Document History

Rev.	SW	Changes	Author(s)
A	1.2	• Initialization of this document history. • Changed commands for the control of external limits by OCPP in sec. 6.3; moved old commands of software version 1.1 to appendix sec. 7.1. • Added commands for Web-Pull-Limits in sec. 6.5. • Added commands for the control of outlet limits in sec. 6.6.	Krass 2017-12-18
B	1.3	- Added information and explananations for OCPP 1.6 supported configuration keys and feature profiles. - Added list of supported hardware devices (meters). - Updated list of supported ABL products.	Mueller 2018-04-04
C	1.3	- Added information for DataTransfer.	Mueller 2018-04-20
D	1.4	Update for version 1.4	Mueller 2018-08-31
E	1.5	Update for version 1.5	Mueller 2018-12-13
F	1.6	- Added explanations for logging gateway and German “Eichrecht”. - Extended error codes for logging gateway. - Added configuration keys related to logging gateway. - Added DataTransfer for acquisition of meter public keys. - Added "GeneralBreaker" for undefined tripped breaker. - Added restriction to MeterValueSampleInterval. - Added error codes for Eichrecht. - Explanation for the reset behaviour of the charge point. - Revised document for formulations. - Updated model lists.	Mueller, Mull 2019-10-24
G	1.6p3	- Added Huawei LTE USB-Stick. - Added new INIT state for StatusNotification.	Mull 2020-05-28
H	1.6p5	- Removed redundant entry for StopTransactionOnInvalidId. - Added default for LogLevel.	Fassnacht 2020-08-13
I	1.7	- Added Alcatel LTE USB-Stick. - meterStart/meterStop reports -1 - Described new format of vendorErrorCode and info. - Updated/completed error descriptions. - Restructured error code table and added countermeasures.	Fassnacht, Mull 2020-12-23

Table of Contents

1 Scope

The Charge Point software comes with the ABL Single Board Computer (SBC) installed on ABL charging stations (also referred to as charge point). This document describes its feature set, lists the supported products and elaborates on the OCPP interface. It is intended to guide backend (“central system” in OCPP) operators that wish to integrate ABL charging product with their software.

In addition to this document, you may also wish to read the Technical Setup Manual. It describes the station indications, local setup of a station and the web administration interface (*WebAdmin)*.

2 Features

2.1 Installation

The Charge Point software running on the SBC implements a smart charging controller compatible with OCPP. For initial setup, a web administration interface can be used to configure basic parameters required for obtaining connectivity to the backend via OCPP. Once connection to the backend has been established, further configuration is done through OCPP.

2.2 Supported Setups

- Single or twin charger
- Group installations for up to 16 charge points
- Basic load control for twin chargers
- Optional metering
- Local administrative web interface; used for setup and diagnostics
- Local and remote start/stop of charging
- Full support for local cache and white-list
  - A single complete update (i.e. loading a new list) may contain up to 9000 entries.
  - A single update of the list (i.e. a modification of an existing list on the charge point) may contain up to 700 entries.
- Detailed status reporting (error codes, diagnostics)
- Software update over the air (using OCPP)
- Resuming of running transactions after a power failure or other outage (see 3.3)
- Support of Logging Gateway for generating metersignatures. Relevant for fullfilling german Eichrecht

2.3 ABL Products Supported by Charge Point Software

- Pole eMC2: EMC444, EMC444K, EMC445, EMC445K, EMC151, 2P4402, 2P4418, 2P4419, 2P4422, 2P4423, 2P4425, 2P2210, 2P4432, 2P4445, 2P4433, 2P4431, 2P4435, 2P4424, 2P4426, 2P4434, 2P4436
- Pole eMC3: 3P4400, 3P4411, 3P4412, 3P4401
- Wallbox eMH2: 2W22M8, 2W22D3, 2W2240, 2W2241, 2W2261, 2W2260, 2W22BK, 2W72M7, 2W2230, 2W2231, 2W2250, 2W2251

- Wallbox eMH3 Single Master: 3W2208, 3W2211, 3W22K3, 3W22N3, 3W22U3, 3W22E3, 2W22W3, 3W2261, 3W22BH, 3W22HH, 3W22RK, 3W2260, 3W2260B
- Wallbox eMH3 Single Slave: 3W2210, 3W22N9, 3W22W9, 3W2212, 3W2251, 3W2250, 3W22BD
- Wallbox eMH3 Twin Master: 3W2215, 3W22I5, 3W22K5, 3W22N5, 3W22U5, 3W2219, 3W22N8, 3W22NA, 3W22W5, 3W2225, 3W2227, 3W2219, 3W2263, 3W22BI, 3W2264, 3W22BJ, 3W22NI, 3W2263L
- Wallbox eMH3 Twin Slave: 3W2220, 3W2221, 3W2253, 3W2254, 3W22N6, 3W22N7, 3W22NB, 3W22W6, 3W2226, 3W2228, 3W22BF, 3W22U6, 3W22NF, 3W22BG
- External control: 1V0001, 1V0002

2.4 Supported Peripheral Devices Integrated in ABL Charging Stations

- ABL EVCC (V2.7, V2.8)
- ABL EVCC2 (V1.7, V1.8, V2.0, V2.1, V2.5)
- ABL RFIDM20 (V2.3, V2.4)
- ABL RFIDM30 (V1.0)
- Meter: EEM-350-D-MCB (Phoenix)
- Meter: PRO380-Mod, PRO1-Mod (Inepro)
- Meter: EM-340 series, EM-210, EM-111 (Carlo Gavazzi)
- Terminal Equipment: GT864E (CEP AG)
- USB-LTE-Dongle: MS2372h-153, MS2372h-158 (Huawei)
- USB-LTE-Dongle: IK41VE (Alcatel)
- Logging Gateway (Seal)

3 General Behavior

3.1 Start Charging

In general, the electric vehicle (EV) has to be connected to the charging point (CP) before the charging can be started, either locally or remotely. In practice this means: A user has to first connect their EV and then initiate the charging transaction, e.g. by presenting their RFID UID or remotely via a mobile phone app.

Details regarding remote starting of charging transaction is configured using the *ConnectionTimeOut*, *LateOccpuied* and *FreeCharging* OCPP configuration keys.

Some ABL CPs provide two connectors for two EVs to charge at the same time (twin chargers) but only one RFID reader. To ensure that the mapping from EV to RFID UID is correct (and the right customer gets billed the right amount), charging is only initiated for the last connected EV within 10 minutes if the EV is *ready for charging but not yet charging*. In practice, that means the following:

- If no EV is connected, any UID is rejected.
- If one or more EVs are ready to charge, the last connected EV will be authorized by UID and processed further by backend (the CP sends a *StartTransaction.req*) or by local caches. All other EVs will lose their readiness and have to be reconnected to regain readiness.
- If a ready EV is not authorized by UID within 10 minutes it loses its readiness to charge. Any UID will now be rejected. To regain its readiness the EV has to be reconnected.

During charging, the CS connector (connected to the EV) is locked. This is to ensure that the charging cable is not accidentally removed during charging.

The meter value for the start of the transaction is read before the EV is allowed to reach charging state (B2/C2).

If no meter value can be acquired, '-1' will be sent as meterStart value.

3.2 Stop Charging

ABL chargers are stopped by unplugging the cable from the EV. There is no need to present an UID token to stop charging. Unplugging the cable from the EV in turn unlocks the cable from the CP.

The backend can also stop the charging process remotely. After a remote stop, the connector remains locked to prevent unintentional removal of the cable from the CP. With the charging stopped and the cable still locked, two next steps are possible:

- The OCPP backend can send a remote start to resume charging (and start a new transaction)
- The user can unplug the cable from the EV. This will release the cable from the CS.

A socket lock can also be released using OCPP (*UnlockConnector*). In some product setups, the connector can only be released for 30 seconds. If the charging cable is not removed within this time limit, the socket locks again and will need to be unlocked again.

The meter value for the end of the transaction is read as soon as the EV is no longer in state B2 or C2.

If no meter value can be acquired, '-1' will be sent as meterStop value.

3.3 Restart Behavior

A CP installation might go down while one or more EVs are connected and charging, e.g. because of a temporary power outage. Once the CP is available again, charging transaction may be able to resume. To do this, the Charge Point software saves information about active transactions together with a time stamp.

If an EV is plugged in while the power is switched on, the charging station will start the charging process immediately without authorization.

As soon as the SBC has booted, the restart behavior is controlled by the Charge Point software. It can be configured using the proprietary configuration keys *HandleNewTransaction, HandleOldTransaction* and *HandleExpiredTransaction.* Each of those keys can be set to one of the following values: *Cancellation, ReenableUnknown* or *ReenableOld.* An additional configuration key *PowerTimeout* specifies (in seconds) when the transaction information is considered to be expired.

The following items explain each configuration key in detail. They control how the CP behaves after a restart.

- If no information about previously active transactions is available and an EV is charging, a running charging session is assumed to be *new*. If ***HandleNewTransaction*** is set to *Cancellation*, this charging session is stopped. If *HandleNewTransaction* is set to *ReenableUnknown*, a new transaction for an unknown user is started and the charging process remains active.
- If there is information about a previously active transaction, the charging vehicle will be treated as described by the key ***HandleOldTransaction****.* Setting it to *Cancellation* makes the CP terminate the running charging session. Setting it to *ReenableOld* sets up the charging session according to the saved information. *ReenableUnknown* leads to Charge Point terminating the previous charging session and starting a new one for an unknown user.
- By setting the configuration key *PowerTimeout* to a value > *0* seconds, the time stamp of the saved transactions is taken into account. If the time stamp is older than defined by *PowerTimeout*, the saved information is considered *expired* and the charging session is handled according the key ***HandleExpiredTransaction*** instead. This configuration key can be set to *Cancellation*, *ReenableOld* or *ReenableUnknown*, in order to stop, re-enable or start a new transaction for the charging EV; the meaning of the different values is identical to the way *HandleOldTransaction* handles them.

The cache of pending transactions can be cleared remotely using OCPP with a *DataTransfer* request. The DataTransfer should look like this:

- *Vendor-ID: ABL*
- *Message-ID: DeleteTransactionCache*

3.4 Reset Behaviour

A reset of the charge point can be triggered via the following methods. The corresponding behaviour concerning ongoing transactions is described, below:

- **Backend**: If a reset is triggered through the Backend with the Reset-Request, the charge point will try to finish the active transactions before rebooting. This includes sending the StopTransaction-Request to the backend before performing the restart. If the StopTransaction-Requests can not be send, the system will restart after an adapted maximum timeout. This ensures the charge point to perform the reset in any case. After the reset, new charging sessions for connected EV's won't be started. So the configuration of the Restart Behaviour doesn't has an effect. A Soft-Reset will only restart the main software-components of the charge point. This type won't delete log-files, which could be important for troubleshooting. A Hard-Reset will trigger the SBC to perform a complete system-reboot. However, it is only possible to restart this hardware-part. After this type of reboot, the log-files and all volatile memory will be deleted.

- **WebAdmin**: If one triggers the reset via the WebAdmin-Interface (see Technical-Setup-Manual for details), ongoing transactions won't be stopped. After the reset the charge point will continue charging sessions which were active before, no matter which Reset Behaviour is configured. For the type of the reset, the above mentioned conditions apply.

- **Switching power off**: If the reset of the charge point is triggered by switching the power off and on again, the configuration of the Restart Behaviour will apply.

4 Fullfilment of Requirements for “Eichrecht”

The following information only applies to stations which are to be conformant to the German law on units and measurements (“Eichrecht”).

To prevent manipulation of meter values, every pole which will be used to place bills to the customer must be conformant to the German “Eichrecht” (law on measurements and units). For this purpose the charge point manufacturer has to ensure that the meter values can be requested in a way so that they cannot be manipulated.

ABL solves this task by using cryptographic signatures on the meter values. Signatures will only be built on total energy values (measurand: Energy.Import.Active.Register). These signatures are transmitted as an additional information to the backend. This information is compliant to the Open Charge Metering Format (OCMF). Such OCMF data sets can be verified with the "Transparenz-Software" available freely from the “SAFE” association. For more informations about this, please visit https://transparenz.software/.

To embed OCMF in OCPP, each MeterValue consists of the SampledValues for the original measurands (OCPP ValueFormat "Raw") and an additional SampledValue carrying signature in the form of a string of hexadecimal encoded data (OCPP ValueFormat "SignedData"). The encoded data itself consists of an OCMF data set which carries transaction related information and a signature over this information. The used format is OCMF in version 1.0. Currently, only the textual representation of this Format is generated.

Note: OCMF is a record based modular format. Thus, either multiple data sets for the beginning and the end of a transaction or a single charge data record (CDR) can be generated. By embedding both the OCMF data sets for begin and end into a single XML based input format for the "Transparenz-Software" the whole data of a transaction can be processed with the "Transparenz-Software" at once which then also performs difference-

calculation. Analogously, a public key can be supplied, as well. These tasks have to be performed by a component in the backend.

At the beginning of a transaction the charge point will send a MeterValue request immediately after the StartTransaction request.

The following example depicts how such a data set looks like for OCPP 1.6-J in a MeterValues request:

```
{
        "connectorId":1,
        "transactionId":2007,
        "meterValue":
        [
                {
                        "timestamp":"2019-07-04T08:30:08Z",
                        "sampledValue":
                        [
                                {
                                        "value":"268.978",
                                        "context":"Transaction.Begin",
                                        "format":"Raw",
                                        "measurand":"Energy.Active.Import.Register",
                                        "location":"Outlet",
                                        "unit":"kWh"
                                },
                                {
                                        "value":"HEXADECIMAL_STRING_WITH_ENCODED_DATA",
                                        "context":"Transaction.Begin",
                                        "format":"SignedData"
                                }
                        ]
                }
        ]
}
```

The hexadecimal string will be added to every MeterValues-Request. For the MeterValues in the StopTransaction request, one can choose between two signature formats, defined by the configuration key "EndSignatureType". The format type "SIGNATURE_TRANSACTION" will only send the data set for the last meterValue. The format type "SIGNATURE_CDR" will send a charge data record containing the begin and end MeterValues (see also 5.3).

This functionality is only available on ABL "Plus" products. For those products the configuration key "TransmitMeterSignature" is "true" by default, which enables the acquisition of meter values. If you set it to false manually, you will not be able to bill the customers for the consumed energy. Also, the default behaviour of the Transaction-Manager is set, such that charging sessions won't be continued after a power fail or won't be started for an unknown user (HandleNewTransaction = HandleOldTransaction = HandleExpiredTransaction =

Cancellation). If you change this configuration, you have to check compliance with the Eichrecht for your use case.

If the communication with one of the signature relevant hardware devices (i.e. the meters or the logging gateway) breaks, the connector will be set to unavailable. Ongoing transactions will be stopped, immediately. This ensures, that no energy can be drawn without validly signing the corresponding meter value. A related error message will be generated.

The logging gateway cares for saving signatures and relevant events into a database. The usage of the database is checked regularly, every 24 hours. If the usage reaches a level of 95%, new charging sessions will be prevented. The outlets will be set to unavailable with an appropriate error message (see 5.4).

5 OCPP Interface

Supported OCPP versions:

- OCPP 1.5
- OCPP 1.6

Supported Transport Layers:

- SOAP/HTTP (no TLS supported)
- Websocket/JSON; with and without TLS encryption

The service endpoint address at the charge point follows the following conventions:

- Address *http://<IP-Address>:<Port>/ChargePoint*
- Example *http://1.2.3.4:7890/ChargePoint*
- The IP address of the charge point is acquired by DHCP or provisioned by the mobile network.
- The port may be adjusted using the web administration interface.

5.1 Supported OCPP 1.6 Profiles

The following table shows the supported OCPP 1.6 Feature Profiles. For details about functions and configuration keys, see the corresponding tables in the following sections.

Profile:	Support:
Core	yes
Firmware Management	yes
Local Auth List Management	yes
Reservation	no
Smart Charging	yes
Remote Trigger	yes

5.2 Functions

The following table shows the support of the various OCPP functions.

Direction: Charge Point → Central System		
Function:	Support:	Remarks:

Authorize	yes	
BootNotification	yes	
DataTransfer	yes	for details see 5.7
DiagnosticsStatusNotification	yes	
FirmwareStatusNotification	yes	
Heartbeat	yes	
MeterValues	yes	
StartTransaction	yes	Meter Value '-1' if meter could not be read
StatusNotification	yes	
StopTransaction	yes	Meter Value '-1' if meter could not be read
Direction Central System → Charge Point:		
Function:	Support:	Remarks:
CancelReservation	no	
ChangeAvailability	yes	
ChangeConfiguration	yes	
ClearCache	yes	
ClearChargingProfile	yes	
DataTransfer	yes	See 5.7
GetCompositeSchedule	yes	
GetConfiguration	yes	
GetDiagnostics	yes	Upload protocols: FTP
GetLocalListVersion	yes	
RemoteStartTransaction	yes	
RemoteStopTransaction	yes	
ReserveNow	no	
Reset	yes	
SendLocalList	yes	
SetChargingProfile	yes	ChargingProfilePurpose *ChargePointMaxProfile* not supported
TriggerMessage	yes	
UnlockConnector	yes	
UpdateFirmware	yes	Download protocols: HTTP, FTP

5.3 Configurations

The following table lists the supported configuration keys.

Standard keys according to OCPP 1.5:			
Key:	Support:	Default value:	Remarks:
HeartBeatInterval	yes	-	

ConnectionTimeOut	yes	0	Defines maximum time from RemoteStart until connecting a car. For local authorization the car has to be connected before placing a tag in front of the reader.
ResetRetries	no		Reset always performs identical procedure. No need for retries.
BlinkRepeat	no		not applicable
LightIntensity	no		not applicable
MeterValueSampleInterval	yes	0	To ensure performance, this value can be set to a minimum of 3. For Eichrecht-compliant installation this value can only be set to a minimum of 46.
ClockAlignedDataInterval	yes	0	
MeterValuesSampledData	yes	Energy.Active.Import.Register	
MeterValuesAlignedData	yes		
StopTxnSampledData	yes	Energy.Active.Import.Register	
StopTxnAlignedData	yes		
Additional standard keys according to OCPP 1.6:			
Key:	Support:	Default value:	Remarks:
StopTransactionOnInvalidId	yes	false	Whether the charge point will end an ongoing transaction when it receives a non-Accepted status for the *UserId*
NumberOfConnectors	yes		The number of connectors
LocalPreAuthorize	yes	true	Use local cache and white-list, if online.
LocalAuthorizeOffline	yes	true	Use local cache and white-list, if offline.
AllowOfflineTxForUnknownId	yes	false	If the charge point is offline, accepted Ids and Ids which are not yet in the local cache or white list are accepted
AuthorizationCacheEnabled	yes	True	The local cache is enabled
AuthorizeRemoteTxRequests	no	false	Transactions initiated by RemoteStart are not authorized locally
ConnectorPhaseRotation	no		
ConnectorPhaseRotationMaxLength	no		
GetConfigurationMaxKeys	no		No restriction applies
MaxEnergyOnInvalidId	no		
MeterValuesAlignedDataMaxLength	no		No restriction applies
MeterValuesSampledDataMaxLength	no		No restriction applies
MinimumStatusDuration	yes	0	Status changes are transmitted

			immediately
NumberOfConnectors	yes		
StopTransactionOnEVSideDisconnect	no	true	Transactions are always stopped if the cable is unplugged by the customer
StopTxnAlignedDataMaxLength	no		No restriction applies
StopTxnSampledDataMaxLength	no		No restriction applies
SupportedFeatureProfiles	no		Supported profiles: Core Profile, Firmware Management, Local List Management
SupportedFeatureProfilesMaxLength	no		No restriction applies
TransactionMessageAttempts	no		As long as no reboot happens, all messages are saved by the charge point, such that all messages are re-sent if the connection is up again. For behavior after reboot, see 3.3.
TransactionMessageRetryInterval	no		see above
UnlockConnectorOnEVSideDisconnect	no	true	As soon as the EV is unplugged, no further locking is possible
WebSocketPingInterval	yes	30	
LocalAuthListEnabled	no		Can be set via LocalPreAuthorize, LocalPreAuthorizeOffline and AuthorizationCacheEnabled
LocalAuthListMaxLength	no		At least 10,000 entries can be stored
SendLocalListMaxLength	no		Up to 9,000 entries can be processed within a single SendLocalList.req
AuthorizationKey	yes		There is the String. It uses by TLS/SSL conntection for authentication (using HTTP Basic Authentication)
Proprietary extensions by ABL:			
Key:	Support:	Default value:	Remarks:
AccessPointName	yes		Allows setting the APN.
AccessPointUser	yes		User name for authentication at the Access Point.
AccessPointPassword	yes		Password for authentication at the Access Point.
ChargeBoxId	yes	ABL_<SBC3-serial-number>	Allows changing the charge box ID.
ServiceURL	yes		Allows changing the central system endpoint address.
OcppVersion	Yes	1.5	The active version of OCPP. Supported versions: 1.5 and 1.6

Comments	yes		Allows placing comments visible to OCPP and the local administration web interface.
LogLevel	yes	INFO	for internal use only
LogOcppMessages	yes	true	for internal use only
LogOcppInvocations	yes	true	for internal use only
LateOccupied	yes	false	If set to true: Connector will be reported as occupied only as of start of charge. Before, available will be reported, even if a car is connected to the station. If set to false: Standard behavior: Occupied as soon as car is plugged in.
FreeCharging	yes	false	Allow charging without authorization. This will start charging sessions immediately when a car is connected.
FreeChargingOffline	Yes	false	If the charging station is off-line, no authentication is required and charging starts immediately.
FreeChargingUid	yes	“00000000000000”	The Uid used for an unknown user. This value should be a 4 or 7 byte hex-String.
PowerTimeout	yes	0	The time in seconds until a previous charging session expires (see 3.3).
HandleNewTransaction	yes	ReenableUnknown*	Defines the behavior after a power failure and an unknown charging EV (see 3.3).
HandleOldTransaction	yes	ReenableOld	Defines the behavior after a power failure and a charging EV with information about a previous active charging session (see 3.3).
HandleExpiredTransaction	yes	ReenableUnknown	Defines the behavior after a power failure (after the PowerTimeout is expired) and a charging EV with information about a previous active charging session (see 3.3).
ShortenUIDs	yes	false	Configure the UID format. Setting this key to true configures the CP to send four byte ISO 14443 UID as-is, with no zero-padding to seven bytes.
SecurityProtocols	yes		There is the comma separated list of the TLS/SSL versions. It prevents the downgrade of the connection protocol.
DisableDowngrading	yes	true	Prevent the downgrading of the SBC operating system.
TransmitMeterSignature	yes	false/true	Products which are conformant with the Eichrecht will transmit a signature for the MeterValue. Products which are not conformant won't transmit a signature. IMPORTANT: 1)If you disable signatures in a product which is Eichrecht-

			compliant, you do it on your on risk! 2) If you set the value to true in non Eichrecht-conformant products, the outlets will be held unavailable (since signatures can not be acquired).
EndSignatureType	yes	SIGNATURE_TRANSACTION	Defines the type of end-signature for a meterValue. If set to "SIGNATURE_TRANSACTION" contains only the last meterValue. If set to "SIGNATURE_CDR" contains the start and end meterValue.

5.4 Error Codes

The following table lists supported OCPP 1.5/1.6 error codes as they appear in *StatusNotification.req* when sent by the SBC charge point software.

The OCPP field vendorErrorCode consists of a device-type followed by a space and the actual error code:

vendorErrorCode: <device-type> <error-code>

The OCPP field info consists of a logical ID (that is an internal address assigned to the device) and an explanatory text to help identify the error:

info: <logical-id>: <text>

OCPP field errorCode and description according to OCPP:	OCPP field vendorErrorCode:	OCPP field info:	Possible reasons and proposed resolutions:
ConnectorLockFailure *Failure to lock or unlock connector.*	EVSE F5	Socket of outlet could not be locked.	Mechanical error, plugged incorrectly or locking actor defect. 1. Re-plug EV 2. OCPP UnlockConnector 3. Reboot
GroundFailure *Ground fault circuit interrupter has been activated.*	EVSE F3	DC residual current detected or self test of RCD-MD failed.	1. Re-plug EV 2. EV may have a fault.
	RCCB TRIPPED	Residual current circuit breaker tripped.	At site: Check installation and reset RCCB.
HighTemperature *Temperature inside charge point is too high.*	EVSE F10	Internal temperature of outlet over 80°C or NTC failure.	1. Charging process will be stopped. Will be resolved when temperature drops below 60°C for at least 10 minutes. Check at site why over-temperature happens. 2. For extreme temperatures will stay until power-cycle.
	EVSE F17	Internal temperature of outlet increased (60°C-80°C).	Charging will continue at reduced rate of 6A.
InternalError *Error in internal hard- or software*		currently not in use	

component.			
LocalListConflict *The authorization information received from the Central System is in conflict with the LocalAuthorization List.*		currently not in use	
Mode3Error/ EVComm-unicationError *Problem with Mode 3 connection to vehicle.*	EVSE F6	Outlet with socket: No valid coding-resistor at CS detected; rated current of cable cannot be read.	1. Re-plug EV. 2. EV has fault. 3. Cable is faulty.
	EVSE F7	Vehicle requests charging with external ventilation; state D not supported.	The obsolete ISO state D is not supported by ABL equipment.
	EVSE F8	Voltage of CP out of limit.	1. Re-plug EV. 2. EV has fault. 3. Cable is faulty.
NoError *No error to report.*			
	EVSE INIT	Initializing outlet.	No action to be taken. This is a normal initialization state during startup of the station. May last up to 30s.
OtherError *Other type of error. More information in vendorErrorCode.*	<type> TIMEOUT	Field-bus communication timeout	One occurence in six months or for short duration (less than 1 minute) may be ignored. Frequent occurrences usually are symptoms of non-correct/faulty bus wiring. Check bus wiring of control and/or meter bus.
	<type> FAULTED	Field-bus protocol error	
	<type> INCOMPATIBLE	Incompatible firmware version of field-bus device	The firmware of this field-bus device is not supported by the SBC software. Check firmware version and correct it by upgrading.
	<type> MISCONFIGURED	Invalid configuration of field-bus device	The settings or parameter set of this field-bus device are not matching the expextations of the SBC software. Fix by using the correct firmware or update the SBC software to a more recent version.
	<type> NOT_PRESENT	Device could not be discovered on field-bus.	This field-bus device could not be detected on the bus. 1. Check product configuration for right model/revision of the enclosing product. 2. Check bus cabling. If this occurs for all devices in a product the

			busses may have been swapped by accident.
	EVSE F2	Invalid condition detected by EVCC self test.	1. Reboot. 2. Check bus wiring. 3. Check EVCC for hardware failures.
	EVSE F4 / EVSE F14	Field-bus communication timeout of SBC detected by outlet.	Check bus wiring. F14 only: Charging will continue at reduced rate of 6A.
	SPD TRIPPED	Surge protection device triggered.	Lighning strike may have occurred. Check installation at site. Replace surge protection.
	METER NOT_SMART	Meter is not suitable for signatures.	1. Check that LGW is built into product and wired between meter and SBC. 2. Check configuration of LGW for correctness.
	SMGW UPDATE_RUNNING	Device is performing an update.	The LGW is updated. This state vanishes after the update is finished. Do NOT reboot.
	GeneralBreaker TRIPPED	Circuit breaker or RCCB tripped.	See descriptions for RCCB and circuit breaker.
	SMGW DB_FULL	LGW database is full.	The LGW storage capacity for transactions is exhausted. Archive the currently built-it LGW and replace it by a new one.
OverCurrentFailure *Over current protection device has tripped.*	EVSE F9	Charging current of at least one phase exceeds programmed limit.	1. Re-plug EV. 2. EV has fault.
	EVSE F15	Load imbalance detected.	Charging will continue at reduced rate of 20A.
	EVSE F16	Communication with overcurrent detection failed.	Charging will continue at reduced rate of 10A. Check EVCC if this error recurs.
	MCB_32 TRIPPED	Circuit breaker tripped.	1. Check reason of trigger. 2. Reset breaker.
OverVoltage *Voltage has risen above an acceptable level.*		currently not in use	
PowerMeterFailure *Failure to read power meter.*	METER NO_SIGNATURE	No device for meter-signature	Check product configuration for right model/revision of the enclosing product.
	METER TIMEOUT	Field-bus communication timeout	One occurence in six months or for short duration (less than 1 minute) may be ignored. Frequent occurrences usually are symptoms of non-correct/faulty bus wiring.
	METER FAULTED	Field-bus protocol error	

			1. Check bus wiring of meter bus. 2. Replace meter.
	METER INCOMPATIBLE	Incompatible firmware version of field-bus device	The firmware of this meter device is not supported by the SBC software. 1. Check and correct application profile of meter. 2. Upgrade SBC software.
	METER MISCONFIGURED	Invalid configuration of field-bus device	The settings of this meter are not matching the expextations of the SBC software. Update the SBC software to a more recent version.
	METER NOT_PRESENT	Device could not be discovered on field-bus.	This field-bus device could not be detected on the bus. 1. Check product configuration for right model/revision of the enclosing product. 2. Check bus cabling. If this occurs for all devices in a product the busses may have been swapped by accident.
PowerSwitchFailure *Failure to control power switch.*	EVSE F1	Unintended closed main contactor (welding)	1. Check main contactor. 2. Check auxiliary switch of main contactor.
	EVSE F11	Main contactor does not close.	
ReaderFailure *Failure with ID tag reader.*	AUTHENTICATOR TIMEOUT	Field-bus communication timeout	One occurence in six months or for short duration (less than 1 minute) may be ignored. Frequent occurrences usually are symptoms of non-correct/faulty bus wiring. Check bus wiring of control bus.
	AUTHENTICATOR FAULTED	Field-bus protocol error	
	AUTHENTICATOR INCOMPATIBLE	Incompatible firmware version of field-bus device	The firmware of this RFID reader is not supported by the SBC software. Check firmware version and correct it by upgrading.
	AUTHENTICATOR MISCONFIGURED	Invalid configuration of field-bus device	The settings or parameter set of this RFID reader are not matching the expextations of the SBC software. Fix by using the correct firmware or update the SBC software to a more recent version.
	AUTHENTICATOR NOT_PRESENT	Device could not be discovered on field-bus.	This RFID reader could not be detected on the bus. 1. Check product configuration for right model/revision of the enclosing product. 2. Check bus cabling. If this occurs for all devices in a product the busses may have been swapped by accident.
ResetFailure		currently not in use	

Unable to perform a reset.			
UnderVoltage *Voltage has dropped below an acceptable level.*		currently not in use	
WeakSignal *Wireless communication device reports a weak signal.*		currently not in use	

5.5 WebSocket Secure Connectivity

The WebSocket protocol (wss://) supports transport layer security (TLS) to protect the connection from eavesdropping. In addition, if client and server certificates are used, the authenticity of the peers can be verified, protecting against man-in-the-middle attacks.

The client can send Server Name Indication (SNI). SNI is an extension for the TLS by which a client specifies the hostname which it uses at the handshaking process. The name must be Fully Qualified Names (FQDN), that is: The name must contain a dot '.' and it should be not a plain IPv4 or IPv6 address.

5.6 Upload Diagnostics

The backend can request the charging station to create and upload a diagnosis file to a given URL. The URL shows the protocol and the place for the upload file. The charge point supports the following protocols: FTP, HTTP, HTTPS.

The File Transfer Protocol (FTP) is the standard protocol for the transfer of the diagnostics file from a charge station to the backend. FTP has the following form for an URL:

ftp://[user[:password]@]host[:port]/url-path

Where:

- ftp is the protocol name;
- user, password are an optional info about user;
- host is a registered server name;
- port is an optional subcomponent. The standard port FTP is 21;
- url-path is a remote directory where the diagnostics file will be saved.

Example URLs:

ftp://userftp:secret@server-ftp.com:2112/upload-directory or ftp://server-ftp.com/upload-directory

The HTTPS and HTTP protocols use the multipart format with the “POST” method.

An example of the transferred data:

```
--16a9b8647642
Content-Disposition: form-data; name="binaryFile"; filename="uploaded-file.tar.gz"
Content-Type: application/octet-stream
Content-Transfer-Encoding: binary
```

<Binary content of the diagnostic file>

--16a9b8647642--

5.7 Proprietary use of OCPP DataTransfer

Since version 1.2, proprietary requests via *DataTransfer* are supported. The central system initiates a *DataTransfer,* the CP reacts as defined below.

VendorId	MessageId	Data	Response	Remarks
ABL	*GetLimit*	*logicalId=<logical Id of limit-device>* e.g. "logicalId=limit200"	current value in A or *REJECTED*	Requests the current value (in Ampere) of the limit-device. For details see 6.3.
ABL	*SetLimit*	*logicalId=<logical Id of limit-device>; value=<LimitValue>* e.g. "logicalId=limit2;value=60"	*ACCEPTED* or *REJECTED*	Sets the value (in Ampere) of the limit-device. For details see 6.3.
ABL	*GetOutletLimit*	*logicalId=<logical Id of evse-device>* e.g. "logicalId=evse101"	current value in A or *REJECTED*	Requests the actuall value (inAmpere) of the limit-device. For details see 6.6.
ABL	*SetOutletLimit*	*logicalId=<logical Id of evse-device>; value=<LimitValue>* e.g. "logicalId=evse101"	*ACCEPTED* or *REJECTED*	Sets the value of the limit-device. For details see 6.6.
ABL	*DeleteTransactionCache*	-	*ACCEPTED* or *REJECTED*	Deletes old saved transactions. Perform a reboot afterwards. For details see 3.3.
ABL	*GetMeterPublicKey*	connectorId [int] e.g. "1"	status: ACCEPTED or REJECTED data: <PublicKey>	The PublicKey is transmitted in Hex-Format

6 External Setting of Current Limits

Charge Point supports setting current limits on individual charge points or master/slave installations. Limits can be configured with (1) OCPP, an (2) HTTP API provided by the Charge Point software or using (3) Web-Pull, in which the Charge Point software reads limit from an (external) URL.

Limits are implemented as virtual devices inserted into the device tree (WebAdmin tab “Devices”). They shall be referred to as *virtual dynamic limits*.

In addition, it is possible to set current limits of single outlets directly with the HTTP API or through OCPP. Since an outlet is a real device with different characteristics than virtual devices, its dynamic current limit will be denoted as *outlet's dynamic limit*.

Please note: In order to be able to dynamically change current limits, an ABL-certified installer has to define and configure these limits properly in advance or these settings have to be done in the factory.

6.1 OCPP SmartCharging (since version 1.4)

Charging profiles are configured with the *SetChargingProfile* OCPP command. OCPP defines three different possible profiles.

ChargingProfileKind	
TxDefaultProfile	Can be set for all connectors (connector=0) or individual connectors
TxProfile	Can be set for any individual conenctor
ChargePointMaxProfile	Not supported

The command *GetCompositeSchedule* reports the *ChargingSchedule* for a specified duration, beginning at the current system time.

To remove a charging profile, the *ClearChargingProfile* request. If a charging session is active and a *ClearChargingProfile r*equest reaches the charge point, the current valid charging profile will be recalculated.

6.2 Standard Properties of all Virtual Dynamic Limits

The basic properties of all limits (API limit, OCPP limit or Web-Pull limit) are shown on the “Devices” page of WebAdmin. The following items explain their properties.

- *Minimum* and *Maximum* value (in Amperes, e.g. 32): External settings can not set the limit below *Minimum* or above *Maximum*.
- *Start value* (in Amperes, e.g. 7): Initial value of the limit after a reboot.
- *Time limit* (in seconds, e.g. 100; 0 for infinity): Amount of time in seconds after which the system returns to a fallback value (see below), if the external controller does not refresh the setting..

  This is to ensure that the external controller is working properly. If the external controller crashes, the charge point falls back to a safety mode which only allows very low currents.
- *Fallback value* (in Amperes, e.g. 7): The limit to fall back to if no updates have been received within in the *Time Limit*.

The configuration of all limits is shown on the “Devices” page in WebAdmin. The “Diagnosis” page lists the current limit configuration and time since the last update.

6.3 OCPP Limits Since Version 1.2 of Charge Point Software

To update OCPP limits, the OCPP function *DataTransfer* is used.

Please note: In order to utilize this function, first a limit of type *OCCP* has to be defined on the “Devices” page of WebAdmin by an ABL-certified installer. For details, consult the Technical Setup Manual.

This section describes the interface for Charge Point software version 1.2. Version 1.1 provides an earlier *and now deprecated* interface.

6.3.1 Getting the Current Limit

The central system sends the following *DataTransfer* to check the current OCPP limit:

- Vendor-ID: ABL
- Message-ID: GetLimit

- Data: logical-Id of the OCPP-limit device in the syntax *logicalid=value*
  - the logical-Id of a device can be found on the diagnosis page of WebAdmin
  - example: for the device with logical-id *limit100* enter into the data field: *logicalid=limit100*

Charge Point replies with the current limit in Amperes or one of the following error messages:

- *Rejected / Data: Access denied*
  device identified by logical-Id is not an OCPP-limit
- *Rejected / Data: Answer = WRONG_TYPE_OF_DEVICE*
  device identified by logical-Id is not a limit at all

6.3.2 Setting the Current Limit

The central system sends the following *DataTransfer* to check the current OCPP limit:

- Vendor-ID: ABL
- Message-ID: SetLimit
- Data:
  - logical-Id of the OCPP limit device in the syntax *logicalid=value*
    the logical-Id of a device can be found on the diagnosis page of WebAdmin
  - value in Amperes (without unit, e.g. *20.5)*, in the syntax *value=value*
    this value has to be between the minimum and the maximum allowed value (including minimum and maximum value)
  - example for a device with logical-ID *limit100* and the value set to *25A*:
    logicalid=limit100;value=25

Charge Point will return one of the following codes to the OCPP backend:

- *ACCEPTED*
- *VALUE_OUT_OF_RANGE* (if the value is out of the allowed range)
- *CONVERSION_ERROR* (if the value to be set cannot be converted to a float)

6.4 API Limits

WebAdmin provides an HTTP API which can be utilized to control current limits by (automatic) web requests. An external system can set the current limit by calling a special URL on the charge point SBC.

In order to check whether an API-controlled limit is available in the system configuration of the product, visit the “Devices” page of WebAdmin. Here, the properties of the limit as well as the individual URLs to control it are displayed. Only ABL-certified installers can set up a API-controlled limit (see Technical Setup Manual).

6.4.1 Using the API

This section explains how to utile the HTTP API.

The following examples assume that the SBC is reachable at IP address *172.16.30.31* and that the configured limit has the logical ID *limit200*. To the the actual id of your limit, consult the “Diagnosis” page in WebAdmin.

In order to get the present setting of the limit, the request URL would be:

http://172.16.30.31:8300/api.html?logicalID=limit200&cmd=GetLimit

The API will return the decimal value (and nothing else) in the response body.

In order to set the limit to a value of e.g. 20 Amperes, the request URL would be:

http://172.16.30.31:8300/api.html?logicalID=limit200&cmd=SetLimit&value=20

The last number in the URL denotes the value to be set. In order to set 15 Amperes, the URL therefore would be:

http://172.16.30.31:8300/api.html?logicalID=limit200&cmd=SetLimit&value=15

The API will return one of the following codes to the requesting external system:

- *ACCEPTED*
- *VALUE_OUT_OF_RANGE* (if the value is out of the allowed range)
- *CONVERSION_ERROR* (if the value to be set cannot be converted to a float)

6.5 Limit Control by Web-Pull

Since version 1.2 of Charge Point, it is possible to utilize so called *Web-Pull limits*. Charge Point calls an external HTTP URL regularly to get values for the limit.

Difference to API limits: When using API limits, an external controller initiates communication with Charge Point to configure the current limits. With Web-Pull limits, Charge Point itself initiates communication with the external controller.

In order to set up a Web-Pull limit, a certified ABL installer has to change the type of the limit on the “Devices” page of WebAdmin to Web-Pull. Configuration requires (1) the URL to fetch (e.g. *http://example.com/limits.txt*) and (2) the refresh rate in seconds and should be handled like this:

1. Change limit type to Web-Pull
2. *Soft reset* the software
3. Change the parameters of the limit, especially URL and fetch frequency
4. *Soft Reset* the software

If Charge Point can not fetch the URL or does not get a valid value, it will set the limits value to the fallback value after the time limit.

6.6 Control of Outlet's Dynamic Limit by API and OCPP

Limits are virtual devices associated with installed products. The main supply can be configured through WebAdmin.

Since Charge Point version 1.2, it is additionally possible to change the dynamic current limit of an outlet itself by OCCP call or by API. This limit can be between *0* and *the factory setting of the outlet*.

6.6.1 OCPP Interface

Configuration of this value is identical to the way limits were configured in the previous sections. The commands are *GetOutletLimit* and *SetOutletLimit* instead of *GetLimit* and *SetLimit*.

Assuming the logical Id of the outlet is *evse1000*, the following *DataTransfer* messages configure the limit:

- Get the dynamic limit value of the outlet
  - Message-ID: GetOutletLimit
  - Data: logical-Id of the outlet device in the syntax *logicalid=value*
    Example: *logicalid=evse10*
- Set the dynamic limit value of the outlet
  - Message-ID: SetOutletLimit
  - Data: logical-Id of the outlet device in the syntax *logicalid=value* and the value to be set
    Example with 20A and outlet evse100: *logicalid=evse100;value=20*
  - Reset the dynamic limit value setting by sending -1 as value in OCPP's data field:
    logicalid=evse100;value=-1

6.6.2 HTTP API Interface

Assuming the SBC is reachable at IP address *172.16.30.31* and the logical Id of the outlet is *evse100*, these are the URLs for configuring the outlet's dynamic limit.

- Get the dynamic limit value
  http://172.16.30.31:8300/api.html?cmd=GetOutletLimit&logicalID=evse100
- Set the dynamic limit value
  http://172.16.30.31:8300/api.html?cmd=SetOutletLimit&logicalID=evse100&value=15
- Reset the dynamic limit by setting the value to *-1*
  http://172.16.30.31:8300/api.html?cmd=SetOutletLimit&logicalID=evse100&value=-1

6.6.3 Notes

- This configuration is only indirectly visible in the line "*Max. current limit:*" of the outlet (EVSE) in the "Diagnosis" page of WebAdmin.
- The setting of this value will be reset by a soft- or hard reset of the Charge Point software.

6.7 Notes on Setting Limits

- Limit values should not change every few seconds. Doing so leads to some EVs reporting an error or refusing to charge.
- Reducing the maximum allowed current is immediately propagated to all EVs currently connected for charging. According to OCPP, EVs have five seconds to adapt their current consumption accordingly.
- Increasing the maximum allowed current will not be propagated to all EVs right away, rather one limit gets updates every six seconds.

  Example: A setup with six charging EVs will take about 30 seconds to propagate the new limits to all EVs.

- ChragePoint informs the EV of the maximum allowed current consumption. The real current consumption of the EV must not exceed this value, *but it can be lower*.
- If the current consumption of an EV is higher than the allowed current limit for more than five seconds, the EV will be disconnected from the charging point by switching the fuses to off.

7 Appendix

7.1 Limit Control with OCPP in Version 1.1 of Charge Point Software

In OCCP, the OCPP function *DataTransfer* is utilized to control the limit.

In order to check present current limit, the OCPP backend has to send the following command:

- Vendor-ID: ABL
- Message-ID: GetOCPPControlledLimitValue

The charge point will return the present current limit in Amperes.

In order to set the current limit, the OCPP backend has to send the following command:

- Vendor-ID: ABL
- Message-ID: SetOCPPControlledLimitValue
- Data: Decimal value in Amperes (without unit), e.g. 20.5
  This value has to be between 0 and the maximum allowed value including 0 and the maximum value.

The Charge Point software will return one of the following codes to the OCPP-backend:

- ACCEPTED
- VALUE_OUT_OF_RANGE (if the value is out of the allowed range)
- CONVERSION_ERROR (if the value to be set cannot be converted to a float)

8 References

- OCPP 1.5 12-06-08: Open Charge Point Protocol – Interface description between Charge Point and Central System
- OCPP 1.6 Edition 2, 2017-09-28: Open Charge Point Protocol – Interface description between Charge Point and Central System
- IEC 61851-1 Ed 2.0:2010: Electrical vehicle conductive charging system – Part1: General requirements
- DIN EN 61851-1:2012-01: Konduktive Ladesysteme für Elektrofahrzeuge; Teil 1: Allgemeine Anforderungen
- IEC 61851-1 Ed 3 69/219/CD: Electrical vehicle conductive charging system – Part1: General requirements