New Contact Address

in-tech smart charging GmbH

Important notice

Dear customer,

in-tech smart charging GmbH is expanding. From now on we will be based in Bitterfelder Str. 1-5 in 04129 Leipzig.

Our email address smartcharging@in-tech.com and phone number +49 341 39290250 remain unchangend.

As of now mail will be received at the new address.

Yours sincerely,

in-tech smart charging GmbH

EVAcharge SE Datasheet

in-tech smart charging GmbH

April 20, 2020

Revisions

Revision	Release Date	Changes
8	April 20, 2020	added max height above mean sea level
7	December 06, 2016	Fixed LGE field in USS request control PWM, add note that USS service 0x11 requires an enabled PWM
6	December 11, 2015	updated contact section
5	October 27, 2015	removed specific RAM part number
4	February 10, 2015	lowered V_PP maximum voltage, refined section ”Applications”, added information for threads of the RS232/CAN connector X5, added Zero Cross Detection, ”USS response to device test 2” corrected service field to 0x84, updated weight, added KL02 Reset to kl02-imx connection, moved Zero cross Detection Overvoltage Category to Absolute maximum ratings and added Zero Cross Degree of Pollution, created chapter mounting holes
3	March 25, 2014	updated boot selection jumpers for different revisions, added chapter ”How to identify the PCB revision number”, updated input voltage requirements
2	December 18, 2013	updated USS documentation to include service 0x12, clarified programming section, added note to Debug UART about voltage levels, added information about connector types of X4 and X6, added information about internal connections
1	October 10, 2013	initial release

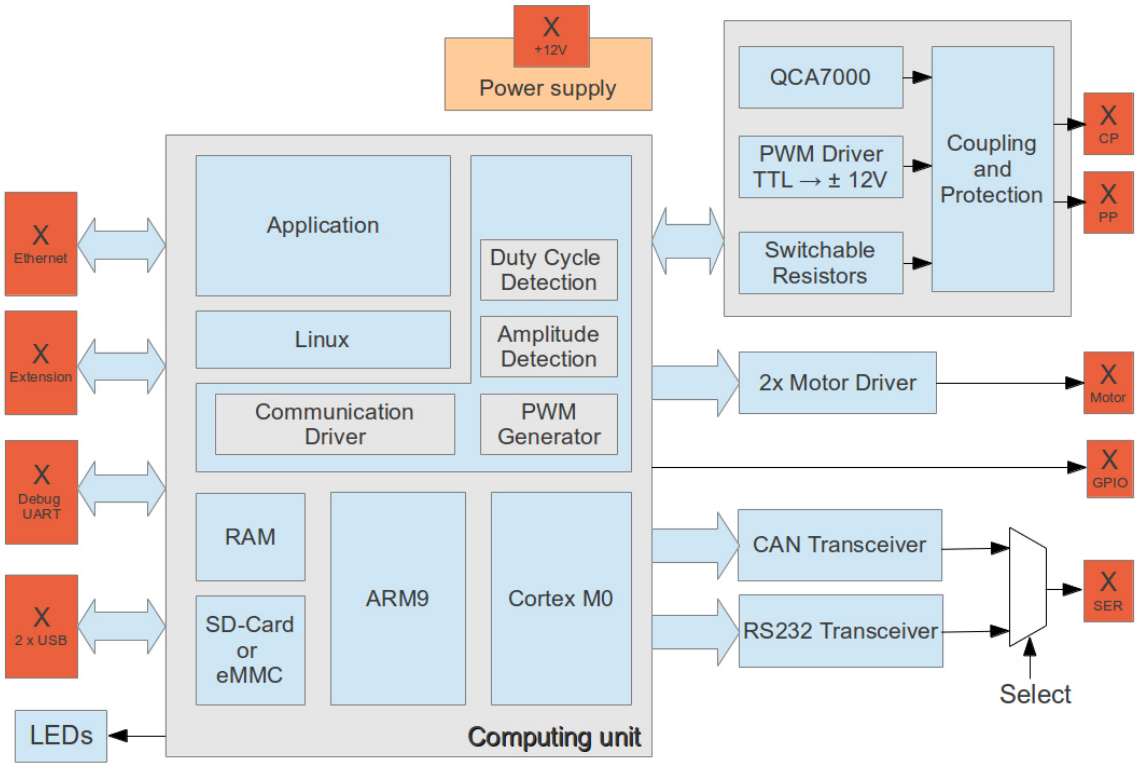
Contents

1 Introduction

EVACharge SE is a ISO 15118 compliant controller for electric vehicle charging stations. The board contains the PLC communication via CP with PWM generation and HomePlug Green PHY integration. The board will be provided with a Linux operating system.
The board can act as EVSE as well as PEV.

- Based on the Freescale i.MX287
- Storage: eMMC 4 GB
- Network interface: Fast Ethernet
- Operating system: Debian jessie, Kernel 3.10 (or newer)
- RAM: 128 Mbyte

Parameter	Value
Power supply	12 V
Power consumption	Max. 4 W (2.6 W in idle mode) - Plus Power for USB devices
Temperature range	-40 °C - +85 °C
air humidity	95% rel. humidity (non condensing)
Outline dimension	100 mm x 120 mm x 20 mm
Weight	92 g
RoHS	EVACharge SE is manufactured RoHS compliant.



2 Applications

EVACharge SE is a communication platform for Electric Vehicle Supply Equipment (EVSE) as well as plug-in electric vehicles (PEV). It enables the charge controller to communicate with electric vehicles (EVs) that are ISO 15118 / DIN 70121 compliant. For communication between EVSE and PEV it supports CP (control pilot) and PP (proximity pilot) signaling including Green PHY communication. The PP signal can also be used to simulate cables with different charge current capability.

Possible Applications:

- charge controller in electric vehicle supply equipment (EVSE)
- charge controller in plug-in electric vehicles (PEV)
- simulators for tests of PEV or EVSE

3 Programming

The embedded eMMC flash can be programmed via Freescale's MfgTool. This uses the USB boot mode of the i.MX28, which can be enabled via Jumper JP6 and JP7. In this mode the MFG Tool writes a small firmware image to the RAM of the i.MX28, which is able to access the eMMC flash.
MfgTool needs access to the processor via USB. Therefore the USB Jack J3 was connected in a way that allows this port to be used as USB device instead as host (as the USB A connector suggests). To connect your PC running MfgTool to EVAChargeSE you need a USB-A-Male to USB-A-Male cable. Alternatively you could also use an ordinary USB-A to USB-B cable and an adapter.

4 Mechanical Dimensions

The mechanical dimensions of this product are shown in figure 1

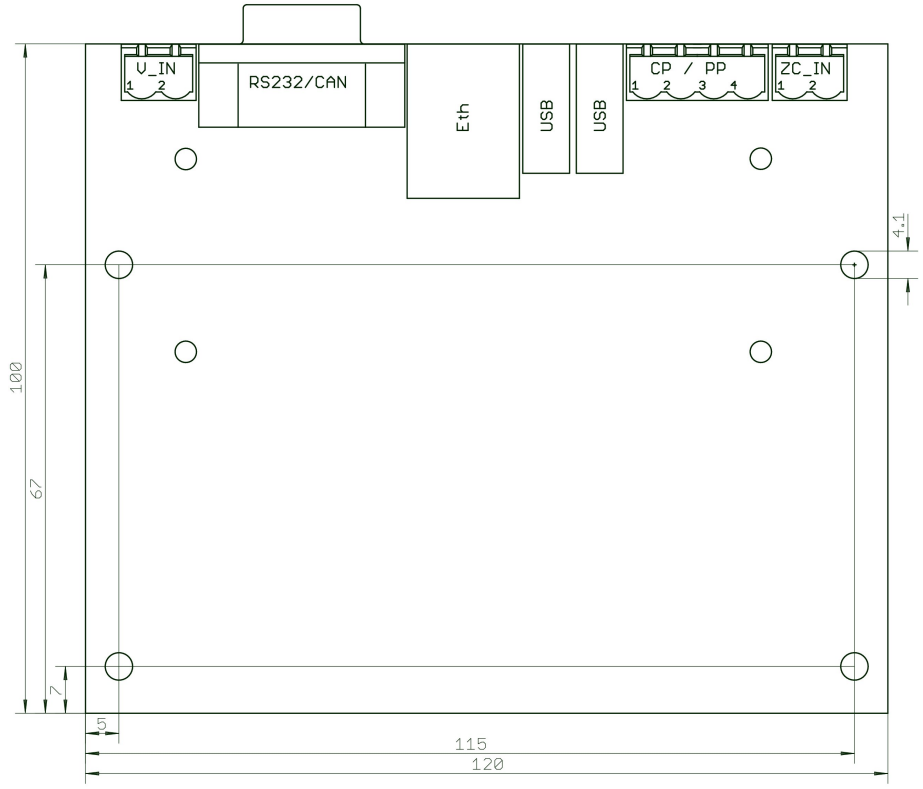


Figure 1: Mechanical drawing of EVACharge SE including connectors

5 LEDs

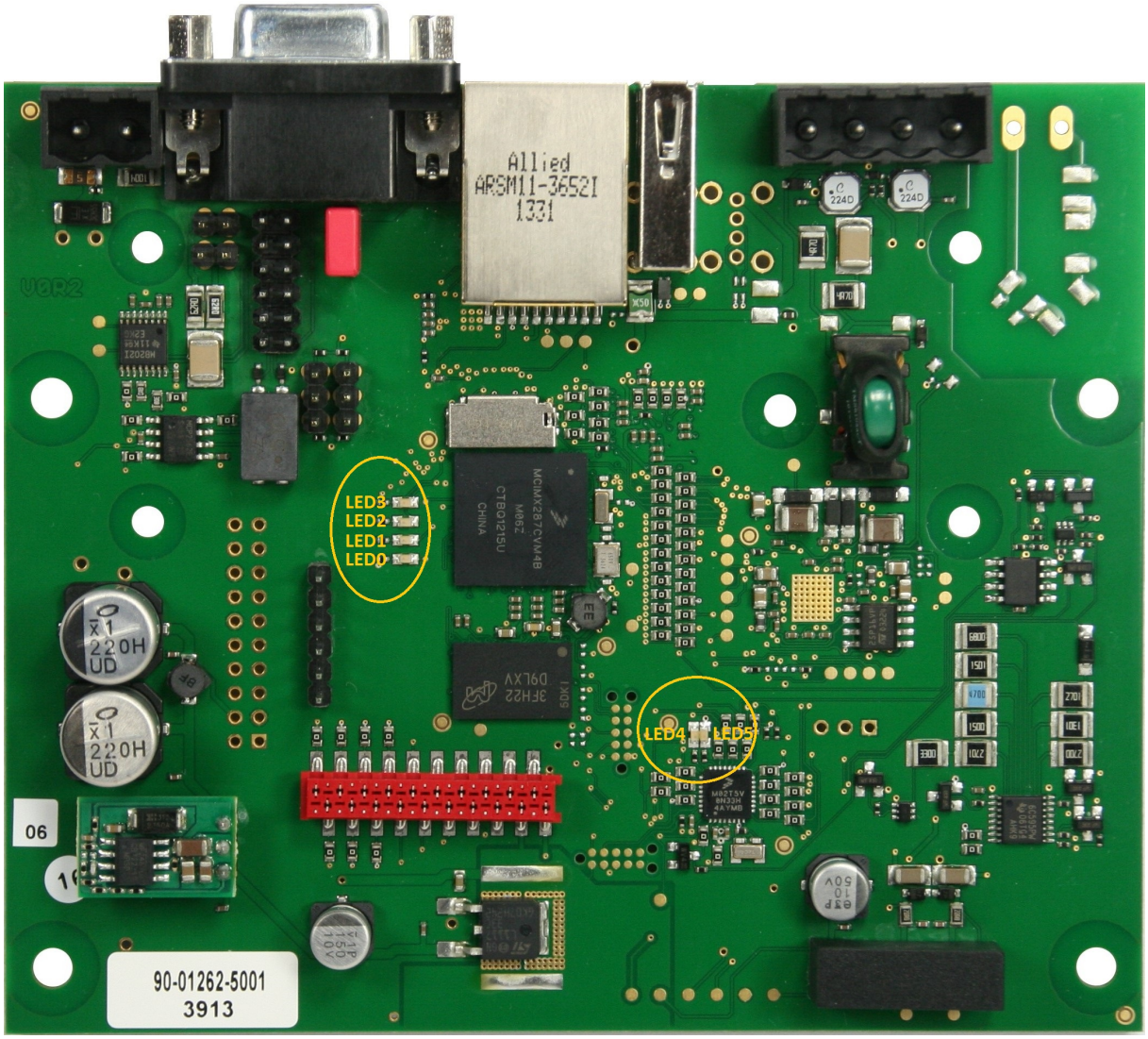


Figure 2: LEDs on EVSE

5.1 LED0 (red)

The LED is connected to the i.MX28 pin K4 / AUART1_TX. Default behaviour is the Linux heartbeat of the i.MX28.

5.2 LED1 (yellow)

The LED is connected to the i.MX28 pin L4 / AUART1_RX.

5.3 LED2 (yellow)

The LED is connected to the i.MX28 pin T3 / LCD_D16.

5.4 LED3 (green)

The LED is connected to the i.MX28 pin P2 / LCD_D08.

5.5 LED4 (yellow)

The USS command indicator of the coprocessor. Each time an USS command is received by the host and processed the LED flashes for a short time. If the coprocessor stays in bootloader (immediatly after reset for a few seconds) this LED is always on.

5.6 LED5 (green)

The coprocessor life sign. The LED blinks in a second interval showing the coprocessor is working and alive. If the coprocessor stays in bootloader (immediatly after reset for a few seconds) this LED is always off.

6 Connectors

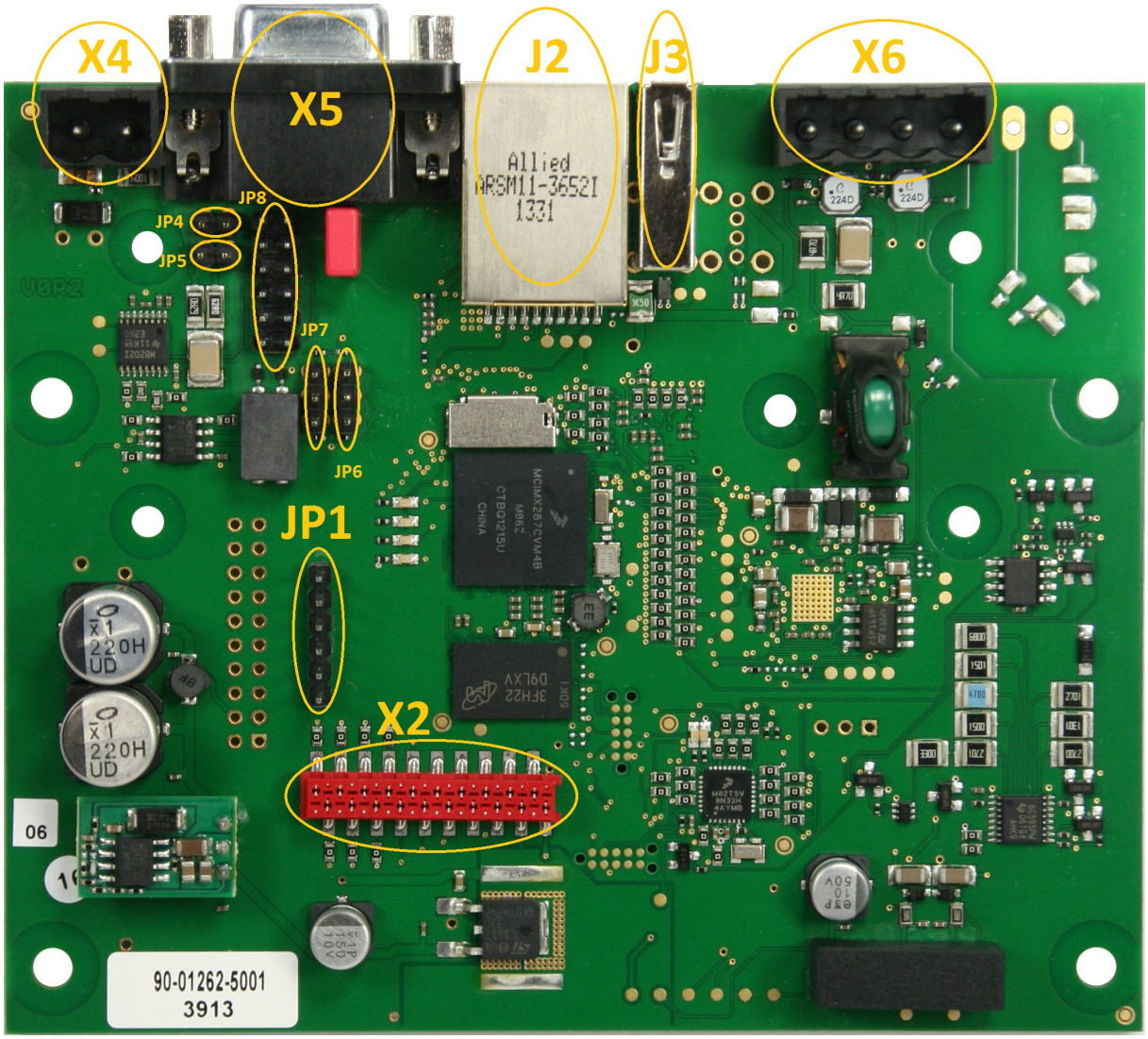


Figure 3: EVSE board with marked connectors

6.1 Debug UART (JP1)

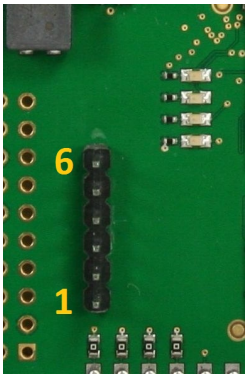


Figure 4: Debug UART in detail

JP1 Pin	Name	i.MX28 Pin
1	GND	-
2	not connected	-
3	not connected	-
4	RX of i.MX28	K7 / PWM0
5	TX of i.MX28	L7 / PWM1
6	not connected	-

This pinout is compatible with a variety of USB/RS232 adapters. Preferable you should use the FTDI cable ”TTL-232R-3V3” or similar, do not use long wires to connect the debug UART.
ATTENTION: Do not use generic RS232 adapters, as they usually have +-12 V voltages for their logic signals. The pins here are only 3.3 V tolerant. You may damage the debug UART with incompatible adapters.
Use the following settings to connect to the debug UART:

Setting	Value
Baud Rate	115200
Data bits	8
Stop bits	1
Parity	None
Flow control	None

6.2 Voltage Input (X4)

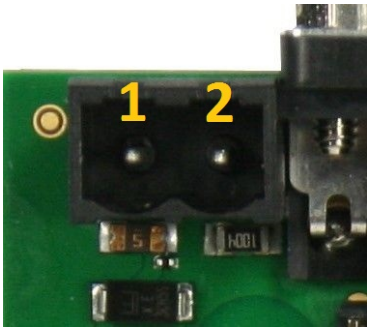


Figure 5: Voltage input connector

This connector is Ria Connect 31177102 (Series 177). The wire for the power supply does not need to be a specific length or cross-section. You should make sure that the cross-section fits the connector you choose and the length does not add too much voltage drop.

X4 Pin	Name
1	+VIN
2	GND

6.3 CAN / RS232 connector (X5)

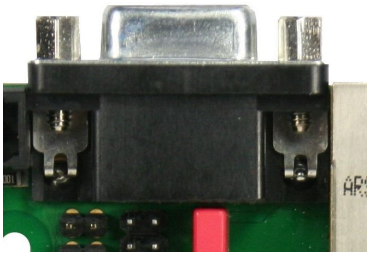


Figure 6: The CAN/RS232 connector

This connector can be used to connect to the i.MX28 using CAN or RS232. Which one of these two interfaces is active is selected by Jumper JP8. To set the board to CAN mode set the jumpers as shown in figure 7. To set the board to RS232 mode set the jumpers as shown in figure 8. The CAN interface is provided of the i.MX28 pins AUART3_RTS (MUX1-CAN1_RX) and AUART3_CTS (MUX1-CAN1_TX). The RS232 interface is connected to the i.MX28 pins AUART0_TX (MUX0) and AUART0_RX (MUX0).
The mating face of the connector is: #4-40 Fixed Jackscrew.
The connected cable should be shorter than two meters.

Pinout of X5 in both modes

X5 pin	Function in CAN Mode	Function in RS232 Mode
1	not connected	not connected
2	CAN_L	UART TX
3	GND	UART RX
4	not connected	not connected
5	not connected	GND
6	not connected	not connected
7	CAN_H	not connected
8	not connected	not connected
9	not connected	not connected

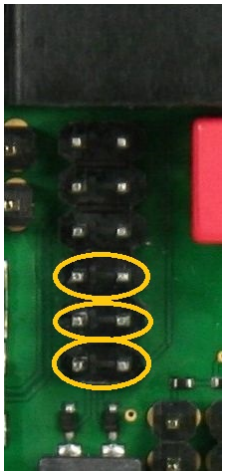


Figure 7: JP8 in CAN MODE

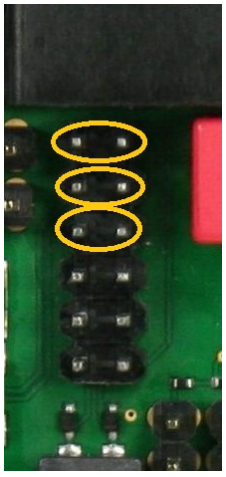


Figure 8: JP8 in RS232 MODE

6.4 CAN termination (JP4 and JP5)

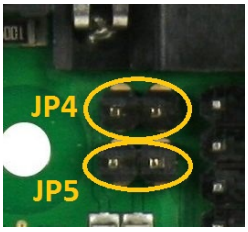


Figure 9: CAN termination jumpers JP4 and JP5

The CAN termination resistor of 120 Ω can be enabled by setting **both** jumpers JP4 and JP5. Removing both jumpers disables the termination resistor.

6.5 Ethernet J2

The Ethernet port supports 10/100 MBit/s and has embedded link and activity LED indicators.
The ENET0 interface of the i.MX28 is connected to a SMSC LAN8720 controller (PHY). The i.MX28 ENET0_RX_CLK pin is connected to the reset pin of the PHY. The i.MX28 pin ENET0_TX_CLK is connectet to the pin nINT of the PHY. It is also connected to a pull up resistor.
The connected cable should be shorter than two meters.

6.6 USB J3

The EVSE usually acts as USB host at this port. Up to 500mA can be drawn from this port.
This USB interface is connected to the USB0 interface of the i.MX28. This interface is used as USB **DEVICE** in USB boot mode.
The connected cable should be shorter than two meters.

6.7 Control and Proximity pilot connector (X6)

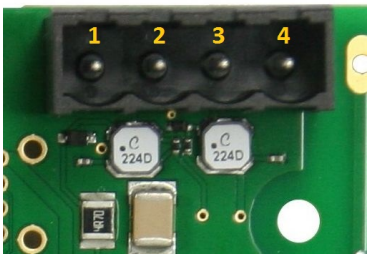


Figure 10: Control and proximity pilot connector

The connector is used for connecting to EV or EVSE (depends on the mode). This connector is Ria Connect 31177104 (Series 177).

X6 Pin	Function	Signal direction in EV mode	Signal direction in EVSE mode
1	Signal ground	-	-
2	Control pilot signal	Input	Output
3	Proximity pilot signal	Input	-
4	Signal ground	-	-

6.8 Zero Cross Detection (X3)

The connector is used to connect the mains voltage to it. This needs to be an AC voltage, polarity does not matter.
This connector is only used for EVSE. The zero cross detection is used for PLC synchronization to the mains frequency.
This connector is Ria Connect 31177102 (Series 177).

6.9 Daughter board connector (X2)

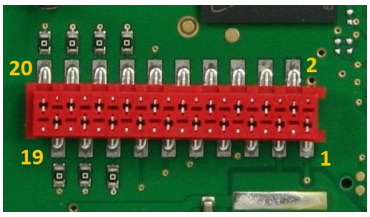


Figure 11: Daughter board connector

Use this Micromatch header to connect additional GPIOs and/or motor driver (i.e. for plug locks).

X2 Pin	Function	Signal direction
1	-12 V power supply (do not load with more than 1 mA)	Output
2	Signal ground	-
3	+3.3 V power supply	Output
4	+12 V power supply (do not load with more than 1 mA)	Output
5	+VIN power supply (fused from X4)	Output
6	+5 V power supply	Output
7	Direction (phase) of lock motor 1	Output
8	Enable of lock motor 1	Output
9	Direction (phase) of lock motor 2	Output
10	Enable of lock motor 2	Output
11	End switch of motor 1 (0..3,3V)	Analog input
12	End switch of motor 2 (0..3,3V)	Analog input
13	Low active fault pin of motor driver	Input
14	IMX_HEARTBEAT from i.MX28: K4 / AUART1_TX	Output
15	DAUGHTER_GPIO_1 from i.MX28: R1 / LCD_D10	Input/Output
16	DAUGHTER_GPIO_0 from i.MX28: P1 / LCD_D97	Input/Output
17	DAUGHTER_GPIO_5 from i.MX28: N1 / LCD_DOTCLK	Input/Output
18	DAUGHTER_GPIO_4 from i.MX28: M1 / LCD_HSYNC	Input/Output
19	DAUGHTER_GPIO_3 from i.MX28: L1 / LCD_VSYNC	Input/Output
20	DAUGHTER_GPIO_2 from i.MX28: K1 / LCD_WR_RWN	Input/Output

6.10 Boot selector jumpers for i.MX28 (JP6 and JP7)

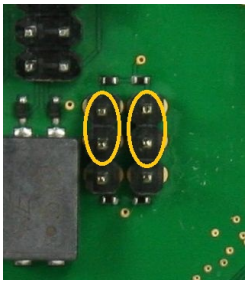


Figure 12: Jumper JP6 and JP7 for booting from eMMC

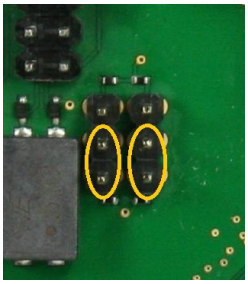


Figure 13: Jumper JP6 and JP7 for booting from USB (J3)

The both jumper settings select the boot source for the i.MX28.

6.10.1 Revision V0R4 and later

Since revision V0R4 it is not necessary to place the jumper for booting from eMMC, you only need jumpers for USB Boot as in figure 13.

7 Internal Connections

7.1 i.MX28 to KL02

Signal Name	i.MX28 Pin	KL02 Pin
UART i.MX28 RX, KL02 TX	F6 / AUART2_RX	PTB1
UART i.MX28 TX, KL02 RX	F5 / AUART2_TX	PTB2
KL02 Reset	T5 / LCD_D22	PTA1

7.2 i.MX28 to QCA7000

i.MX28 Pin	QCA7000 Pin
D8 / I2C0_SDA	SERIAL_IO[0]
A3 / SSP2_SCK	SERIAL_IO[1]
C4 / SSP2_SS0	SERIAL_IO[2]
B3 / SSP2_MISO	SERIAL_IO[3]
C3 / SSP2_MOSI	SERIAL_IO[4]
U2 / LCD_D14	GPIO_0
U3 / LCD_D15	GPIO_1
U4 / LCD_D18	GPIO_2
U5 / LCD_D21	GPIO_3
T2 / LCD_D13	Reset_L

7.3 i.MX28 to eMMC

The eMMC is connected to the i.MX28 via the SSP0 interface. This bus is 8 bit width. The card detect pin of the i.MX28 is pulled low. Pin D3/SSP2_SS1 of the i.MX28 is connected to the eMMC reset pin.

8 Mounting Holes

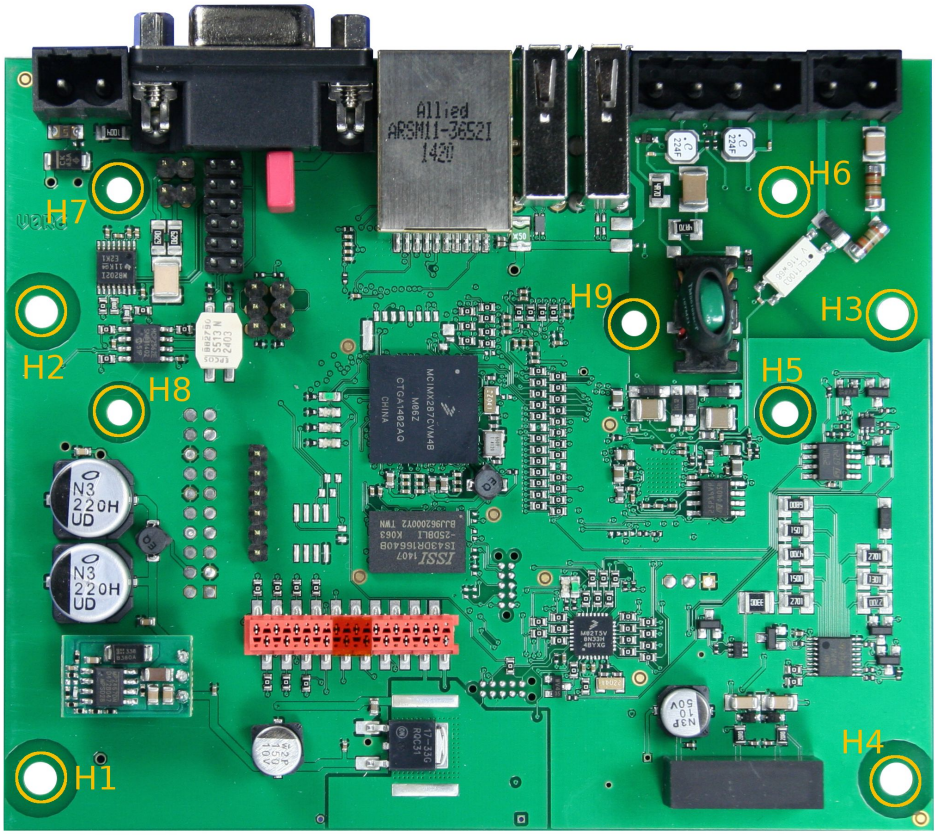


Figure 14: Mounting Holes of EVACharge SE

EVACharge SE supports nine mounting holes, as shown in figure 14.

- H1. . . H4 - main PCB mounting holes
- H5. . . H9 - daughter PCB mounting holes

Note: Please only use isolating mounting material with mains connected to X3 especially for H6.

9 Technical Data

9.1 Absolute Maximum Ratings

SYMBOL	PARAMETER	Min.	Max.	UNIT
VCC	Supply voltage (from revision V0R4)	0	30	V
VCC	Supply voltage (until revision V0R3)	0	+18 (1s)	V
V_CP	Control pilot voltage	-14.0	+14.0	V
V_PP	Proximity pilot voltage	-0.8	+5.0	V
TSTORE	Storage temperature	-40	+85	℃
RAH	Relative air humidity (non condensing)	0	85	%
ZC_OC	Zero Cross Detection Overvoltage Category		CAT III	
DP	Degree of Pollution		2	
ALT	altitude above sea level		2000	m

9.2 Operating conditions

SYMBOL	PARAMETER	Min.	Typ.	Max.	UNIT
VCC	Supply voltage (from revision V0R4)	7	12	27	V
VCC	Supply voltage (until revision V0R3)	11.4	12	12.6	V
TCASE	Top of case temperature	-40	-	85	℃
VZC	Zero Cross Detection Voltage	85		260	V

10 Software Access

10.1 Controlling the on board coprocessor

The two processors on the board are connected via UART in a master/slave configuration which uses the protocol defined in this section. The i.MX28 is the master, while the KL02 is the slave processor.
The following features can be controlled via this protocol:

- PWM generation
- PWM amplitude detection
- PWM loading
- Proximity voltage readout

- Proximity loading
- Controlling lock motors
- Controlling the push button simple connect feature of the Powerline Chip
- Management features to detect and manipulate working state of the coprocessor

Generally speaking you can use this control to switch the board to be EVSE or EV. It only depends on the way you interface the coprocessor.
The UART is configured as described in 1.

Setting	Value
Baud Rate	57600
Data bits	8
Stop bits	1
Parity	None
Flow control	None

Table 1: UART settings

The protocol uses the USS frames as annotated in 2. Each row of such a service table has the length of one byte. Most of the communication follows the request – response principle: for every request sent a response is expected. There are exceptions to this principle where response messages can be sent without a prepending request, which are called auto response (see 3).
If the master controller sends no USS message within five minutes the coprocessor automatically executes a controlled reset. The integrated bootloader is active immediatly after reset for a few seconds. Hence it is ensured to execute firmware updates even if any blocking fault occurred in the mean time.

STX	LGE	ADR	Service	Data 1	...	Data n	BCC
Start of message (0x02)	Length of message (Adr + service + data + BCC)	Target address (device address)	Requested service (command)	Payload			Block check sum

Table 2: Serial protocol message frame

Currently all messages have the address 0.

Services are defined in 3 below.
The response has bit 7 (MSB) of the service byte always set, therefore this bit also defines the direction of the communication.
The block check sum (BCC) of a message links all bytes (STX . . . data n) with the previous sum of all bytes by XOR, starting with a value of 0. For example an typical USS packet for requesting the device test has the content 0x02, 0x03, 0x00, 0x01, 0x00. The last byte is BCC with 0x00 which results from 0x02 ˆ 0x03 ˆ 0x00 ˆ 0x01.

Service id in request	Auto response	Application
0x01		Device-test functions
0x04		Device-test functions 2
0x10		Get PWM
0x11		Set PWM
0x12		Control PWM
0x14		Get Ucp
0x15		Set Ucp
0x17		Lock/unlock cable
0x18		Lock/unlock cable 2
0x1A		Get lock motor fault pin
0x20		Set cyclic process data
0xC0	x	Cyclic process data
0x31		Push button simple connect
0x33		Executes a software reset on device
0xB3	x	Is sent by device after reset
0x50		Activate proximity pilot resistor
0x51		Enable or disable the pullup resistor of the proximity pin
0x52		Get the voltage of the proximity signal

Table 3: Service definition for the serial protocol

10.1.1 Device test (service 0x01)

This service gives access to system reset causes as well as the Software and the Hardware version (one byte each) of the coprocessor.

USS request for device test

STX	LGE	ADR	Service	BCC
0x02	0x03	0x00	0x01	

USS response to device test

STX	LGE	ADR	Service	Sw. Nr.	Hw. Nr.	Reset reason	BCC
0x02	0x06	0x00	0x81	Software version number	Hardware version number	Last reset reason	

The last reset reason status byte is represented by bits (more than one can be set) defined in table 4. The bits in 5 extend the reason with more possible flags.

Bit	Flag
0	Power-on Reset (active = 1)
1	External Reset
2	Brown-out Reset
3	Watchdog Reset
4	JTAG Reset
5	0
6	0
7	0

Table 4: Last reset reason flags

10.1.2 Device test 2 (service 0x04)

This service gives access to reset causes (i.e. why the coprocessor restarted) as well as the software build number.

USS request for device test 2.

STX	LGE	ADR	Service	BCC
0x02	0x03	0x00	0x04	

USS response to device test 2

STX	LGE	ADR	Service	Build Low	Build High	Reset reason	BCC
0x02	0x06	0x00	0x84	Build number		Last reset reason	

The last reset reason status byte is represented by bits (more than one can be set) defined in table 5. It is an extension to the reset flags defined in 4.

10.1.3 Get PWM (service 0x10)

The pulse width of the PWM signal can be read by sending the device-get PWM service. The request is shown in the table below.

Bit	Flag
0	Reset due an internal error while entering stop mode.
1	Reset due an internal core lockup.
2	Software reset (i.e. due reset service request)
3	Loss of clock reset
4	Wakeup reset
5	0
6	0
7	0

Table 5: Last reset reason flags

USS request to read the pulse width

STX	LGE	ADR	Service	BCC
0x02	0x03	0x00	0x10	

The response is given in the table below with the resolution of the signal as 0.1 %, meaning the value “500” corresponds to 50,0 % PWM.

USS response to pwm read request

STX	LGE	ADR	Service	Fi Low	Fi High	Ti Low	Ti High	BCC
0x02	0x07	0x00	0x90	Measured frequency		Measured duty cycle		

10.1.4 Set PWM (service 0x11)

The pulse width of the PWM signal can be set by sending the device-set PWM service with the resolution 0.1 % and modulation frequency Fi in Hz (normally 1000). This command requires that the PWM generation is already on. The corresponding request and response are given in the tables below.

USS request to change pulse width and frequency

STX	LGE	ADR	Service	Fi Low	Fi High	Ti Low	Ti High	BCC
0x02	0x07	0x00	0x11	Frequency to set		Duty cycle to set		

USS response to set the pulse

STX	LGE	ADR	Service	Error code	BCC
0x02	0x04	0x00	0x91	Response error code (See table 6)	

Error code	Description
0	No error
1	Invalid parameter

Table 6: Error code

10.1.5 Control PWM (0x12)

The control PWM service turns the generation of the PWM on or off or queries the state. This allows you to switch roles between EVSE and EV via software control.

USS request to control the PWM generation

STX	LGE	ADR	Service	Control Code	BCC
0x02	0x07	0x00	0x12	Control code (see table 7)	

Control code	Description
0	disable PWM generation
1	enable PWM generation
2	query PWM generation status

Table 7: Control Code

USS response to control service messages

STX	LGE	ADR	Service	Status code	BCC
0x02	0x07	0x00	0x92	Response code (See table 8)	

Error code	Description
0	PWM generation is disabled
1	PWM generation is enabled

Table 8: Status code

10.1.6 Get CP voltage (service 0x14)

Device-Get-Ucp is the request for the control pilot (CP) voltage. Due to the fact that the voltage is changing with 1 kHz, the highest and lowest voltage value will be measured. The data resolution is 10 bit. The measuring limit is set by the maximum of $\pm$15 V. The resolution is 29 mV/bit. The corresponding request and response are given in the tables below.

USS request to get the CP voltage

STX	LGE	ADR	Service	BCC
0x02	0x03	0x00	0x14	

USS response to get the CP voltage. Bit 15 of the negative level represents the leading sign.

STX	LGE	ADR	Service	Pos CP Low	Pos CP High	Neg CP Low	Neg CP High	BCC
0x02	0x07	0x00	0x94	pos. voltage		neg. voltage		

10.1.7 Set CP voltage (service 0x15)

This service switches the load resistors for the CP signal.
The status of every switch will be stated in the parameter as independend bits, where a bit that is set (1) means that the load resistor is connected and a reset bit (0) means that the resistor is not connected. The parameter resistance of the response should match the request parameter.

Bit position	resistor value
0	2.7 kΩ
1	1.3 kΩ
2	347 Ω
3..7	reserved

Table 9: Possible CP resistance bits

The corresponding request and response are given in the tables below.
USS request to set the resistors for the EV-State-circuit. The parameter resistance is defined only by the 3 LSB bits.

STX	LGE	ADR	Service	Resistance	BCC
0x02	0x04	0x00	0x15	See table 9	

USS response to set the resistors for the EV-State-circuit.

STX	LGE	ADR	Service	Resistance	BCC
0x02	0x04	0x00	0x95	See table 9	

10.1.8 Lock / unlock (service 0x17 and 0x18)

The device supports two separate locks for locking the charging sockets. Lock 1 can be controlled via service id 0x17, lock 2 via service id 0x18.
The locks can be locked and unlocked with the request given in table below with the parameter lock command as defined in 10.

Value	Meaning
0	unlock the socket
1	lock the socket
2	query status of the limit switch
3..255	Reserved

Table 10: Lock command meaning

A connected DC motor is driven into either lock or unlock position depending on the command value. The response to the command is send immediately after the request.
The locking motor of each channel is driven for the fixed period of 0.5 seconds.
For a qualified status a separate query command should be sent after this time to get the lock state.
The motor fault service could be used to get more information about failures.

STX	LGE	ADR	Service	Lock command	BCC
0x02	0x04	0x00	0x17 or 0x18		

The corresponding response is given in table below.

STX	LGE	ADR	Service	Lock status	BCC
0x02	0x04	0x00	0x97 or 0x98	(see table 11)	

Value	Meaning
0	open
1	closed
2	not connected

Table 11: Lock status meaning

10.1.9 Get lock motor fault (service 0x1A)

The full bridge drivers for both, lock motors 1 and 2, have a fault pin. This fault pin is activated in case one or both full bridges detect the following problems:

- Overcurrent condition
- Undervoltage condition
- Thermal shutdown

The motor fault pin is wired OR of all possible conditions of both full bridges.
The corresponding request is given in the table below.

STX	LGE	ADR	Service	BCC
0x02	0x03	0x00	0x1A	

The corresponding response is given in the table below. The status code is 0 if the motor fault pin is not activated. The status code is not 0 if the motor fault pin is activated.

STX	LGE	ADR	Service	Status code	BCC
0x02	0x04	0x00	0x9A		

10.1.10 Push button simple connect (service 0x31)

Push button simple connect can be executed by sending the request given in the table below. The request parameter can be *1 = execute*. Optionally the time to wait before execution in ms (i.e. 2..255) can be given with this parameter.

STX	LGE	ADR	Service	Parameter	BCC
0x02	0x04	0x00	0x31		

The corresponding response is given in table below with the parameter stating the occurrence of errors, 0 = executed without errors, 1 = invalid parameter.

STX	LGE	ADR	Service	Error code	BCC
0x02	0x04	0x00	0xB1		

The response is given immediatly independent of the delayed execution time. After the requested delay the command activates the push button simple connected for approx. 1 second. If other push button simple connect services arrive within this time they lengthen the activation time accordingly.

10.1.11 Set cyclic process data (service 0x20)

The device is able to send cyclic data messages in a given interval. The messages contain the PWM values, the CP voltage and the plug lock status. The device-set request is given with the parameter interval. Valid values are in the range of 0..FF, where 0=off, 1=100ms, 2=200ms, etc..
Cyclic USS request for PWM values, CP voltage and plug lock status. The parameter interval sets the time interval in which the cyclic messages from the device are expected.

STX	LGE	ADR	Service	Interval	BCC
0x02	0x04	0x00	0x20		

USS response to the request for cyclic messages. The parameter status shows the status for cyclic messages, 0=off, 1=active.

STX	LGE	ADR	Service	Status	BCC
0x02	0x04	0x00	0xA0		

The cyclic message from device to host

STX	LGE	ADR	Service	Ti low	Ti high	Pos CP low	Pos CP high	Neg CP low	Neg CP high	Lock status 1	Lock status 2	BCC
0x02	0x0B	0x00	0xC0	duty cycle		pos. voltage		neg. voltage		Lock status (see table 11)		

The measured duty cycle has the resolution of the signal as 0.1 %, meaning the value “500” corresponds to 50,0 % PWM.
The data resolution of the measured voltages is 10 bit. The measuring limit is set by the maximum of $\pm$15 V. The resolution is 29 mV/bit.

10.1.12 Reset command (service 0x33)

The command executes a software reset on device. No direct response is sent. The host may wait on the POR message sent by the device on power up.

STX	LGE	ADR	Service	BCC
0x02	0x03	0x00	0x33	

10.1.13 POR message (service 0xB3)

The message is sent by the device on power up.

STX	LGE	ADR	Service	Status	BCC
0x02	0x04	0x00	0xB3	0x00	

The sent status byte is currently set to 0.

10.1.14 Activate proximity pilot resistor (service 0x50)

This service enables or disables resistors that load the proximity signal. These resistors are switched between proximity and GND.

Proximity resistor enable request

STX	LGE	ADR	Service	Control	BCC
0x02	0x04	0x00	0x50	(see table 12)	

Value	Resistor value
0	2700 Ω
1	150 Ω
2	487 Ω
3	1500 Ω
4	680 Ω
5	220 Ω
6	100 Ω
7	Off
8. . . 255	reserved

Table 12: values of Control Byte

Proximity resistor enable response

STX	LGE	ADR	Service	Error code	BCC
0x02	0x04	0x00	0xD0	(see table 13)	

Error code	Description
0	No error
1	Invalid parameter

Table 13: Activate Proximity resistor error codes

10.1.15 Enable or disable the pullup resistor at the proximity signal (service 0x51)

There is a pullup resistor of 330 Ohm to +5 V at the proximity pilot signal which can be activated or deactivated with this service. *Control=0* deactivates the pullup, *all other values* activate the pullup. The request and the corresponding response are given in the tables below.

Proximity pullup enable request

STX	LGE	ADR	Service	Control	BCC
0x02	0x04	0x00	0x51		

Proximity pullup enable response

STX	LGE	ADR	Service	Status	BCC
0x02	0x04	0x00	0xD1	0x00	

10.1.16 Get the voltage of the proximity signal (service 0x52)

The service requests the measured voltage at the proximity pilot pin. The resolution is 29 mV/LSB. The request and corresponding response are given in the tables below.

Proximity pilot voltage request

STX	LGE	ADR	Service	BCC
0x02	0x03	0x00	0x52	

Proximity pilot voltage response

STX	LGE	ADR	Service	Low byte voltage	High byte voltage	BCC
0x02	0x05	0x00	0xD2	The measured voltage		

10.2 MAC Addresses

The board has three MAC addresses programmed in the OTP memory of the iMX28 Processor. Each OTP register is 32 bit / 4 byte long while each MAC address is 6 byte long. iMX28 Linux support suggests to have the last 4 Byte of each MAC address defined in the OTP section.
The first is for the eth0 wired ethernet interface. The second one is for the qca0 interface of the iMX28 - i.e. it is the MAC address of the linux on the powerline network. The third MAC address is for the QCA7000 spi to powerline bridge. This device is not only a bridge but also a MAC end point and thus needs it's own firmware. Hence it has no non-volatile memory the host processor (the iMX28) needs to tell it it's address on startup.

OTP Register Name	**use**
HW_OCOTP_CUST0	the 4 least significant bytes of eth0 mac address
HW_OCOTP_CUST1	the 4 least significant bytes of qca0 mac address
HW_OCOTP_CUST2	the 4 least significant bytes of the QCA7000 device mac address

The stored information therefore does not include all information you need to have a complete MAC address. You need to prepend the fixed value "00:01" to each of the programmed MAC Address fragments to complete the information. This gives you an address in the range 00:01:87:XX:XX:XX for each of the interfaces.

11 How to identify the PCB revision number

Depending on the PCB revision number there are some small differences. It can be identified by the marking on the PCB itself. Figure 15 shows a board with revision V0R2.

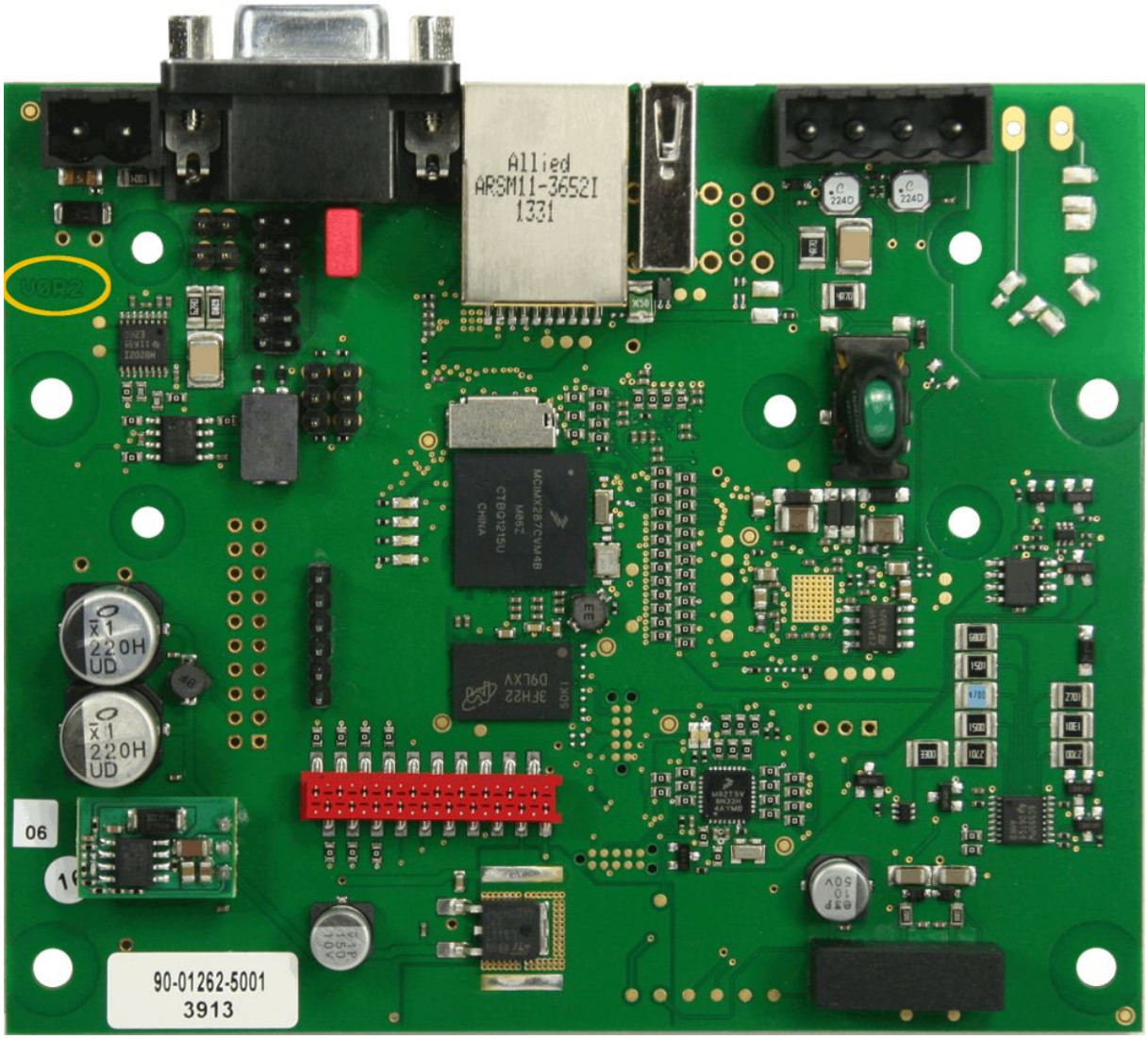


Figure 15: revision marking on PCB

12 Order Information

EVACharge SE can be populated in different configurations.
The default configuration since V0R6 is:

- populated Zero Cross Detection
- 1 USB Port (J3)

The default configuration until V0R6 was:

- no Zero Cross Detection populated
- 1 USB Port (J3)

To save cost and electrical power some parts can be left unpopulated. Please contact *in-tech smart charging* if you have special requirements.

13 Contact

Website: http://smartcharging.in-tech.com
in-tech smart charging GmbH
Friedrich-List-Platz 2
04103 Leipzig
Germany