to the connected EV's'. Based on the resulting measurements and other input parameters the Central System continuously recalculates the most optimal energy distribution and notifies the participating Charging Stations of any desired changes.

Local Load Balancing by the way, will always stay active, this way assuring that an overload situation of the local Charging Station will never occur. When for some reason an erroneous charging profile has been send resulting in a possible dangerous overload situation the standard Load Balancing algorithm will scale down the energy delivery to save operating conditions.

7 USING A CHARGE CURRENT INTERRUPTING DEVICE (CCID)

For additional safety the Single Socket Control Unit (SSCU) offers the possibility to integrate the usage of a Charging Current Interrupting Device (CCID) developed by *Western Automation*. The RCM14-0*x* range is ideally suited for fault current monitoring on AC installations (detecting both AC and DC faults). This way guaranteeing safe operation of the system. Only the SSCU supports the usage of a CCID device.

7.1 HOW THE CCID BASICALLY WORKS

Before the CCID functionally can be used the actual setup of the hardware needs to be adapted for this. Please refer to the SSCU *Smart Charging Control Installation Manual* and *SSCU wiring reference schematics* for detailed information about connecting a CCID to the SSCU. Also, a dedicated configuration is needed to connect the SSCU with the CCID and make all safety functionality available.

Before a charge session is started, all power cabling (N, L1, L2 and L3) are tested by the CCID for a possible unsafe situation (AC/DC fault current). When no fault situation is found, the charge session will continue. However, when during the charge session a fault current is detected, the SSCU is signaled at once by the CCID and the energy delivery to the EV will be stopped instantly, preventing any dangerous situation.

The CCID safety functionality consist basically out of three different stages:

- Test starting conditions for a safe charge session.
- Continuous monitoring safety during a charge session.
- Auto-test for resetting an interrupted charge session after a fault current problem is solved / has disappeared. This will take place every 15 minutes, with a maximum of three times for one charge session.

Figure 3 picture shows the functional connection between the CCID component and the SSCU:

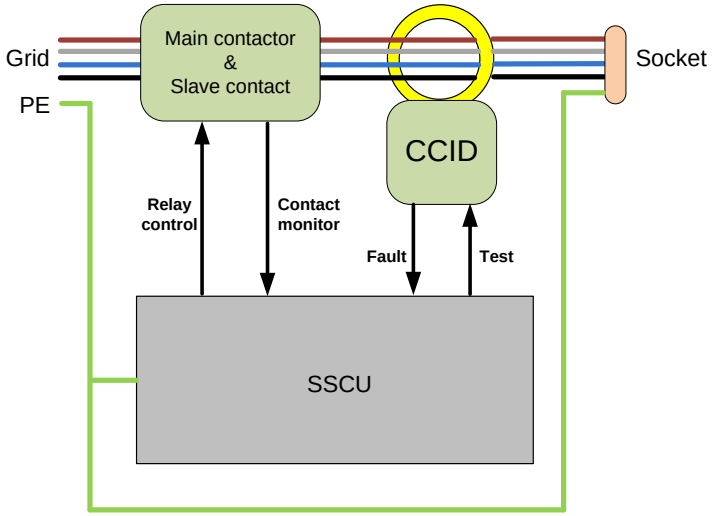


Figure 3 Functional diagram of integrating a CCID with the SSCU.

The CCID connects with the SSCU using two signals (*Test* and *Fault*). Besides the standard relay control line (*Relay control*) also a signal is needed for monitoring the actual relay state (*Contact monitor*).

7.1.1 TESTING CHARGE START CONDITION

Before a charge session will be initiated, the SSCU performs a test whether the connected CCID is correctly working. Therefore, the *Test* signal is activated and must be followed by raising a *Fault* signal. The SSCU will monitor and validate this behavior. Only when the test succeeds the charge session may continue.

7.1.2 CONTINUOUS MONITORING DURING CHARGING

During operation (charge session active) the CCID will continuously monitor the electric safety of the Charging Station. In case of a detected fault current (AC or DC) the CCID will indicate the problem immediately by raising the *Fault* signal. The SSCU detects this error condition and will directly open the main relay that provides the electric energy to the vehicle (no energy delivered, safe situation).

7.1.3 AUTO TEST AND RESET

When during a charge session the CCID has tripped (a fault condition was detected) the charge session is interrupted, and no energy is delivered to the EV. For convenience of the driver the SSCU will sense the electric system after 15 minutes to see whether the unsafe condition still exits. If so, thee system remains powerless. If no problem is detected anymore the power delivery will be restored. Sensing for safe conditions will only be repeated 3 times in a row, when still not safe, the charge session will be stopped completely, and the car needs to be disconnected before it is possible to start a new charging session.

7.2 CONFIGURING THE CCID FUNCTIONALITY

To enable the CCID functionality a specific configuration must be activated for external sensor **1 (only sensor 1!)**. Next settings overview shows the configuration to use a CCID device connected to sensor 1:

ExternalSensor1Type	**8** (CCID)
ExternalSensor1Name	*n.a.*
ExternalSensor1NormallyOpen	False
ExternalSensor1CoupledConnector	0
ExternalSensor1TriggerActions	[[1, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"], [0, "HardReset"], [0, "SetLEDIntensityLow"], [1, "SetLEDIntensityHigh"]]
ExternalSensor1RecoverActions	[[1, "StatusNotification"], [0, "StopTransaction"], [0, "UnlockConnector"], [0, "SetError"], [0, "ClearError"],


```
  [0, "HardReset"],
  [1, "SetLEDIntensityLow"],
  [0, "SetLEDIntensityHigh"]
]
```

Please refer to Section 5 for more details for information how to configure an external sensor in general.

8 STATUS AND ERROR MESSAGES

8.1 HARDWARE SETUP CONTROLLER

Both the DSCU (Double Socket Control Unit) and SSCU (Single Socket Control Unit) have two separate Central Processing Units (CPU's) to take care of all available functionality within the controller (and therefore the Charge Point(s)). Figure 4 shows the high-level architectural design of the control unit:

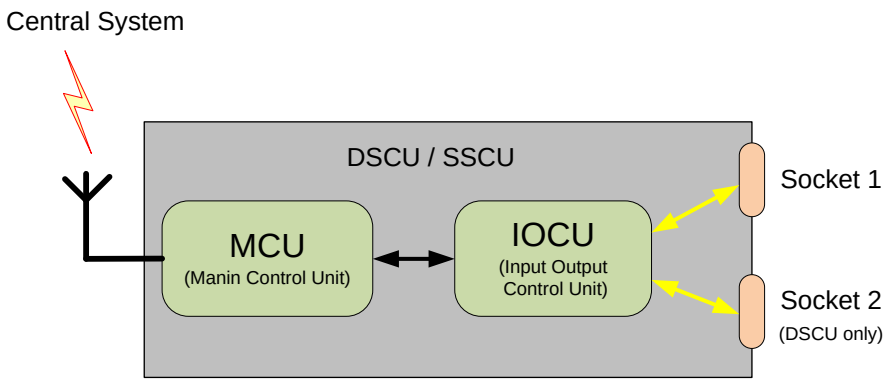


Figure 4 High-over architectural design of the DSCU and SSCU.

The responsibilities of for both CPU's are roughly divided as follows:

MCU : Maintains connection with the Central System, handles the standard charging workflow (transactions). Supervises the available RF tagId reader(s), kWh meter(s) and optional extra safety control components/functions (E.g. overload protection).

IOCU : Responsible for all EV connection signals like detecting EV presence and type of charging cable. The *IOCU* handles all *Mode3* charging communication with the connected EV. The *Mode3* charging protocol regulates the way the Charge Point and EV should "talk" with each other, needed for starting a charge session. The *IOCU* also controls the delivery of electric energy by switching the contactor (power relay) when ready for charging.

In case of a warning or error, the corresponding OCPP message will be created and sent to the Central System (responsibility of the *MCU*), refer to Section 8.2.

Specific *IOCU* related messages (errors) are sent to the Central System using the standard OCPP error formatting. The *errorCode*, *info* and *vendorErrorCode* fields will in that case give specific information about the problem detected by the *IOCU* (refer to Sections 8.4 and 8.5 for detailed information about these errors).

The next OCPP message is an example of how a detected EV diode error is sent to the Central System:

```
'info': 'IOCU',
'status': 'Faulted',
'vendorErrorCode': 'ERR_DIODE_FAILURE',
```



```
'connectorId': 1,
'errorCode': 'EVCommunicationError',
'timestamp': '2019-04-26T09:08:49Z'
```

The Central system can use this message to inform the system operator about this problem / situation.

8.2 OCPP CHARGE POINT ERROR DESCRIPTIONS

The errors described in this Section are the standard OCPP status (error) messages as described in the OCPP documentation (ref to Section 1.2.2). In case of the occurrence of such a Charge Point error, the controller will send it to the Central System. An OCPP error (status) message contains several fields that will be filled according the issue that has been encountered. An example of such a message:

```
'info': 'IOCU',
'status': 'Faulted',
'vendorErrorCode': 'ERR_SERVO_EXPECTED_LOCK',
'connectorId': 1,
'errorCode': 'EVCommunicationError',
'timestamp': '2019-04-26T11:08:49Z'
```

The *errorCode* field contains the actual type of error. How the information is displayed to the operator depends on the Central System software interface.

Description of the available error codes:

errorCode	Description
ConnectorLockFailure	Failure to lock or unlock the connector. **Possible causes and solutions:** • A broken locking mechanism • Dirt • Incorrect wiring of the connector • A small object inside the lock-cavity of the connector of the user's EV charging cable or a small object inside either the connector or socket which doesn't allow the plug to be fully inserted into the socket.
EVCommunicationError	Communication failure with the EV. This might be a *Mode3* or other communication protocol. problem. Charging will be aborted or suspended until the problem is solved. **Possible causes and solutions:** If this occur only once or more times with the same user, it is likely that there is an error in the EV. If this always happens at a specific charger it could be that it is a station with a tethered cable in which an incorrect resistor value is used. In other cases, a more thorough analysis is needed by second line support. In this case it is good to download the diagnostics file (please refer to Section 0).
GroundFailure	Ground fault circuit (CCID) interrupter has been activated.

HighTemperature	Temperature inside the Charge Point is too high. Please refer to *TempUpperOffsetDegrees* in Section 4.4.
InternalError	Error in internal hard- or software component. More information might be available in the *vendorErrorCode* field. This does not reflect errors originating from any connected peripheries (kWh meter, IdTag reader, etc) **Possible causes and solutions:** Possible problem with the MCU and/or IOCU hardware or inter communication. When errors remain the controller might have a defect and more detailed inspection might be needed.
LocalListConflict	The authorization information received from the Central System conflicts with the *LocalAuthorizationList*. **Note :** not used / active.
NoError	No error to report. As soon as a specific error / warning is solved the *info* field will give information about the cause that has been solved. E.q. "TEMPERATURE HIGH LIMIT" means that a previous detected temperature to high situation has been resolved and that the error / warning is no longer active.
OtherError	Type of error that has no "own" error handling. These kinds of errors are not specifically handled and have no dedicated error messages. More information might be available in the *vendorErrorCode* field. **Possible causes:** Voltage error, input bounce.
OverCurrentFailure	The current overload protection has tripped. **Possible causes and solutions:** The EV is charging on a higher amperage value than is allowed / expected. As a result of this the charge session will also be interrupted (no energy delivered to the EV anymore).
OverVoltage	The grid voltage has risen above an acceptable level. Please refer to *OverVoltageLimit* in Section 4.4.
PowerMeterFailure	Failure to read power meter. No valid kWh reading is available. **Possible causes and solutions:** • Wrong type of power meter is configured (please refer to Section 4.4 (*MeterType*). Check the value with the meter connected. • The actual configuration of the connected power meter does not match the expected communication settings. Please refer to the corresponding *Controller installation manual*, Section 2.7.3 (**DSCU**) or Section 2.5.1 (**SSCU**) and set the

	communication parameter as defined. • The RS485 communication wiring is faulty (the “Data P (+)” and “Data N (-)” of the RS485 signals are mixed up). • The power meter itself is faulty.
PowerSwitchFailure	Failure to control the power switch. Only relevant for the **SSCU**, please refer to the corresponding *SSCU Controller installation manual*, Section 2.10.2. **Possible causes and solutions:** • The main contactor relays guard detects a power switch failure. • The wiring for supporting the detection of a failing power switch is not correctly implemented causing an invalid error message.
ReaderFailure	Failure with RF idTag reader. **Possible causes and solutions:** • No valid RF IdTag reader hardware is connected to the controller, please connect a GFX RF IdTag card reader. • The RF IdTag reader hardware is malfunctioning, replacement of the reader should fix the problem.
ResetFailure	Unable to perform a reset. **Possible causes and solutions:** When an OTA (over the air) software update is initiated a remote reset request will be rejected during the time the new software version is starting up and has evaluated correct functioning of “itself”.
UnderVoltage	The grid voltage has dropped below an acceptable level. Please refer to *UnderVoltageLimit* in Section 4.4.
WeakSignal	Wireless communication device reports a weak signal. **Note** : Will be implemented in the future.

In case of a specific *IOCU* error (the *info* field of the error message has the value “IOCU”), the *vendorErrorCode* field will contain the actual error information given by the *IOCU* (as described in Section 8.4).

8.3 OCPP CHARGE POINT STATUS DESCRIPTIONS

The controller will also send status notifications regarding the actual state of the Charge Point at any time the status of a Charge Point (connection) changes. These notifications are standard OCPP messages as defined in the OCPP documentation (ref to Section 1.2.2). An example of such a message:

```
'info': 'Connector',
'status': 'Charging',
```



```
'vendorErrorCode': '',
'connectorId': 1,
'errorCode': 'NoError',
'timestamp': '2019-04-26T10:30:37Z'
```

Next status messages are defined:

status	Description
Available	When a Charge Point becomes available for a new user.
Charging	When the contactor (power relay) of a Charge Point closes, allowing the vehicle to charge.
Faulted	When a Charging Station or Charge Point has reported an error and is not available for energy delivery.
Finishing	When a charging session has stopped at a Charge Point, but the Charge Point is not yet available for a new user, e.g. the cable has not been removed.
Preparing	When a Charge Point is no longer available for a new user, but no charging session is active. Typically, a Charge Point is occupied when a user presents a tag or inserts a cable.
Reserved	When a Connector becomes reserved as a result of a Reserve Now command (Operative).
SuspendedEV	When the Charge Point is ready to deliver energy but the contactor (power relay) is open, e.g. the EV is not ready for charging or is a fully charged.
SuspendedEVSE	When the contactor (power relay) of a Charge Point opens upon request of the Charging Station, e.g. due to a smart charging restriction or as the result of *StartTransaction.conf* indicating that charging is not allowed.
Unavailable	When a Charging Point becomes unavailable as the result of a Change Availability command form the Central System or an event upon which the Charge Point transitions to unavailable at its discretion. Upon receipt of a Change Availability command, the status may change immediately, or the change may be scheduled. When scheduled, the Status Notification is sent when the availability change becomes effective.

8.4 IOCU ERROR MESSAGES

In case of a specific IOCU error the *vendorErrorCode* will contain the error as listed below and is described in detail in Section 8.5.

8.5 IOCU ERROR MESSAGES OVERVIEW

This Section lists all available error message the *IOCU* unit can generate. The actual errors will be sent to the Central System using a status notification (error / warning) message. Please refer to Section 8.1 for an explanation about the general setup of a controller and what the function and responsibilities are of an *MCU* and *IOCU*.

The *IOCU* implements the Charge Point / EV communication protocol according to IEC61851. This specification describes the so called *Mode3* communication protocol between Charge Point and EV and is the base for all state handling. Quite a few *IOCU* error messages therefore find their cause of triggering for not complying with this specification.

ERR_CCID_TRIPPED

Controller Support	SSCU
Description	When the Charge Point is equipped with a specific *CCID* (Charge Current Interrupting Device), the presence of residual-current (leakage current) can be detected. Both AC and DC *CCID* devices are supported. The moment this device detects a possible dangerous situation) mentioned error message will be send to the MCU. On detection the energy delivery will be stopped at immediately. **Note:** Please refer to the *GreenFlux SSCU_Smart Charging Controller installation manual* for more information about usage and installation.
Possible cause / trigger	Internal error that should normally not occur.

ERR_CHANGE_PWM_NOT_YET_ALLOWED

Controller Support	DSCU, SSCU
Description	The *MCU* processor asked for a charge rate not yet available. This error will occur when setting the charge rate too soon to a high rate (>40A). When using high charge rates (>40A) the charge rate will be reached in two separate steps to make a valid diode test possible (*Mode3*). After the diode test is passed, automatically the higher selected charge rate will become active. Switching to the higher current rate will take maximal 5 seconds. If the MCU tries to set a new high rate value before this process is finished, given error is raised.
Possible cause / trigger	Internal error that should normally not occur.

ERR_DIODE_FAILURE

Controller Support	DSCU, SSCU
Description	During an active charging session, the controller performs a real-time diode availability check within the connected EV. If the controller fails to detect the presence of a diode, the controller will give this error. The running charge session will be aborted immediately.
Possible cause / trigger	• Failure of the charging cable. • A defective EV diode. • A bad PWM signal (indicating problems with the wiring). • Damaged or dirty connector sockets.

ERR_EXT_SENS1_BOUNCED

Controller Support	SSCU
Description	The controller detects multiple state changes for the external sensor input 1, resulting in stopping the process of the state changes (bouncing). After 60 seconds checking state changes will automatically be activated again.
Possible cause / trigger	• Cabling became disconnected or damaged. • Connected sensor shows erroneous behavior. • Sensor source gives instable signal.

ERR_EXT_SENS2_BOUNCED

Controller Support	SSCU
Description	The controller detects multiple state changes for the external sensor input 1, resulting in stopping the process of the state changes (bouncing). After 60 seconds checking state changes will automatically be activated again.
Possible cause / trigger	• Cabling became disconnected or damaged. • Connected sensor shows erroneous behavior. • Sensor source gives an instable signal.

ERR_HEARTBEAT

Controller Support	DSCU, SSCU
Description	The heartbeat signal between *MCU* and *IOCU* processor failed. When the heartbeat between both units is failing the correct functioning of the controller is no longer guaranteed. When this situation occurs, any running charge session will be stopped. The LED indicator(s) will turn RED.
Possible cause / trigger	Normally this message should not occur at any time. If it does, this means there might be a serious hardware problem with the controller itself.

ERR_MODE3_PILOT_SHORTED

Controller Support	DSCU, SSCU
Description	The Control Pilot signal (CP) between Charge Point and EV is shorted to ground (*Mode3*, state

E error). The Charge Point will indicate the error with a red LED.

Possible cause / trigger

- Possibly a faulty charge cable.
- Charge Point socket cabling issue.
- EV connector problem.

ERR_NOT_LOCKED_FOR_POWER

Controller Support DSCU, SSCU

Description Before the controller can deliver any energy to the EV (at the start of a charge session) the correct state of the servo lock will be checked (must be closed). During delivering of energy the servo-lock should also always be locked. When an unexpected open state of the lock is detected the power supply will be stopped immediately. The error situation is indicated by a red LED.

Possible cause / trigger

- A problem with the connector.
- The cable is not properly connected.
- the servo might be defective.
- the servo failed to lock during connection (dirt, ice).
- The servo lock pin is (partly) obstructed.
- There is a small object in either the socket or the charge plug.

ERR_PLUG_EXPECTED_LOCKED

Controller Support DSCU, SSCU

Description During the preparing stage of a charge session the controller detected an unexpected lock failure.

Possible cause / trigger

- Servo (gearbox) lock is defected, the charge plug could not be locked within the time allowed.
- Charge plug is not correctly fitted into the connector. Actual locking is obstructed.
- Servo lock contains dirt (sand, small stones, ice etc.). Proper locking is not possible.

ERR_PLUG_EXPECTED_PRESENT

Controller Support DSCU, SSCU

Description During the preparing stage of a charge session the controller detected an unexpected plug absence.

Possible cause / trigger	• Servo (gearbox) lock is defected, the charge plug could simply be removed. • The lock has been forced to open and the charge plug was removed. • Defective charging cable (locking notch is damaged).

ERR_RELAYS_MULTIPLE_SWITCHES

Controller Support	DSCU, SSCU
Description	When an EV switches between different *Mode3* states far too frequently, the charging session is interrupted. The limit is set at 10 switches per 5 seconds (not configurable). Charge Point will indicate the error with a red LED.
Possible cause / trigger	The *Mode3* state communication between the EV and the Charge Point might be unstable caused by: • A bad charging cable. • Bad Charge Point connector or bad contact points. **Note**: If it occurs on a frequent basis, it could also be a hardware problem on the controller.

ERR_REQ_CRC_WRONG

Controller Support	DSCU, SSCU
Description	Communication between the *MCU* and *IOCU* is (temporarily) disturbed, messages between them cannot be processed correctly. The CRC message validation was wrong / unexpected.
Possible cause / trigger	Can occur sporadically. If it happens on a more frequent basis, it could indicate a hardware problem on the DSCU/SSCU circuit board.

ERR_RX_BUFFER_OVERFLOW

Controller Support	DSCU, SSCU
Description	The incoming command buffer of the *IOCU* is flooded (The *MCU* sends too many messages in a short period of time). This might result in an incorrect functioning of the Charging Station.
Possible cause / trigger	Possible cause might be a loose cable inside the Charging Station of some sort, generating lots of signal changes. **Note**: In normal conditions this error should never occur.

ERR_SERVO_EXPECTED_LOCK

Controller Support	DSCU, SSCU
Description	The controller expects the servo lock to be locked but detects an unlock situation.
Possible cause / trigger	• Servo (gearbox) lock is defected. • The lock has been forced to open. • Servo status indication signal is defect (cabling). • The Charge plug is not correctly fitted into the connector and obstructs the locking mechanism of the servo lock. • Servo lock contains dirt (sand, small stones, ice etc.). Proper locking might not be possible.

ERR_SERVO_EXPECTED_UNLOCK

Controller Support	DSCU, SSCU
Description	The controller expects the servo lock to be unlocked but detects a locked situation.
Possible cause / trigger	• Servo (gearbox) unlock is defected. • The lock has been forced to close. • Servo status indication signal is defect (cabling). • The Charge plug is not correctly fitted into the connector and prevents the lock from being opened. • Servo lock contains dirt (sand, small stones, ice etc.). Proper unlocking is not possible.

ERR_SERVO_LOCK_FAILED

Controller Support	DSCU, SSCU
Description	The servo lock failed to lock or unlock.
Possible cause / trigger	• Servo lock does not work properly; it may be blocked or dirty. • The position indicator of the servo may be defect or badly wired. • The servo may not be able to enter the right position (not reaching closing or opening position). • A problem with the connector. • There is a small object in either the socket or the charge plug.

ERR_SOCKETID_NOT_ALLOWED

Controller Support	DSCU, SSCU

Description	The *IOCU* processor received a command from the *MCU* processor for a not supported or expected socket Id.
Possible cause / trigger	Internal error that should never occur. Possible reason might be a faulty update procedure.

ERR_TEMPERATURE_SOCKET

Controller Support	DSCU
Description	The socket temperature exceeds the max allowed value. **Note**: Will be implemented in the future.
Possible cause / trigger	The external temperature sensor mounted in the charge socket detects a high temperature. This can only occur if an external temperature sensor is in fact connected in the socket.

ERR_TEST_CMD_NOT_ALLOWED

Controller Support	DSCU, SSCU
Description	Controller production related error message.
Possible cause / trigger	Purely used during *IOCU* test mode, this message should never occur during operation.

ERR_TX_BUFFER_OVERFLOW

Controller Support	DSCU, SSCU
Description	The message transmit buffer of the *IOCU* is flooded (The IOCU has too many messages waiting to be send to the *MCU*).
Possible cause / trigger	Possible cause might be a lose cable of some sort, generating lots of signal changes. **Note**: In normal conditions this error should never occur.

ERR_UNEXP_CHRG_ALLOWED_STATE_A

Controller Support	DSCU, SSCU
Description	Unexpected or unauthorized *Mode3* state change activity. The Charge Point will indicate the error with a red LED. Charging will be aborted.
Possible cause / trigger	This situation is normally never allowed to happen and is not part of an expected state transition. The communication between Charge Point and EV went wrong and caused the abort.

ERR_UNEXP_INV_STATE

Controller Support	DSCU, SSCU
Description	Unexpected or unauthorized *Mode3* state change activity. The Charge Point will indicate the error with a red LED. Charging will be aborted.
Possible cause / trigger	This situation is normally never allowed to happen and is not part of an expected state transition. The communication between Charge Point and EV went wrong and caused the abort. A possible cause could be: • Bad or damaged Charge point socket. • Bad or damaged charge cable (plug).

ERR_UNEXP_INV_STATE_FROM_BCD

Controller Support	DSCU, SSCU
Description	Unexpected or unauthorized *Mode3* state change activity. The Charge Point will indicate the error with a red LED. Charging will be aborted.
Possible cause / trigger	This situation is normally never allowed to happen and is not part of an expected state transition. The communication between Charge Point and EV went wrong and caused the abort. A possible cause could be: • Bad or damaged Charge point socket. • Bad or damaged charge cable (plug).

ERR_UNEXP_STATE_F_FROM_BCD

Controller Support	DSCU, SSCU
Description	Unexpected or unauthorized *Mode3* state change activity. The Charge Point will indicate the error with a red LED. Charging will be aborted.
Possible cause / trigger	This situation is normally never allowed to happen and is not part of an expected state transition. The communication between Charge Point and EV went wrong and caused the abort. A possible cause could be: • Bad or damaged Charge point socket.

- Bad or damaged charge cable (plug).

ERR_UNEXPECTED_HW_SETUP

Controller Support	DSCU, SSCU
Description	The *IOCU* processor received an unexpected hardware setup form the MCU processor. The *IOCU* processor will not start executing any charge request.
Possible cause / trigger	Internal error that should never occur. Possible reason might be a faulty update procedure or production failure.

ERR_UNEXPECTED_PLUG_REMOVAL

Controller Support	DSCU, SSCU
Description	The controller detected an unexpected plug removal.
Possible cause / trigger	• Servo (gearbox) lock is defected, the charge plug could simply be removed. • The lock has been forced to open and the charge plug was removed. • Defective charging cable (locking notch is damaged).

ERR_UNKNOWN_REQ

Controller Support	DSCU, SSCU
Description	The IOCU received an unknown or not supported request from the MCU.
Possible cause / trigger	Can only occur if the MCU and IOCU firmware versions are not matching and should not happen during normal operation.

ERR_UNLOCK_DURING_TRANSACT

Controller Support	DSCU, SSCU
Description	A (remote) unlock action has been executed while the charging session was still active. Normally a unlock action can only occur after the charge session is stopped. Charge Point will indicate the error with a red LED.
Possible cause / trigger	This error is mostly triggered by a remote unlock action and will result in the charging session being interrupted (and stopped).

ERR_VOLTAGE_TOO_HIGH

Controller Support	SSCU
Description	The *IOCU* processor standard monitors three on board voltages. In case of a detected undervoltage on one of the voltage measurements this error will be generated The *IOCU* monitors VCCmain (the standard power supply), VCCBuff (the power bank for controlling the servo lock) and VCC3.3IO (main power supply for the *MCU* and *IOCU*.
Possible cause / trigger	• Internal error that should normally not occur. • The main power adapter is not working correctly or is not correctly adjusted to 12V. • Power convertors on the Controller might be faulty.

ERR_VOLTAGE_TOO_LOW

Controller Support	SSCU
Description	The *IOCU* processor standard monitors three on board voltages. In case of a detected undervoltage on one of the voltage measurements this error will be generated The *IOCU* monitors VCCmain (the standard power supply), VCCBuff (the power bank for controlling the servo lock) and VCC3.3IO (main power supply for the *MCU* and *IOCU*.
Possible cause / trigger	• Internal error that should normally not occur. • The main power adapter is not working correctly or is not correctly adjusted to 12V. • Power convertors on the Controller might be faulty.

IOCU_EVENT_POWER_FAILURE

Controller Support	SSCU
Description	VCCmain (the standard power supply) of the controller reached the absolute minimum value of 10V, resulting in a controlled power down of the controller. By default, the contactor (power relay) will be opened (no power supply to the EV possible anymore).
Possible cause / trigger	• A controlled reboot by the *MCU* (resulting in an off/on power cycle of the *IOCU*). • Defected power adapter. • Power adapter experiences a peak overload (power dip), because it is under

dimensioned for its setup (hardware configuration).

