RRC-PMM35

Power Management Module for integration into all applications using RRC35xx batteries
P/N: 110297

The PMM35 enables internal charging of batteries and facilitates a seamless switch between mains and battery power, ensuring uninterrupted operation and reliable power backup in a space-saving design. Multiple PMMs can be used in parallel inside one device to combine more batteries.

Features & Benefits

- **Easy to design in**
  - Easily integrable into slot design
  - Integrated 90° battery connector for different connection options
  - Maximum flexibility: Various mounting options
  - Small footprint & slim design to not waste space within the application
  - wide DC input voltage range to perfectly match the application's needs
- **Plug & Play available embedded charging solution for RRC standard RRC35xx battery packs**
  - Time to market: no development time, immediate product availability
  - No NRE: no additional development, approvals, or design costs
  - Low total cost of ownership
- **Power management functionality**
  - Seamless switch between mains and battery power
  - Up to 180W charging power in power supply mode
- **Fully Compliant with Smart Battery Specification**
  - SMBus communication with battery and host
- **Worldwide certified for industrial and medical applications**
- **Configurable**
  - Programmable limits for input current, charging current and charging voltage
  - Status signal can directly drive a LED

1. Overview ... 3

2. Pin Configuration and Functions ... 4

3. Specifications ... 6

3.1. Absolute Maximum Ratings ... 6
3.2. Recommended Operating Conditions ... 6
3.3. Electrical Characteristics ... 7
3.4. Lifetime ... 8
3.5. Timing ... 8
3.6. Typical Characteristics ... 9

4. Feature Overview ... 10

4.1. Smart Charging ... 10
4.2. Automatic Switching between AC Adapter and Battery ... 10
4.3. Power Limiting ... 10
4.4. Dynamic Power Management ... 10
4.5. Switchable pull-up resistors ... 11
4.6. LED status indicator ... 11

5. SMBus Communication Interface ... 13

5.1. Using the standard "PMM35 Smart Charger" Functions ... 13
5.2. Using the "PMM35 Extended" Functions ... 18

6. Application Examples ... 24

6.1. Start & Stop Charging on Command ... 24
6.2. Put the battery into shipping mode (device-shutdown, long-term storage and shipping) ... 24
6.3. Use your own pull-up resistors ... 25
6.4. Charging two or more batteries ... 26
6.5. Determine whether a power supply voltage is connected ... 28
6.6. Monitor the power supply current and power ... 28
6.7. Charging the battery gently (life-prolonging) ... 28
6.8. Using the Battery as Backup Power Source (UPS) ... 30
6.9. Inrush Current ... 32

7. EMC ... 34

8. Safety ... 34

9. Regulatory Compliance / Certifications ... 34

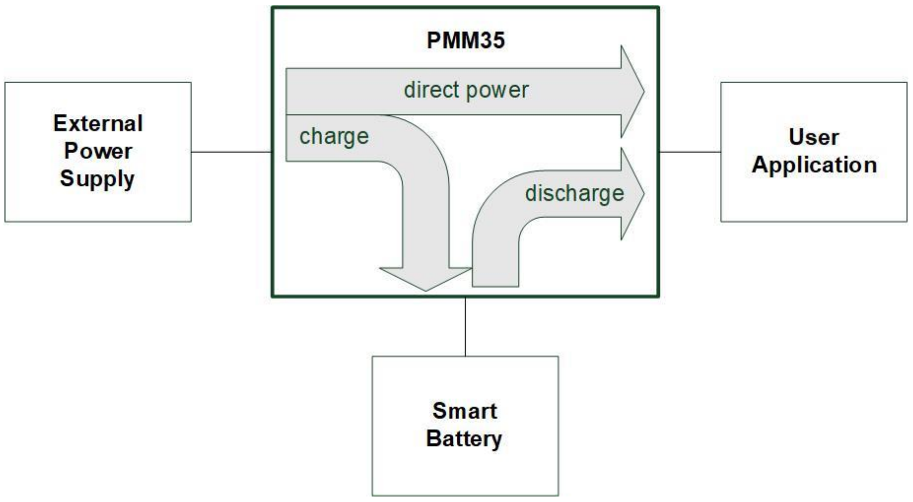
10. Mechanical Information ... 35

10.1. Dimensions ... 35
10.2. Mounting screws ... 36
10.3. Battery insertion ... 37

11. Documentation Support ... 38

12. Revision History ... 38

1. Overview



The PMM35 is a compact and efficient charger designed for smart RRC35xx batteries. It automatically adapts the charging voltage and current to match the battery's requirements, while also taking into account the temperature, following the JEITA standard.

When connected to an AC adapter, the PMM35 charges your battery while powering your device simultaneously. Once the adapter is unplugged, the battery discharges through the PMM35 to keep your device running.

To prevent the AC adapter from overloading, the PMM35 has dynamic power management that limits input power. As your device's power usage increases during charging, the charger reduces charging current to keep the total input current below the adapter rating.

You can communicate with the PMM35 and your battery through SMBus. The charger also features a 3.3 V status signal that can drive an LED to indicate the battery, AC adapter, and error status.

2. Pin Configuration and Functions

Figure 1 shows connector locations of the DC Input & Output, Smart Battery, and User Interface (UI) connectors.

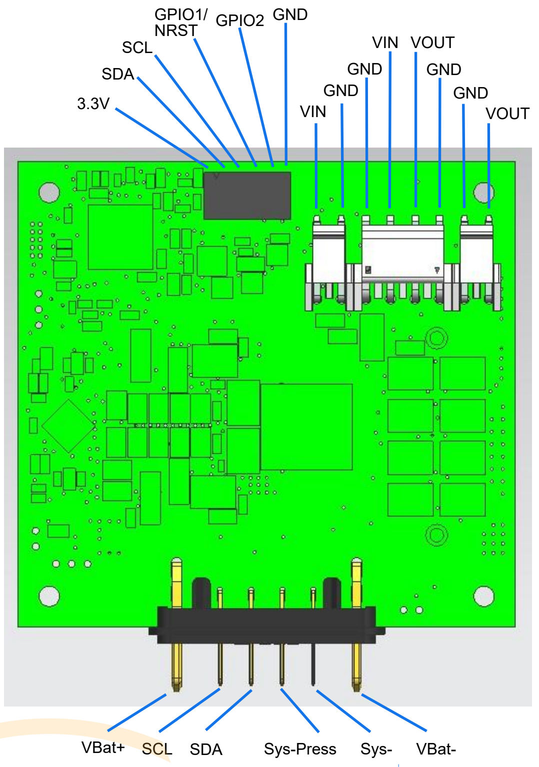


Figure 1 - Connector locations of the DC Input & Output, Smart Battery, and User Interface (UI) connectors.

Connector	Description	
DC Input & Output	This connector functions as both the input and output power connection. Do *not* reverse the polarity	
	Pin 1: V_{out} (connects to the application)	
	Pin 2: GND	
	Pin 3: GND	
	Pin 4: V_{out} (connects to the application)	
	Pin 5: V_{in} (connects to an external power supply)	
	Pin 6: GND	
	Pin 7: GND	
	Pin 8: V_{in} (connects to an external power supply)	
Smart Battery	Smart Battery Connector. Do *not* reverse the polarity.	
	Pin 1: VBat+	Positive voltage pin.
	Pin 2: SMBus Clock (SCL)	Clock signal for SMBus.
	Pin 3: SMBus Data (SDA)	Data signal for SMBus.
	Pin 4: Sys-Press	The PMM35 connects this pin to Sys- to activate the battery.
	Pin 4: Sys-	The PMM35 use this pin to detect the battery.
	Pin 6: VBat-	Ground pin.

User Interface (UI)	Connect to User Application to exchange information between the application, the PMM35, and the battery.	
	Pin 1: VCC (3.3V)	3.3V LDO output for alternative Pull-up Resistors.
	Pin 2: SMBus Data (SDA)	Data signal for SMBus.
	Pin 3: SMBus Clock (SCL)	Clock signal for SMBus.
	Pin 4: NRST (GPIO1)	Pull low for resetting the PMM35.
	Pin 5: LED Indicator (GPIO2)	Connect an LED for outputting status information.
	Pin 6: GND	Ground voltage pin.

Table 1 – Connectors sequence of PMM35.With Smart Battery connector, DC Input & Output connector, and User Interface connector.

Connector mating parts and alternatives:

DC Input & Output	JST VHR-4N JST VHR-4M TE 3-1123722-4 (rated to 8 A) JST VHR-2N JST VHR-2M JST VHR-8N

User Interface	TE 1735447-6 (with latch) JST PHR-6 (without latch)

3. Specifications

3.1. Absolute Maximum Ratings

Stresses beyond the limits listed below may cause permanent damage to the device. These are stress ratings only, which do not imply the device's functional operation at these or any other conditions beyond those indicated under *Recommended Operation Conditions*. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

All voltages are with respect to their corresponding (-) Pin or GND, respectively.

	Description	Min	Max	Unit
Voltage	DC Input +	-0.3	35.0	V
	BAT + or V_{BAT}	-0.5	35.0	
	SMBus Clock SCL	-0.3	VCC+4.0*	
	SMBus Data SDA	-0.3	VCC+4.0*	
	VCC Output	-0.3	4.0	
	GPIO 1	-0.3	4.0	
	GPIO 2	-0.3	4.0	
Current	DC Input		20	A
	DC Output		20	
	GPIO output	-5	5	mA
Storage Temperature		-25	85	°C
Operating Temperature		-20	60	°C

*VCC = 3.3V if the PMM35 is powered by the battery or power supply and is not in the Low-Power-Consumption-Mode.

3.2. Recommended Operating Conditions

All voltages refer to the described pin of the PMM35 and the corresponding (-) pin or GND. Please note that the current flow from the power-supply-unit over the supply lines to the PMM35 can lead to voltage fluctuations at the pins of the PMM35.

	Description	Min	Max	Unit
Voltage	DC Input +	17	26	V
	SMBus Clock SCL	2.6	5.5	
	SMBus Data SDA	2.6	5.5	
Current	GPIO output	0	2.5	mA
Storage Temperature		-20	60	°C
Operating Temperature		-20	60	°C
Relative Humidity, non-condensing		5	95	%
Ambient Pressure		533	1070	hPa
Altitude		-/-	5000	m

3.3. Electrical Characteristics

19V ≤ V_{PSU} ≤ 24 V, -20 °C ≤ T_A ≤ 60 °C, typical values are at T_A = 25 °C, with respect to GND (unless otherwise noted)

Parameter	Description	Min	Typ	Max	Unit
Operating Conditions					
V_{PSU}	AC adapter voltage	19		24	V
Standby Current (in Low-Power-Consumption-Mode)					
I_{SBY}	Battery connected, nothing else			1.0	mA
Charge Voltage					
Range		10		30.45	V
Accuracy	Charge voltage 29.4 V	-0.5		0.5	%
Charge Current & Charge Current Limit					
Default Value			6000		mA
Range		612		6000	mA
Step Width			48		mA
Accuracy	Charge current 0.612 A	-50		50	%
	Charge current 2.016 A	-20		20	
	Charge current 4.032 A	-10		10	
	Charge current 6.000 A	-6		6	
Input Current Limit					
Default Value OCP	>10sec		>20000		mA
	>5sec		>25000		mA
	>2sec		>30000		mA
Default Value Charge Power Reduction			8000		mA
Range		612		20000	mA
Step width			612		mA
Accuracy	Input current 1.0 A	-25		25	%
	Input current 5.0 A	-15		15	
	Input current 10.0 A	-5		5	
	Input current 20.0 A	-3		3	
LED status indicator					
Voltage	Logical Low		0		V
	Logical High		3.3		
Current	Output current	-5		5	mA
Internal Resistance	Internal Resistance → voltage drop		70		Ω

SMBus					
Voltage	Input Low Voltage	-0.3	0	0.95	V
	Input High Voltage	2.4	3.0	5.5	
	Output Low Voltage	0.04	0.06	0.09	
	Output High Voltage	2.85	3.0	3.15	
Internal Soft Start					
I_{STEP}	Soft start current step		128		mA

3.4. Lifetime

	Description	Max	Unit
MTTF	Mean Time To Failure, operating conditions: 5 h charge and 19 h standby or discharge time per day @ 25°C	$5.2 \cdot 10^6$	hours

3.5. Timing

Parameter	Description	Min	Typ	Max	Unit
Overcurrent Protection					
t_{OCP}	Time until PMM35 switches off		>2		s
Internal Soft Start					
t_{STEP}	Soft start current step time		5		ms
Output voltage transition					
$t_{out,AC\text{-}BAT}$	Transition time from AC to battery		100		µs
$t_{out,BAT\text{-}AC}$	Transition time from battery to AC		4		s
SMBus Timing					
f_{SMBus}	Clock frequency	10		100	kHz
$t_{W(H)}$	SCL pulse width high	4.0		50	µs
$t_{W(L)}$	SCL pulse width low	4.7			µs
$t_{SU(START)}$	Setup time for START condition	4.7			µs
$t_{H(START)}$	START condition hold time after which first clock pulse is generated	4.0			µs
$t_{SU(STOP)}$	Setup time for STOP condition	4.0			µs
t_{BUF}	Bus free time between START and STOP condition	4.7			µs
$t_{timeout}$	SMBus release time-out (*)	25		35	ms

(*) Devices participating in a transfer will time out when any clock low exceeds the 25 ms minimum time-out period. Devices that have detected a time-out condition must reset the communication no later than the 35-ms maximum time-out period. Both a master and a slave must adhere to the maximum value specified, as it incorporates the cumulative stretch limit for both master (10 ms) and a slave (25 ms).

3.6. Typical Characteristics

3.6.1. Charge Power

The PMM35 is continuously monitoring its temperature and reduces charge power if the temperature rises beyond 75 °C as shown in Figure .

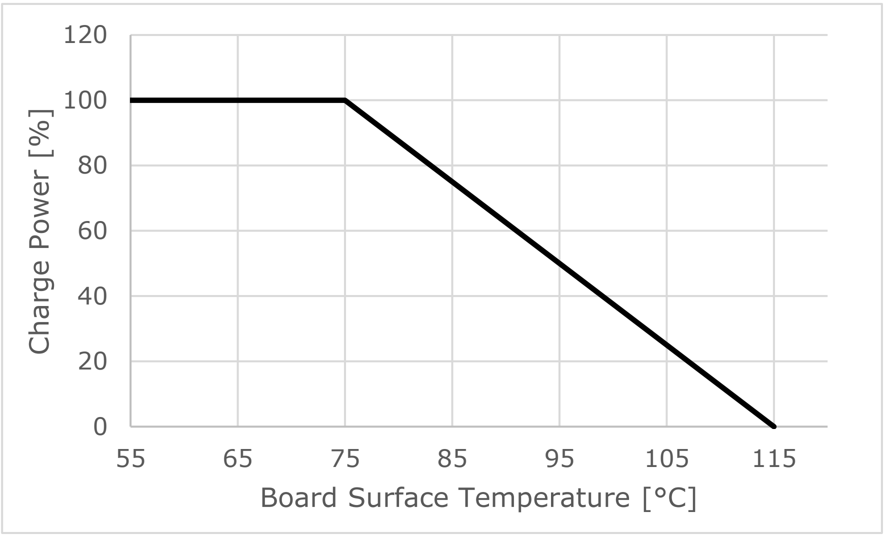


Figure 2 – Charge power derating

3.6.2. Transition from AC to Battery

Upon disconnection of the AC adapter, the battery takes over. The disconnection of the AC adapter is recognized by the PMM35 if the input voltage is <12V. **Figure 3** shows a typical transition of the PMM35 output voltage with an output current of 20A : For a short moment (100 µs typ.), the output voltage gets supplied by an on-board capacity. As the output current gets higher, the capacity discharges more in this time window. For low voltage drops it is recommended to disconnect the AC adapter under light load conditions.

For an even smoother transition it is also possible to activate the battery-path (PowerPathControl()) before you unplug the AC adapter.

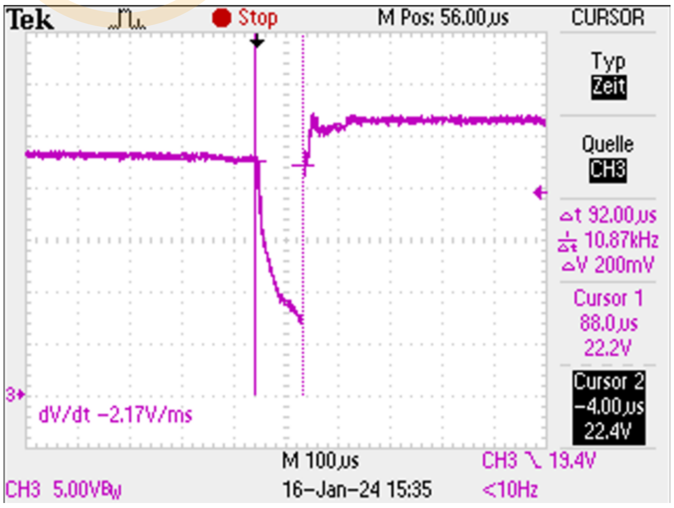


Figure 3 - PMM35 output voltage during a typical transition from AC to the battery at 20A output current.

4. Feature Overview

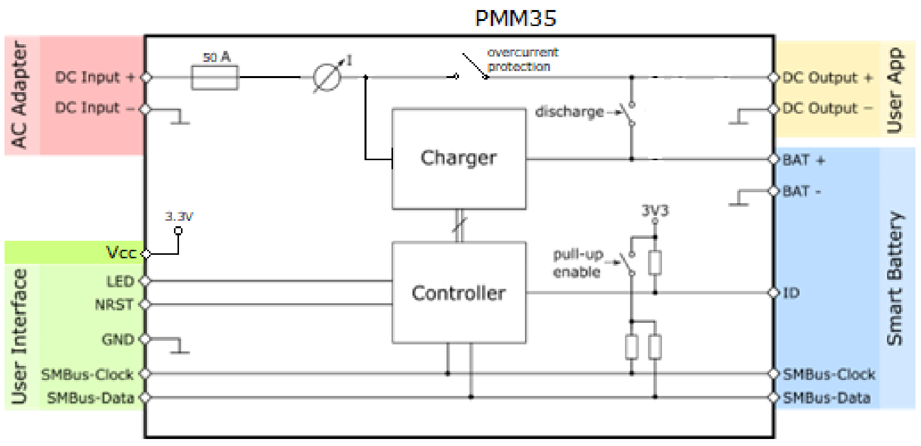


Figure 4 – Functional Block diagram

4.1. Smart Charging

The PMM35 can charge smart batteries automatically. By communicating with them through the SMBus, it reads the temperature-adjusted charging parameters from the battery (as per the JEITA standard) and adapts to them. This allows the PMM35 to charge any lithium-ion smart battery up to 5S-7S configurations.

If the battery cannot communicate via SMBus because it is in shipping mode or deeply discharged, the PMM35 still detects a smart battery and offers a wake-up charge.

4.2. Automatic Switching between AC Adapter and Battery

The PMM35 can switch automatically between the battery and AC adapter. When the AC adapter is detected, the system power source switches from the battery to the adapter in about 100 µs.

If the adapter is removed, the battery takes over and powers the User Application.

Via SMBus commands, you can also activate/deactivate the battery or AC power-paths.

4.3. Power Limiting

Via SMBus commands, you can limit the charging parameters to values below the requested values by the battery.

- Limiting the charging current reduces thermal stress on the battery. If charging time is not crucial for your application, less charging current may be beneficial for battery life expectation and EMI.
- Limiting the charging voltage reduces voltage-related stress in the battery cells. If your application can live with less full charge capacity and hence, a reduced runtime, reducing the charging voltage may increase the battery life expectation strongly.

4.4. Dynamic Power Management

Dynamic power management comes into play if your input power supply is not powerful enough to support both your host application and the battery charging at full rate. By setting an input current limit, the PMM35 can dynamically slow down the charging rate. This reduces the risk of overloading the power supply.

4.5. Switchable pull-up resistors

The PMM35 provides the pull-up resistors necessary for SMBus communication so it can be used as a stand-alone charger.

In case, you want to use a 5V SMBus you can disable the PMM35's pull-up resistors through an SMBus command and provide external pull-up resistors which are connected to your 5V supply.

Do not use the Low-Power-Consumption-Mode if you use a supply voltage >4V.

This is because the max. voltage at the microcontroller pins is 4V higher than the supply voltage of the microcontroller. In Low-Power-Consumption-Mode the microcontroller has a supply Voltage of 0V. Therefore the max. SMBus voltage hast to be limited to 4V in this mode.

4.6. LED status indicator

You can connect an LED to the status indicator pin to display status information. Make sure to include a resistor in series with the LED to prevent the current from exceeding your LED's requirements, or 3 mA, whichever is lower.

The LED will show a blinking pattern according to Table 2.

LED blinking pattern:		
On-time	**Off time**	**Meaning**
–	Permanently off	No power supply connected.
Permanently on	–	Charging.
900 ms	100 ms	Charging interrupted, because of "ChargerMode(). InhibitCharge set to 1.
50 ms	950 ms	Idle.
250 ms	250 ms	Error. See 5.2 for details.
500 ms	500 ms	Wake-up current applied. Used to wake a battery from shipping mode or from deep discharge.
100 ms	100 ms	Error on Initialization (means: PMM35 does not boot).

Table 2 – LED Status Indicator pattern 1.

LED blinking pattern:		
On-time	**Off time**	**Meaning**
–	Permanently off	No power supply connected or idle.
1000 ms	1000 ms	Charging.
Permanently on	-	Fully Charged
200 ms	200 ms	Error. See 5.2 for details.
100 ms	100 ms	Error on Initialization (means: PMM35 does not boot).

Table 3 - LED Status Indicator pattern 2.

LED blinking pattern:		
On-time	**Off time**	**Meaning**
–	Permanently off	No power supply connected.
Permanently on	–	Power supply connected.
200 ms	200 ms	Error. See 5.2 for details.

Table 4 - LED Status Indicator pattern 3.

5. SMBus Communication Interface

The PMM35 charger module uses SMBus for communication with both the battery and the host. To learn more about SMBus communication, including hardware setup and command usage, refer to the "RRC35xx Design-In Guidelines" available from your local RRC salesperson: contact us at sales@rrc-ps.com.

Note that in Low-Power-Consumption-Mode, the PMM35 microcontroller is not running. This prevents the battery from discharging in the PMM35, and thereby, saves standby power. But it also prevents the host from communicating with the PMM35. Additionally, the pull-up resistors will be turned off, so the host needs to provide its own pull-up resistors to communicate with the battery (see EnablePullups() in section 5.2).

As a level 3 charger, the PMM35 charges the battery autonomously based on its requirements. You can also configure and monitor the PMM35 using the commands outlined below.

The SMBus specification provides slave addresses for all bus participants in Table . We have added an additional address for Extended Configuration of the PMM35, which goes beyond the scope of standard SMBus.

Description	7-bit Address	Write Address	Read Address
SMBus Host (your device)	_000 1000 (0x08)	0001 000 **0** (0x10)	0001 000 **1** (0x11)
PMM35 Smart Charger Module	_000 1001 (0x09)	0001 001 **0** (0x12)	0001 001 **1** (0x13)
Smart Battery	_000 1011 (0x0B)	0001 011 **0** (0x16)	0001 011 **1** (0x17)
PMM35 Extended Registers	_011 0011 (0x33)	0110 011 **0** (0x66)	0110 011 **1** (0x67)

Table 5 – SMBus Addresses for Host, Charger, and Battery. To send a write or read command, shift the 7-bit device address to the left, with the 8th bit indicating the read or write operation.

For a quick guide on frequently used configurations, you can skip ahead to the Application Examples section starting on page 24. The rest of this chapter delves into the details of all available PMM35 registers including the extended registers for a more comprehensive understanding.

5.1. Using the standard "PMM35 Smart Charger" Functions

The registers discussed in this section are the standard SMBus charger registers, as defined in the [SBCS][1]. You can access these registers via the 7-bit address 0x09 (refer to Table).

Function	Command	Access	Description
ChargerSpecInfo()	0x11	Read word	Gives information on the supported SMBus specification.
ChargerMode()	0x12	Write word	Sets the charger mode. The default values allow a Smart Battery and the Smart Charger to work in concert without requiring an SMBus host.
ChargerStatus()	0x13	Read word	Returns status information about the charger.
ChargingCurrent()	0x14	R/W word	The battery or the host sends the desired charging current.
ChargingVoltage()	0x15	R/W word	The battery or the host sends the desired charging voltage.
AlarmWarning()	0x16	R/W word	Lets the charger react to BatteryStatus() information sent by the smart battery.

Table 6 – List of Standard SMBus Registers.

[1] Smart Battery Charger Specification, Revision 1.1. You can download it at http://sbs-forum.org/specs/sbc110.pdf.

The [SBCS] also defines optional manufacturer functions. The PMM35 makes use of them to offer the following functionality:

Function	Com-mand	Access	Description
ExtConfigAddress()	0x3c	R/W word	Sets the configuration address for extended configuration.
PcbTemperature()	0x3d	Read word	Temperature of the hotspot on the PMM35 board, near the switching MOSFETs.
InputVoltage()	0x3e	Read word	Voltage at the input connector, as it comes from the input power supply.
InputCurrent-Limit()	0x3f	R/W word	Limits the input current to save the input power supply from overloading.

Table 7 – Usage of the Optional Manufacturer Functions.

All functions of Table 6 and Table 7 are detailed below:

ChargerSpecInfo() 0x11

Using ChargerSpecInfo() allows you to identify the supported revision of the SMBus specification. The PMM35 will always respond with the value 0x0003, indicating support for SMBus specification 1.1, without any Smart Battery selector commands.

ChargerMode() 0x12

Using ChargerMode() allows your host to inhibit the charging process, to reset the charger and to change the default modes.

Bit	Name	Description
15…4	Reserved	–
3	Reset to Zero	0: Charging current & voltage values unchanged (default) 1: Set charging current & voltage values to zero On this command, both ChargingCurrent() and ChargingVoltage() shortly become 0, which inhibits the charging process. However, as soon as the battery sends the next broadcast, or on the next polling event (if "Enable Polling" = 1), the charging process will resume.
2	Power-On Reset	0: Mode unchanged (default) 1: Set ChargerMode() to Power-On defaults (Enable Polling = 1, Inhibit Charging = 0)
1	Enable Polling	0: Disable polling (Level 2 charger) 1: Enable polling (Level 3 charger, default) ChargerStatus() lets you read back the charger level.
0	Inhibit Charge	0: Enable charging (default) 1: Inhibit charging ChargerStatus() lets you read back whether the charging is inhibited or not. Charging is always on enabled after resetting the PMM35 **and** changing the battery.

Table 8 – ChargerMode() register.

ChargerStatus() 0x13

Allows your host to determine the charger status:

Bit	Name	Description
15	AC Present	0: Charge power is not available 1: Charge power is available
14	Battery Present	0: Battery is not present 1: Battery is present
13	Power Fail	0: Input voltage is high enough to charge the battery 1: Input voltage is too low to charge the battery
12	Alarm Inhibited	0: Charging process not inhibited due to an AlarmWarning() 1: Charging process is inhibited after reception of an AlarmWarning() message This bit is cleared if both the ChargingVoltage() and ChargingCurrent() are re-written to the charger, power is removed, or if a battery is removed.
11…8	Reserved	–
7	Voltage Over-range	0: ChargingVoltage() is valid 1: ChargingVoltage() is set too high and cannot be provided by the charger
6	Current Over-range	0: ChargingCurrent() is valid 1: ChargingCurrent() is set too high and cannot be provided by the charger
5…4	Charger Level	00: Reserved 01: Charger is configured as Level 2 charger 10: Reserved 11: Charger is configured as Level 3 charger You can configure the charger level by setting the "Enable Polling" bit in ChargerMode().
3..2	Reserved	–
1	Polling Enabled	0: Charger is in slave mode (polling disabled) 1: Charger is in master mode (polling enabled)
0	Charge Inhibited	0: Charger is enabled 1: Charger is inhibited You can inhibit charging by setting the "Inhibit Charge" bit in ChargerMode().

Table 9 – ChargerStatus() register.

ChargingCurrent() 0x14

The ChargingCurrent() function sets the maximum current that the PMM35 can provide to the battery, but only when configured as a level 2 charger (see "Charger-Status()").

Access	R/W
Data format	Unsigned Integer
Unit	mA
Range	0…65,534 mA
Stored in Flash	No

Only the smart battery is allowed to use the ChargingCurrent() function.[2] If you want to set the charging current to a custom value, please use CustomChargingCurrent() within the extended configuration registers.

Note that PMM35 has a maximum charging current of 6.00 A, so any current higher than that will be limited to that value. Setting ChargingCurrent() to 0 stops the charging process, when configured as a level 2 charger.

ChargingVoltage() 0x15

The ChargingVoltage() function sets the maximum voltage that the PMM35 can provide to the battery, but only when configured as a level 2 charger (see "Charger-Status()").

Access	R/W
Data format	Unsigned Integer
Unit	mV
Range	0…65,534 mV
Stored in Flash	No

Only the smart battery is allowed to use the ChargingVoltage() function.[2] If you want to set the charging voltage to a custom value, please use CustomChargingLimit() within the extended configuration registers.

Note that PMM35 has a maximum charging voltage of 34.8 V, so any voltage higher than that will be limited to that value. Setting ChargingVoltage() to 0 stops the charging process, when configured as a level 2 charger.

AlarmWarning() 0x16

If AlarmWarning() is enabled in the smart battery, it will automatically send status BatteryStatus() information on address 0x16. The PMM35 reacts on the charging-related information in BatteryStatus(). For more information, take a look into the RRC35xx Design-In Guidelines.

ExtConfigAddress () 0x3c

The command ExtConfigAddress() allows you to set, or disable, the address for extended configuration of the PMM35.

Standard-wise, the PMM35 uses the 7-bit address 0x33 for extended configuration, as shown in Table above. You can change this address within the range of 0x01 to 0x7f except for the addresses

- 0x08 (defined as host address by SMBus specifications)
- 0x09 (defined as smart charger address by SMBus specifications)
- 0x0b (defined as smart battery address by SMBus specifications)
- 0x0c (defined as alert response address by SMBus specifications)

By writing "0xeeee", you can also switch off access to the extended configuration registers. You can re-enable access anytime by writing a valid address using this function.

After writing a new value to this register you need to wait 20 ms before sending the next SMBus command to the PMM35, since it needs some time to store the new value in its flash memory.

[2] There is one exception to the rule. If you would turn off Polling via ChargerMode(), and if you would also turn off the battery broadcast, the host *must* use this function to set the charging parameters.

PcbTemperature() 0x3d

The PcbTemperature() function returns the temperature measured on the PMM35 PCB close to the hotspot (inductor and output transistors).

Access	Read only
Data format	Signed Integer
Unit	0.1 °C
Range	-400…1250, corresponding to -40…+125 °C

InputVoltage() 0x3e

The InputVoltage() function returns the voltage measured on the input connector of PMM35 PCB, which is supplied by the external power supply.

Access	Read only
Data format	Unsigned Integer
Unit	mV
Range	0…65,534 mV

InputCurrentLimit() 0x3f

Setting the input current limit can be extremely helpful if your input power supply is not powerful enough to support both your host application and the battery charging at full speed. By setting the input current limit, the PMM35 slows down the charging rate if the total power reaches the input current limit setpoint. This reduces the risk of overloading the power supply.

In both cases illustrated in Figure , the PMM35 works at the Input Current Limit. On the left, the charging rate reduces while the User Application draws much current. On the right, the User Application needs a smaller amount of current, allowing the PMM35 to charge at a higher rate, still within the total input current limit.

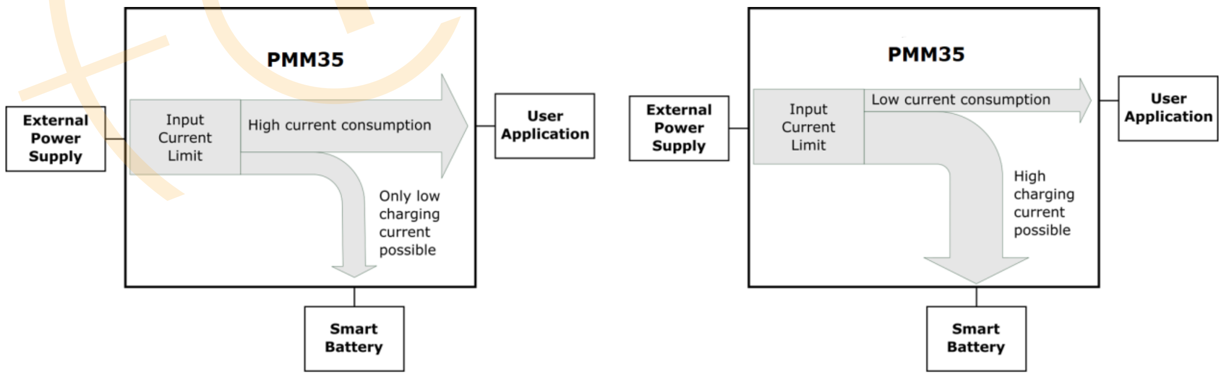


Figure 5 – InputCurrentLimit() adjusts the charging current depending on the User Application's current consumption.

It is important to note that the PMM35 cannot limit the current drawn by the host application; it can only reduce the charging rate. If the input current exceeds its limit, the charger reduces the charging current to allow the input current to drop. If the input current is still above the limit the charger boosts the input supply with current from the battery. This behavior is only available if the battery is charging, otherwise the charging path is switched off. When the load current becomes smaller, the charger soft-restarts to continue charging the battery.

To use the InputCurrentLimit() function, send the value in mA to which the current through the input connector should be limited.

Access	Read and Write
Data format	Unsigned Integer
Unit	mA
Range	1…65,534 mA
Maximum accepted value	20,000 mA
Factory Default	8,000 mA
Stored in Flash	Yes

Note that the InputCurrentLimit() gets capped to a maximum value of 20 A. You cannot set it to any higher value. The Input Current Limit is always active, which reduces the risk of damage to the PMM35 itself.

Once configured, the value of the input current limit is immediately valid, and it is stored in flash memory. Therefore, it is automatically set at the next restart.

The PMM35 can only adjust the current limit in steps of 240 mA, and there may be component tolerances (which are accounted for as the PMM35 has been calibrated during production). As a result, the actual input current limit may not correspond precisely to the set-value, and the actual current can be up to 239 mA lower. However, the PMM35 is programmed to prevent the input current from exceeding the set value. If you want to determine the actual input current limit, you can read it back.

After writing a new value to this register you need to wait 20 ms before sending the next SMBus command to the PMM35, since it needs some time to store the new value in its flash memory.

5.2. Using the "PMM35 Extended" Functions

Besides the standard SMBus charger functions, the PMM35 smart charger module understands extended commands that you can use to configure and monitor the charging module.

You can access these commands through the PMM35 Extended Registers address, which standard-wise is 0x33, but can be set to different address (or even disabled) via ExtConfigAddress() (see section 5.1 above).

Function	Com-mand	Ac-cess	Description
CustomChargingCurrent()	0x60	R/W word	Limits the charging current to a custom value.
CustomChargingLimit()	0x61	R/W word	Limits how much the battery is charged.
PowerLossDetection()	0x62	R/W word	Detect power loss over the output connector.
EnablePullups()	0x63	R/W word	Controls the pull-up resistors on SMBus data and clock lines.
Error()	0x64	Read word	Shows the error state of the PMM35.
InputCurrent()	0x65	Read word	Monitors current through the input connector, as it comes from the external power supply.
Reset()	0x66	Write word	Resets the PMM35.
Low-Power-Consumption-Mode()	0x67	Write word	Enables the Low-Power-Consumption-Mode

CustomRechargeThreshold()	0x6B	R/W word	Controls when the battery will be recharged after self-discharge.
PowerPathControl()	0x70	R/W word	Disconnects the external power supply.

Table 10 – Extended registers.

CustomChargingCurrent() 0x60

The CustomChargingCurrent() acts as a limit: The battery will be charged with the smaller value between CustomChargingCurrent() and ChargingCurrent() as requested by the battery. Use it to charge the battery at a lower rate, which will produce less heat in the battery and potentially prolonging its life expectancy.

Access	Read and Write
Data format	Unsigned Integer
Unit	mA
Range	0…65,534 mA
Maximum accepted value	6,000 mA
Stored in Flash	Yes
Factory Default	0, meaning: Feature is disabled

Note that the CustomChargingCurrent() has a maximum value of 6.0 A. Once configured, the custom charging current limit is immediately valid, and it is stored in flash memory. Therefore, it is automatically set at the next restart.

To turn off the CustomChargingCurrent() feature, set its value to 0. If you want to completely stop the charging process, use the Inhibit Charge function in ChargerMode() instead.

The PMM35 can only adjust the current limit in steps of 48 mA, and there may be component tolerances (which are accounted for as the PMM35 has been calibrated during production). As a result, the actual charging current limit may not correspond precisely to the set-value, and the actual current can be up to 47 mA lower. However, the PMM35 is programmed to prevent the charging current from exceeding the set value.[3] It is also important to know that the smallest custom charging current limit that you can set is 612 mA. If you want to determine the actual charging current limit, you can read it back.

After writing a new value to this register you need to wait 20 ms before sending the next SMBus command to the PMM35, since it needs some time to store the new value in its flash memory.

CustomChargingLimit() 0x61

With the CustomChargingLimit() you can determine how much the battery will be charged. You can either set an SOC limit or a charging voltage limit. Charging the battery not completely will potentially have a great beneficial effect on battery life because it relieves the lithium-ion cells from voltage-related stress. For more details, read the corresponding Application Example, section 6.7.1 on page 28.

If you set a value from 1 to 100, the value will be interpreted as an SOC. The battery will be charged until its RSOC (register 0x0D) reaches this value.

[3] For set points below 612 mA, there is an exception to this rule: they will always result in a current of 612 mA ± tolerances, which is between 243 and 269 mA. For example, if you want to limit the charging current to 100 mA, the actual current will be set to 612 mA because this is the smallest current that can be set.

If you set a value from 101 to 65535, the value will be interpreted as a charging voltage. The battery will be charged with this voltage until a taper-current condition is met. If a value above the battery's requested charging voltage is set, the CustomChargingLimit will be ignored. The PMM35 will ensure that the battery's charging voltage won't be exceeded.

Access	Read and Write
Data format	Unsigned Integer
Unit	% SOC or mV
Range	0…100% or 101…65535 mV
Step size	1% or 20 mV
Maximum accepted value	34,800 mV
Stored in Flash	Yes
Factory Default	0, meaning: Feature is disabled

Note that the CustomChargingLimit() has a maximum value of 34.8 V and can be configured in steps of 20 mV. Once configured, the custom charging voltage limit is immediately valid, and it is stored in flash memory. Therefore, it is automatically set at the next restart.

To turn off the CustomChargingLimit() feature, set its value to 0. If you want to completely stop the charging process, use the Inhibit Charge function in ChargerMode() instead.

To control when the battery will be recharged after it self-discharged, see command CustomRechargeThreshold 0x6B.

After writing a new value to this register you need to wait 20 ms before sending the next SMBus command to the PMM35, since it needs some time to store the new value in its flash memory.

PowerLossDetection() 0x62

The PowerLossDetection() function is used to detect if there is an oxidized battery connector that could cause increased heat at the connection point.

When PowerLossDetection() is turned on, the PMM35 reads the battery to determine the charging power that reaches the battery and compares it to the power output of the PMM35. If the difference between the two exceeds 6 watts, the PMM35 stops charging, assuming that the battery connector is oxidized.

By default, this function is turned on. However, if you are using a longer cable between the PMM35 and the battery, you may want to disable this function due to power loss in the cable.

Access	Read and Write
Data format	Unsigned Integer
Possible values	1: Power loss detection enabled 0: Power loss detection disabled (default)
Stored in Flash	Yes

Once configured, the setting is immediately valid, and it is stored in flash memory. Therefore, it is automatically set at the next restart.

After writing a new value to this register you need to wait 20 ms before sending the next SMBus command to the PMM35, since it needs some time to store the new value in its flash memory.

EnablePullups() 0x63

EnablePullups() controls the pull-up resistors required for SMBus communication. SMBus requires pull-up resistors on its data and clock lines. The PMM35 has on-board pull-up resistors with a value of 10 kΩ, which by default are enabled. These pull-up resistors are only available while the PMM35 is powered externally.

However, when the battery powers the application, these pull-ups are not available, so the application needs to provide their own. In this case, you should disable the PMM35's pull-up resistors through EnablePullups(). The same applies if your host system is a pre-manufactured board that provides pull-up resistors anyway. Or if your system is a 5 V system, in which case you should disable the on-board resistors and instead use 15 kΩ pull-up resistors connected to the 5 V supply.

Access	Read and Write
Data format	Unsigned Integer
Possible values	1: Pull-up resistors enabled (default) 0: Pull-up resistors disabled
Stored in Flash	Yes

Once configured, the setting is immediately valid, and it is stored in flash memory. Therefore, it is automatically set at the next restart.

After writing a new value to this register you need to wait 20 ms before sending the next SMBus command to the PMM35, since it needs some time to store the new value in its flash memory.

Error() 0x64

Error() shows the error state of the PMM35.

Bit	Name	Description
13…15	Reserved	–
12	Flash Write Error	0: Writing last setting to flash memory successful. 1: Writing last setting to flash memory failed.
11	Faulty Current Measurement	0: The current read back from the battery corresponds to the charging current. 1: The current measured in the battery exceeds the intended charging current by more than 1 A. This error indicates a faulty current measurement.
10	Communica-tion Failure	0: The communication with the battery works as intended. 1: The communication with the battery failed. Charging is inhibited. Re-try every 5 s.
9	Wake-up Timeout	0: Battery awake, or no battery attached. 1: Battery has not woken up from shipping mode or from deep discharge within 210 s.
8	Charging Timeout	0: No charging timeout. 1: Timeout inhibits charging (happens after 24 h of continuous charging without reaching FullyCharged state)
6, 7	Corrupted Flash	0: The flash memory is working as intended 1: The flash memory seems to be corrupted. If this error does not clear after reboot, the PMM35 must be replaced.
5	Charger Too Hot	0: Charger hotspot temperature < 115°C 1: Charger hotspot temperature ≥ 115°C, charging is interrupted.
4	Charging Power Reduced	0: PMM35 can charge at full rate. 1: Charger hotspot temperature ≥ 75°C, leads to a reduced charging rate.

3	Charging Parameters Rejected	0: Charging parameters correctly applied. 1: Charging parameters rejected, either because of internal communication failure, or because of a faulty charging IC.
2	High Power Loss	0: Power loss between PMM35 output and battery input < 3 W. 1: Power loss between PMM35 output and battery input ≥ 3 W.
1	No Charge Current Charger	0: The measured charging current corresponds to the set value. 1: There is no charging current measured by the charger even though the PMM35 should be charging.
0	No Charge Current Battery	0: The measured charging current corresponds to the set value. 1: There is no charging current measured by the battery even though the PMM35 should be charging.

Table 11 – Error() flags.

InputCurrent() 0x65

The InputCurrent() function returns the current measured at the input connector of PMM35 PCB, which is supplied by an external power supply.

Access	Read
Data format	Unsigned Integer
Unit	mA
Range	0…65,535 mA

Reset() 0x66

This function reboots the PMM35 microcontroller on sending the command 0x4D53.

Access	Write
Data format	Unsigned Integer
Possible values	0x4D53

As a consequence, the PMM35 stops charging, resets the microcontroller and thereby reinitializes by reading the settings out of Flash memory. Meanwhile, the host stays powered.

LowPowerConsumptionMode() 0x67

This function puts the PMM35 in a low power consumption mode.

Access	Write
Data format	-
Possible values	any

In LowPowerConsumptionMode() the microcontroller of the PMM35 is switched off. The output of the PMM35 is supplied with the battery voltage, over a Low-Power-Path. The current of this Low-Power-Path is limited to approx. 20 mA. If the output load draws more than the 20 mA the PMM35 switches back to normal operation.

CustomRechargeThreshold() 0x6B

The PMM35 starts recharging the battery if the voltage or SOC falls below the set value.

Access	Read and Write
Data format	Unsigned Integer
Unit	% SOC or mV
Range	0…100% or 101…65535 mV
Step size	1% or 20 mV
Maximum accepted value	34,800 mV
Stored in Flash	Yes
Factory Default	0, meaning: Feature is disabled

Note that the CustomRechargeThreshold() has a maximum value of 34.8 V and can be configured in steps of 20 mV. Once configured, the custom recharge threshold is immediately valid, and it is stored in flash memory. Therefore, it is automatically set at the next restart.

After writing a new value to this register you need to wait 20 ms before sending the next SMBus command to the PMM35, since it needs some time to store the new value in its flash memory.

PowerPathControl() 0x70

This function allows you to disconnect the external power supply, causing the PMM35 to stop charging and supplying power to the application. Instead, power will be provided by the battery if one is connected. Be aware that using this function can result in your application shutting down unexpectedly due to the battery running out of power, even if an external power supply is connected. Check the Application Example section for examples on how to use this function.

Access	Read and Write
Data format	Unsigned Integer
Possible values	0x64F8: Power supply disconnected Any other value: Power supply connected (default) Reading back results in 0x0000.

6. Application Examples

This section answers some commonly asked questions that can help you optimize your system's configuration.

Throughout this section, you will find communication examples like the one below, where we set the CustomChargingCurrent() to 2000 mA:

S	PMM35 Address	Wr	A	Command Code	A	Data Byte Low	A	Data Byte High	A	P
S	0110 011	0	A	0110 0000	A	0001 1001	A	0000 0111	A	P
S	0x66		A	0x60	A	0xd0	A	0x07	A	P
	PMM35 Address for Extended Registers			Custom Charging Current()		LSB from 2000 = 0x07d0		MSB from 2000 = 0x07d0		

In these examples, red values are shown in binary, and blue values are in hexadecimal form. A "non-shaded" block indicates a message sent by the host, while a "shaded" block represents a message returned by the PMM35. According to the [SMBS][4], the single bits outside of the data bytes indicate the following operations:

S	A	/A	Wr	Rd	P
1	2	3	4	5	6

1: Start; 2: Acknowledge (ACK); 3: Not Acknowledge (NACK); 4: Write bit; 5; Read bit; 6: Stop

6.1. Start & Stop Charging on Command

To stop the charging process at any time, use the "Inhibit Charging" command set by ChargerMode().

S	PMM35 Address	Wr	A	Command Code	A	Data Byte Low	A	Data Byte High	A	P
S	0001 001	0	A	0001 0010	A	0000 00x1	A	0000 0000	A	P
S	0x12		A	0x12	A	0x01 for level 2 charger 0x03 for level 3 charger	A	0x00	A	P
	PMM35 Address			ChargerMode()		LSB		MSB		

"Inhibit Charging" will not be stored in flash. On charger reset, or on plugging in a new battery, "inhibit charging" will be reset, allowing the PMM35 to charge the battery.

Resume the charging process any time by writing 0x00 (or 0x02 for a level 3 charger) to ChargerMode().

6.2. Put the battery into shipping mode (device-shutdown, long-term storage and shipping)

When a battery is in Shipping Mode, its fuel gauge electronics are shut down, and it won't show any voltage at the terminals. This is done to minimize self-discharge, making it ideal for long-term storage or shipping the battery.

We also encourage you to put a battery into shipping mode when it reaches a specific state of charge, such as 5% or 10%, after providing a warning to the user. Instead of allowing the battery to completely run out of power, this approach helps prevent deep discharge of the battery when the device is turned off and can help extend the battery's lifespan.

If the PMM35 is powered by an external power supply, you must first stop charging, because charging would wake the battery from Shipping Mode immediately. Use the "Inhibit Charge" command in Charger-Mode() as described in the section above.

[4] SMBus specification rev. 1.1, which you can freely download at http://smbus.org/specs/smbus110.pdf

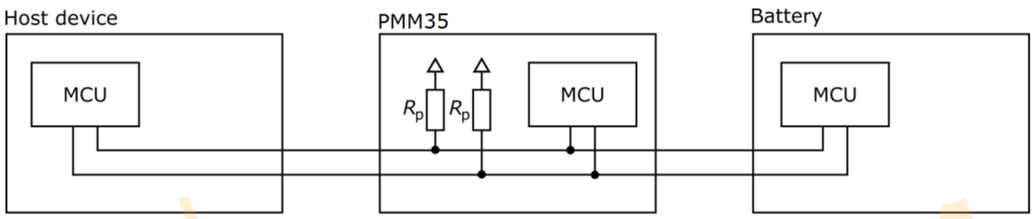
Next, send the shipping mode command to the battery. Ensure to send it twice within 4 seconds, without interrupting it by other SMBus commands:

S	PMM35 Address	Wr	A	Command Code	A	Data Byte Low	A	Data Byte High	A	P
S	0001 011	0	A	0000 0000	A	0001 0000	A	0000 0000	A	P
S	0x16		A	0x00	A	0x10	A	0x00	A	P
	Smart Battery Address			Manufacturer Access()		Shipping Mode Command LSB		Shipping Mode Command MSB		

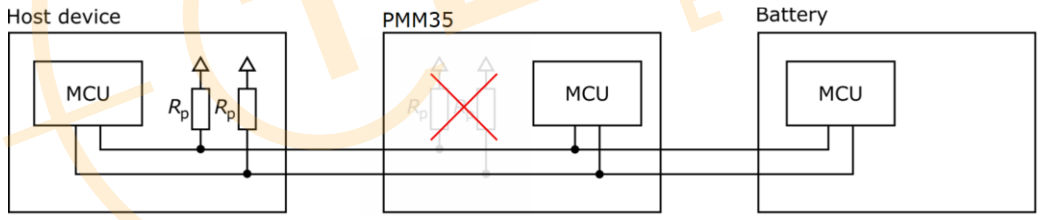
The battery should enter shipping mode within a couple of seconds, which can be seen by the fact that the voltage at its terminals turns off and the battery won't react to further SMBus commands.

6.3. Use your own pull-up resistors

The PMM35 offers the pull-up resistors needed for SMBus communication. By default, they are turned on so that the PMM35 can charge a smart battery autonomously. However, these pull-up resistors are only available while the PMM35 is powered externally.



When the battery powers the application, the pull-ups are not available, so the application needs to provide their own. Or, if your device already offers pull-up resistors which can not be removed (which is often the case for single board computers), then you can easily turn off the pull-up resistors on the PMM35 board. You also need to do this if your device runs on 5 V logic, in which case you want to use your own pull-up resistors, connected to the 5 V rail.[5]



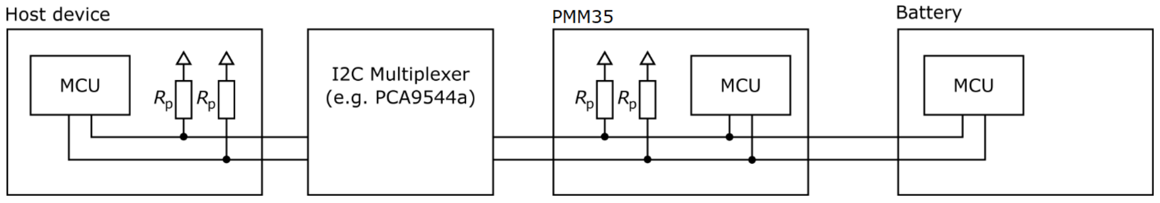
To turn off the pull-up resistor, use the EnablePullups() function and write "0x0000". The PMM35 will remember this setting even after a reset, so it will start up with the pull-up resistors disabled. Use this command with caution because you can only communicate if you have some pull-up resistors.

S	PMM35 Address	Wr	A	Command Code	A	Data Byte Low	A	Data Byte High	A	P
S	0110 011	0	A	0110 0000	A	0000 0000	A	0000 0000	A	P
S	0x66		A	0x63	A	0x00	A	0x00	A	P
	PMM35 Address for Extended Registers			Enable Pullups()		LSB		MSB		

Note that most single board computers offer support for SMBus 2.0 and higher, while the PMM35 and RRC batteries use the more energy-saving SMBus 1.1 standard. If your pull-up-resistors have a resistance lower than 6.8 kΩ, the battery might not be able to bring the signal to a low logic level. If you cannot unsolder your

[5] Read the respective chapter in the "RRC35xx Design-In Guidelines" for recommendation on the pull-up resistor values.

pull-up resistors, you will need to use a multiplexer to separate the PMM35 and your device. In this scenario, you can keep the pull-up resistors on the PMM35 turned on.



6.4. Charging two or more batteries

6.4.1. Parallel configuration

You can use any number of PMM35 connected in parallel. Figure shows how:

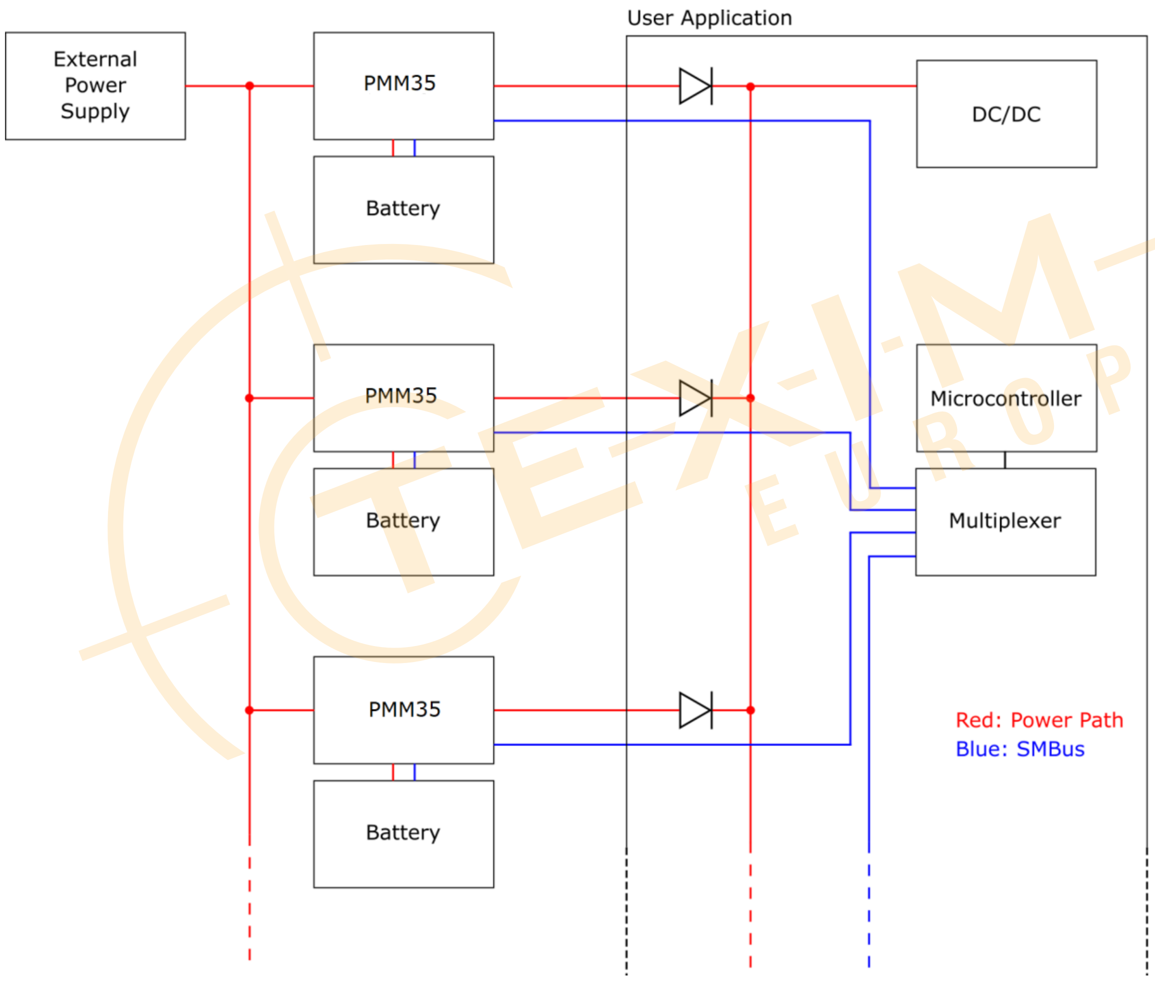


Figure 6 - Using PMM35 in a parallel configuration. The number of PMM35 is not limited.

- Use a dedicated PMM35 for each battery.
- For each PMM35, make sure to set an appropriate InputCurrentLimit(). If you are using multiple PMM35 units with a single power supply, ensure that the power supply can deliver the combined input currents, which may be a multiple of the configured limits.
- To avoid back-driving currents, the use of ideal diodes (e.g., LM5050 from TI) is recommended.
- Use a multiplexer (e.g., PCA9544a).
  This is necessary as per SMBus specification: All smart batteries share the same SMBus address.
- Take care of inrush currents, as they might not distribute equally among all PMM35's.

The PMM35 input transistors switch through after 4 s (typically). Every PMM35 takes a slightly different time to start up. This results in having your application's inrush current run through exactly one PMM35 – this current will not split up. If your inrush current is high enough to blow a fuse on a PMM35, consider putting one diode in parallel to all PMM35. It can be a relatively small diode, but its single pulse capability must match the expected inrush current.

Further requirements are:

- To have balanced discharge rates between all batteries, ensure that their states of charge are approximately equal.
- Do not use batteries with considerably different states of health.
- Do not use batteries with different lithium-ion cells, which can happen if you use different battery generations.

Why is it important to have equally charged batteries?

Think of a scenario where you have 8 batteries connected in parallel. One of the batteries is fully charged, while the other 7 are almost empty. In this situation, the fully charged battery would carry the entire load until its voltage drops to the level of the other 7 batteries. Once that happens, all 8 batteries would share the current.

It is essential that the fully charged battery does not fail while supporting the entire load by itself.

At first, determine whether the most extreme case would overload the 1 battery. If not, then there is no need for additional circuitry. If yes, consider using current limiting circuitry like the MAX17526 or similar.

Depending on your application, you could also consider a software solution. You could read the state of charge and voltage of each battery. If the batteries are not balanced, you could show a message like "change battery xy" and wait until the task is completed before consuming more current.

6.4.2. Series configuration

While possible, we don't recommend using smart batteries in series because of increased circuit complexity. If you want to know more about the pros and cons of a series connection and how to design the circuit, refer to our application note "RRC3570 Design-in Guidelines".

6.5. Determine whether a power supply voltage is connected

Sometimes, you want to know if an input power supply is attached. You can use the InputVoltage() command to retrieve this information. In the following example, we request the input voltage at the PMM35 connector, and the PMM35 returns 18.75 V (18750 mV = 0x493e).

S	PMM35 Address	Wr	A	Command Code	A	S	PMM35 Address	Rd	A	Data Byte Low	A	Data Byte High	A	P
S	0001 001	0	A	0011 1110	A	S	0001 001	1	A	0011 1110	A	0100 1001	/A	P
S	0x12		A	0x3e	A	S	0x13		A	0x3e	A	0x49	/A	P
	PMM35 Write Standard Registers			InputVoltage()			PMM35 Read Standard Registers			LSB		MSB		

6.6. Monitor the power supply current and power

Besides the input voltage, you can also read the input current as delivered from an external power supply. In the following example, we request the input current, and the PMM35 returns 2.35 A (2350 mA = 0x092e).

S	PMM35 Address	Wr	A	Command Code	A	S	PMM35 Address	Rd	A	Data Byte Low	A	Data Byte High	A	P
S	0110 011	0	A	0110 0101	A	S	0110 011	1	A	0010 1110	A	0000 1001	/A	P
S	0x66		A	0x65	A	S	0x67		A	0x2e	A	0x09	/A	P
	PMM35 Address for Extended Registers			InputCurrent()			PMM35 Read Extended Registers			LSB		MSB		

To find out the power delivered by the external power supply, you can read both input voltage and input current and multiply both results.

6.7. Charging the battery gently (life-prolonging)

Battery health is just like taking care of a living organism - it requires some attention to age well. Aging translates to irreversible capacity loss, until the battery cannot meet the runtime requirements of your device anymore. However, if the individual battery cells age at different rates very quickly, it can also result in a voltage imbalance that cannot be corrected, which may lead to immediate battery failure.

The critical elements to maintaining good battery health are, in the order of importance:

- Temperature
- State of Charge
- Current

With the PMM35, you can control the charging parameters and thus, contribute to a long battery life.

6.7.1. Limit the charging voltage

If the device runtime you are aiming for can be reached with less than 100% of the battery capacity, and especially if you have a UPS (uninterruptable power supply) application, it may be a good idea to limit the charging voltage. For a UPS application, please also read section 6.8.

Equally, it is also recommended not to discharge the battery completely. For the lifetime of the battery it is better to turn off your system while the battery charge is above 0% and put the battery into shipping mode, which is described in section 6.2. This would protect the battery from deep discharge.

The reason is: Scientific studies have shown that Lithium-ion batteries degrade faster when they are at extreme states of charge, such as above 90% or below 10%. However, cycling a battery around 50% can greatly contribute to the battery's lifespan. For example, if you use the battery between 75% and 25% (thus, halving the depth-of-discharge), you could double the battery's lifespan as compared to using it between 100% and 0%, without changing any other conditions. Therefore, it is beneficial to use the battery at medium charge levels. Additionally, Li-Ion batteries do not suffer from the memory effect, so it is safe to charge your device more often.

Use the CustomChargingLimit() command to limit the voltage to the desired level. It is suggested that you use a charging voltage of at least 4.0 V per cell, such as 20 V for a 5S battery or 32 V for an 7S battery. This is because the battery requires a minimum charging voltage to balance the battery cells and determine the full charge capacity. As a rule of thumb, a 70 mV change corresponds to change of around 10% in state-of-charge (SOC):

Voltage per cell	Absolute SOC
4.20 V	100%
4.13 V	90%
4.10 V	86%
4.06 V	80%
4.00 V	71%

Please note that the correlation between 70 mV and 10% SOC is not always accurate as it is not a linear relationship and is specific to the type of lithium-ion cell being used. Moreover, this relationship may become obsolete over time as new cell generations are introduced. Once you have finished charging, you can always check the true SOC with the AbsoluteStateOfCharge() function and change the charging voltage accordingly.

As an example, suppose you have an RRC3570 battery and you want to restrict the maximum charge to 80% of the absolute SOC. This battery is a 7S battery, and assuming that 4.06 V per cell is required for 80%, let's limit the charging voltage to 7 x 4.06 = 28.42 V or 28,420 mV:

S	PMM35 Address	Wr	A	Command Code	A	Data Byte Low	A	Data Byte High	A	P
S	0110 011	0	A	0110 0001	A	0000 0100	A	0110 1111	A	P
S	0x66		A	0x61	A	0x04	A	0x6f	A	P
	PMM35 Address for Extended Registers			Custom Charging Voltage()		LSB from 28420 = 0x6f04		MSB from 28420 = 0x6f04		

6.7.2. Reduce the charging current

Due to their electrochemical impedance, the Lithium-ion cells heat up in direct correlation to the current amplitude. And because chemical decay processes happen faster at higher temperatures, that could lead to increased aging. The cell components may also saturate while charging. Therefore, if you do not need to charge your battery quickly, it is recommended to charge it at a slower rate.

For example, if you want to limit charging current limit to 2000 mA, you can do so by writing this value to the CustomChargingCurrent() function.

S	PMM35 Address	Wr	A	Command Code	A	Data Byte Low	A	Data Byte High	A	P
S	0110 011	0	A	0110 0000	A	0111 0000	A	0000 0111	A	P
S	0x66		A	0x60	A	0xd0	A	0x07	A	P
	PMM35 Address for Extended Registers			Custom Charging Current()		LSB from 2000 = 0x07d0		MSB from 2000 = 0x07d0		

As explained in section 5.2, the adopted current limit may not be precisely 2000 mA due to the tolerances of the components and the granularity of the system. If you want to determine the actual charging current limit, you can read it back:

S	PMM35 Address	Wr	A	Command Code	A	S	PMM35 Address	Rd	A	Data Byte Low	A	Data Byte High	A	P
S	0110 011	0	A	0110 0000	A	S	0110 011	1	A	1000 0111	A	0000 0111	/A	P
S	0x66		A	0x60	A	S	0x67		A	0x87	A	0x07	/A	P
	PMM35 Address for Extended Registers			Custom Charging Current()			PMM35 Address for Extended Registers			LSB		MSB		

For example, if you read a value of 0x0787, it means that the CustomChargingCurrent() is set to 2016 mA. This is because the PMM35 can only adjust the current in increments of 48 mA, and the system takes into account component tolerances during production. So, the PMM35 sets the closest possible value to 2000 mA, which in this case is 2016 mA.

Please note the following:

- The charging current will always be limited to the lower value between the value set in CustomChargingCurrent() and the current requested by the battery.
- You can disable the charging current limitation by setting the CustomChargingCurrent() value to “0”.

6.8. Using the Battery as Backup Power Source (UPS)

If you plan to use the battery as a backup power source and have the PMM35 always connected to an external power supply, there are important things to keep in mind:

Battery aging

As explained in more detail in section 6.7.1, keeping a battery charged above 90% all the time can lead to faster aging. To prevent this, follow these steps:

1. Choose a battery, or a combination of multiple batteries in parallel, that can provide the required runtime with only 80% or less of the absolute state of charge. For guidance, refer to the relevant chapter in the "RRC3xxx Design-In Guidelines” document.
2. Lower the charging voltage to a level between 4.0 and 4.1 V per cell. See section 6.7.1 for more information.
3. Check the absolute state of charge after charging and adjust the charging voltage if necessary.

CTO timer

The battery has a timer called “Charge Timeout” (CTO) to protect itself from overcharging. When the timer reaches a total charge time of 18 hours, the battery stops charging until it is discharged. Charging times add up, and the timer only resets

- If you discharge the battery by at least 100 mA for at least 1 second,
- Or if you temporarily put the battery into shipping mode.

When using the battery as a backup power source, it is unlikely that the battery will discharge. However, due to self-discharging and short charge cycles, the battery may enter the Charge Timeout at some point during the device's lifespan (usually after 2 to 5 years). To prevent this issue, you have two options:

Option 1: You can briefly discharge the battery into the device by turning off the power path using the function PowerSupply() (as described in section 5.2):

S	PMM35 Address	Wr	A	Command Code	A	Data Byte Low	A	Data Byte High	A	P
S	0110 011	0	A	0111 0000	A	1111 1000	A	0110 0100	A	P
S	0x66		A	0x70	A	0xf8	A	0x64	A	P
	PMM35 Address for Extended Registers			PowerSupply()		LSB of 0x64f8		MSB of 0x64f8		

After a few seconds (for example, 20 seconds if the discharge power is lower than 320 mA), turn on the power path again by sending the appropriate command to the PowerSupply() function.

S	PMM35 Address	Wr	A	Command Code	A	Data Byte Low	A	Data Byte High	A	P
S	0110 011	0	A	0111 0000	A	0000 0000	A	0000 0000	A	P
S	0x66		A	0x70	A	0x00	A	0x00	A	P
	PMM35 Address for Extended Registers			PowerSupply()		LSB of 0x0000		MSB of 0x0000		

You can do this after each charging cycle or every few months while the device is connected to AC mains. If you are using multiple PMM35/battery combinations in parallel, you must use this command with all PMM35 simultaneously to ensure none of them continues to power the application via AC mains. Note that if you use a bypass diode across those PMM35's to cope with the inrush current, this method will not work.

Option 2: You can reset the timer by putting the battery into shipping mode according to the instructions in section 6.2. A couple of seconds later, wake the battery by resuming the charging process through "Inhibit Charging" in ChargerMode():

S	PMM35 Address	Wr	A	Command Code	A	Data Byte Low	A	Data Byte High	A	P
S	0001 001	0	A	0001 0010	A	0000 00x0	A	0000 0000	A	P
S	0x12		A	0x12	A	0x00 for level 2 charger 0x02 for level 3 charger	A	0x00	A	P
	PMM35 Address			ChargerMode()		LSB		MSB		

You can do this after each charging cycle or every few months while the device is connected to AC mains. However, there is a risk that your device may request backup power from your battery while it is in shipping mode, causing your device to shut down unexpectedly. If you are using multiple PMM35/battery combinations in parallel, you can use this command with all batteries consecutively so that only one battery is in shipping mode at a time, with the other batteries able to provide power to the load.

To cover all bases, note that it is also possible to delay the occurrence of this issue. The final few percent of charging take the longest time to complete. So instead of constantly charging between 95% and 100%, you can allow the battery's state of charge to drop to the lowest level possible before recharging it again. This way, you can extend the time before the battery reaches the Charge Timeout.

6.9. Inrush Current

To protect the PMM35, it is equipped with fast-acting 25 A fuses. However, if your application has a total input capacitance exceeding 1000 µF, we recommend using an inrush current limiter to prevent the fuses from acting. If you encounter issues with high inrush currents, consider using a current-limiting circuit such as the one presented below.

Easy start-up current limiting circuit

The easy start-up current limiting circuit involves using a resistor in series with your circuit and a switch (preferably a FET transistor) in parallel to that resistor.

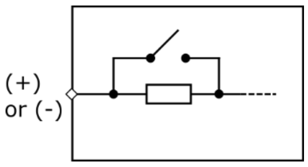


Figure 7 – Easy current limiting circuit using a switch and a resistor.

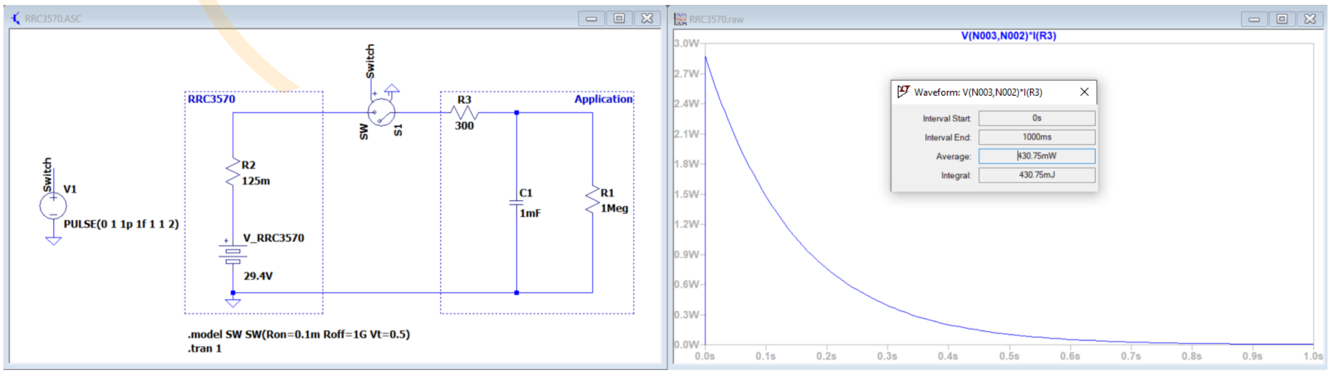
During start-up, the FET shall be open, and all the current passes through the resistor, slowing down the inrush current. After a specific time, or after reaching a defined voltage level, either an MCU or an analog circuit (slowed down by an RC filter) can close the FET, thus permitting full load current. You can use an NFET in the GND line or a PFET in the (+) path.

When selecting the resistor, ensure it can handle the impulse load. Let's make an example:

Example on choosing a suitable resistor

In the LTSpice simulation below, we see a simple model of the battery on the left-hand side, the application on the right-hand side, and a switch in-between. The switch S1 and the resistor R1 serve only for simulation purposes – they are needed so that the capacitor C1 is empty at the beginning of the simulation.

The example application incorporates a big 1 mF input capacitor. We want to limit the inrush current with a series resistor of 300 Ω. Earlier, we spoke about a FET parallel to that resistor – in the simulation, we omit that FET for clarity.



```
.model SW SW(Ron=0.1m Roff=1G Vt=0.5)
.tran 1
```

Waveform: V(N003,N002)*I(R3)

Interval Start:	0s
Interval End:	1000ms
Average:	430.75mW
Integral:	430.75mJ

Figure 8 – Inrush current simulation.

As you can see in the simulation result, the capacitor needs approximately 1000 ms until it is fully charged. The peak power dissipated across the resistor is 2.8 W, and the total energy loss is 430 mJ.

Resistor datasheets should provide information on the permitted single pulse power. To convert the simulation results into a single pulse load, you need to

- Take the simulated peak power as pulse load
- Calculate the single pulse duration by t = E / P

In this example, the pulse load is 2.8 W, and the duration is 430 mJ / 2.8 W = 154 ms. Compared to the datasheet of an arbitrary manufacturer (refer to Figure), you see that a case size of 1210 would be necessary to ensure the power capability. You could still optimize the schematic by playing with the resistance or looking for other manufacturers and case types.

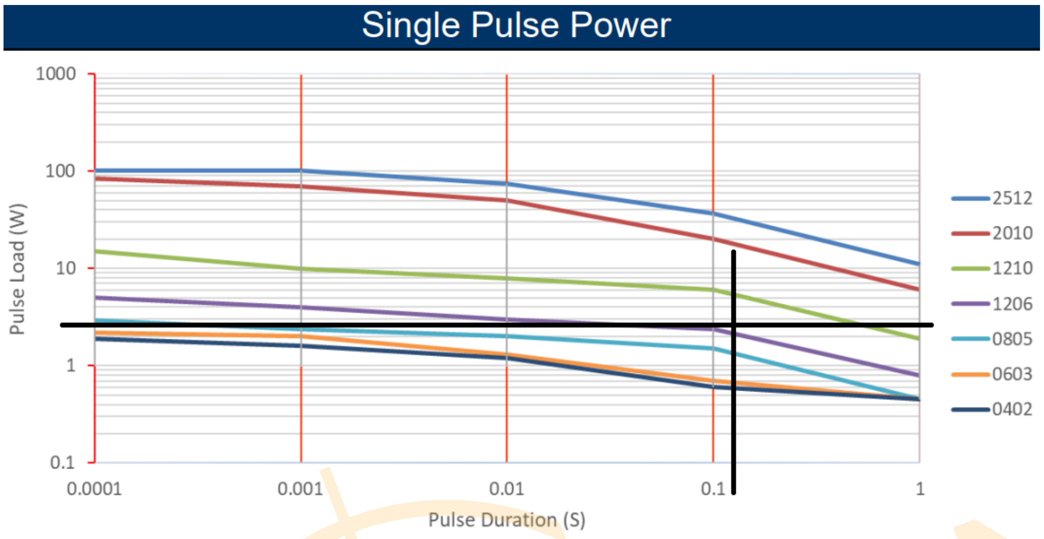


Figure 9 – Datasheet from an arbitrary manufacturer.
The calculated single pulse load requirement is drawn into the graph using black lines.

7. EMC

The PMM35 power management module was pre-compliant tested as a component according to the following EMC Standards:

- EN 60601-1-2:2007 + Corr.1:2010
- EN 55011:2009 + A1:2010

The final EMC compliance is dependent on the user application and power supply.

8. Safety

The PMM35 complies with

- CB acc. IEC62368-1
- UR acc. UL60601-1

9. Regulatory Compliance / Certifications

The PMM35 meets all necessary standards and regulations for the countries listed below, including those related to safety, electromagnetic compatibility, environmental protection, and recycling.

The PMM35 also complies with

- RoHS Directive 2011/65/EU + 2015/863 (RoHS 3)
- REACH Directive 1907/2006/EC

Region	Certificate	Marking
International	CB	-
Europe	CE	CE
United Kingdom	UKCA	UK CA
USA / Canada	c UR us	c UR us ®

10. Mechanical Information

10.1. Dimensions

The mechanical dimensions in this document follow the ISO2768-fK standard for tolerances.

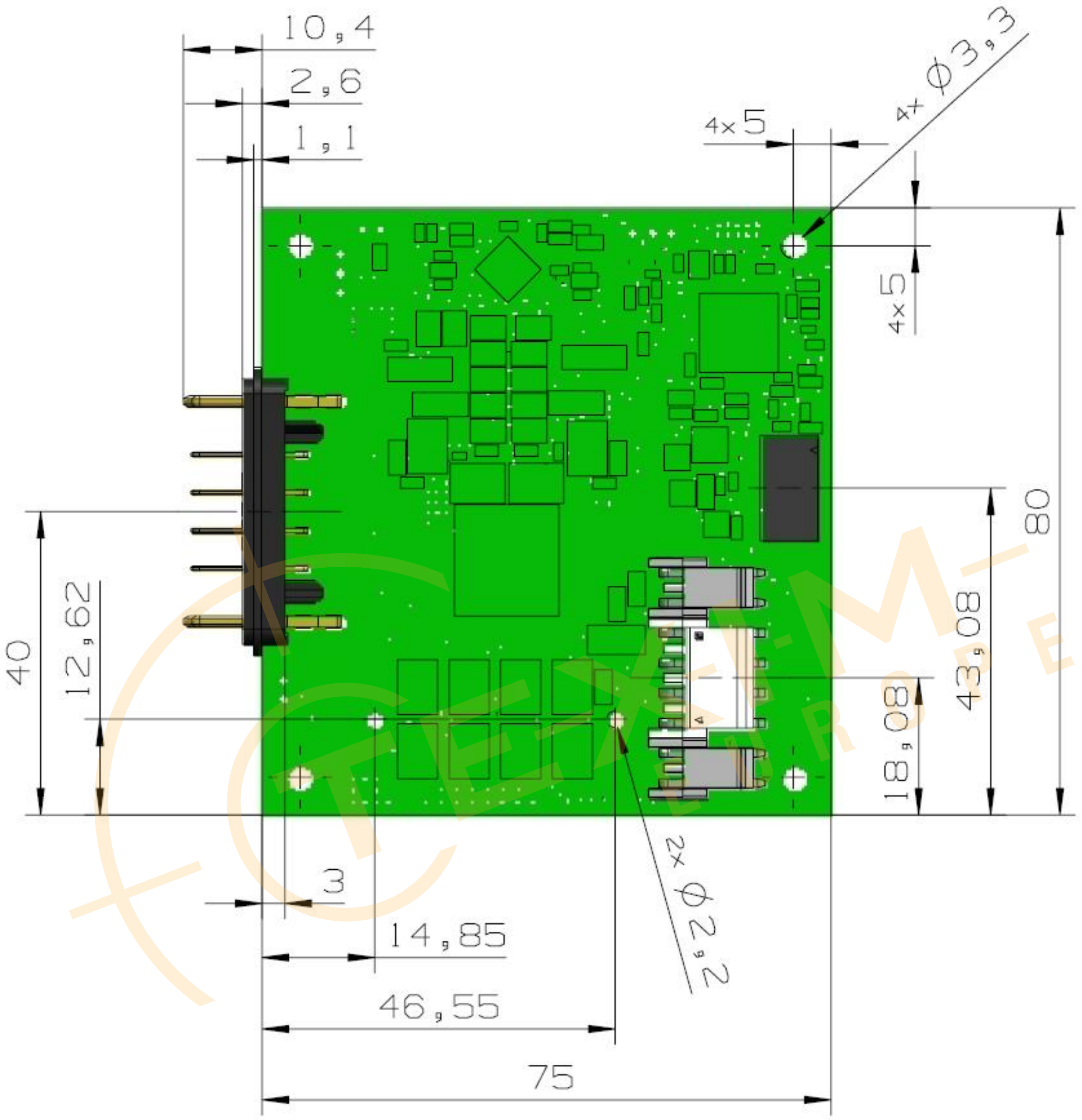


Figure 10 – Mechanical dimensions, as seen from the top side.

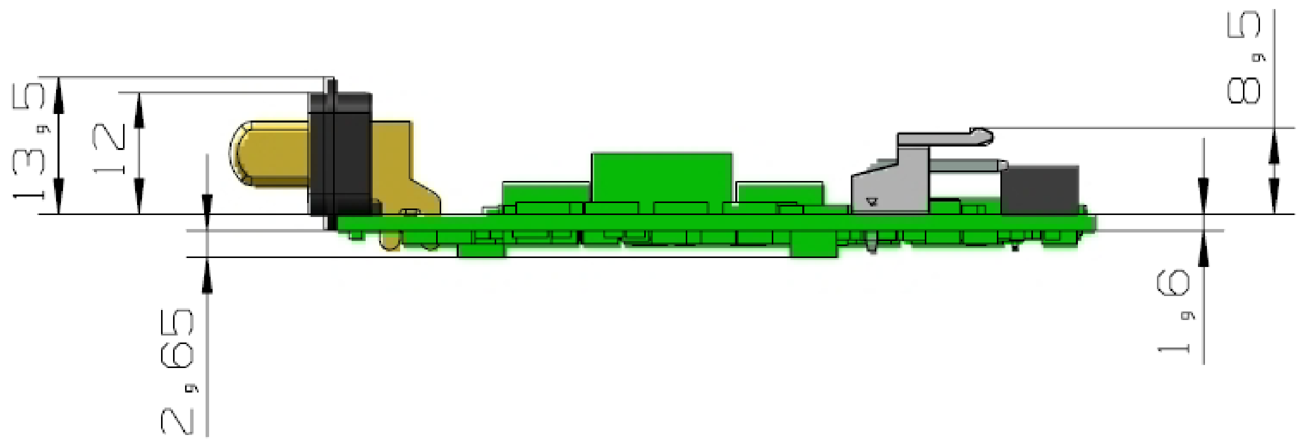


Figure 11 – Mechanical dimensions, side view.

10.2. Mounting screws

When attaching the PMM35 to the mounting holes, the maximum diameter of the washer or screw head should not exceed 5.5 mm.

The outer 2-pin connection plugs are required if the application is operated with more than 160W. Otherwise, it is sufficient to only use the inner 4-pin connection.

If the application is operated with more power and the outer 2-pin connection plugs are therefore required, we recommend using the covered hole as a collecting hole.
The following image shows an example of how the PMM35 can be attached/supported at this point without screws.

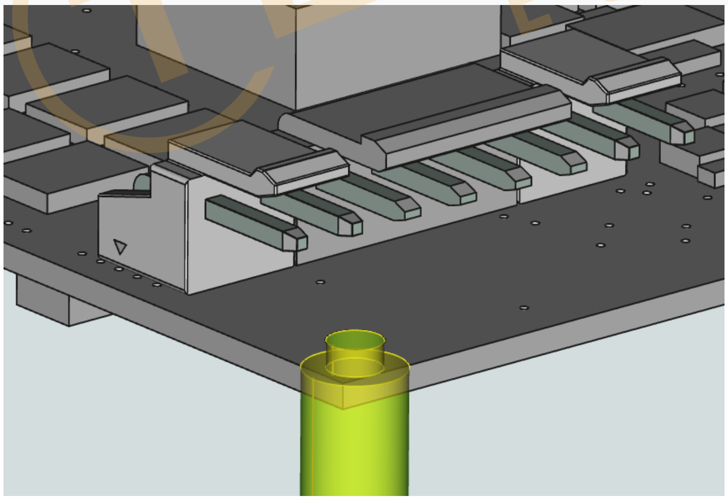


Figure 12 – Possible design for a collecting hole support structure.

10.3. Battery insertion

Figure shows the correct insertion direction for an RRC3570 battery.

Note that there are small "VBat+" and "GND" labels on the PMM35 PCB next to the connector that correspond to the location of the "+" and "-" symbols on the battery.

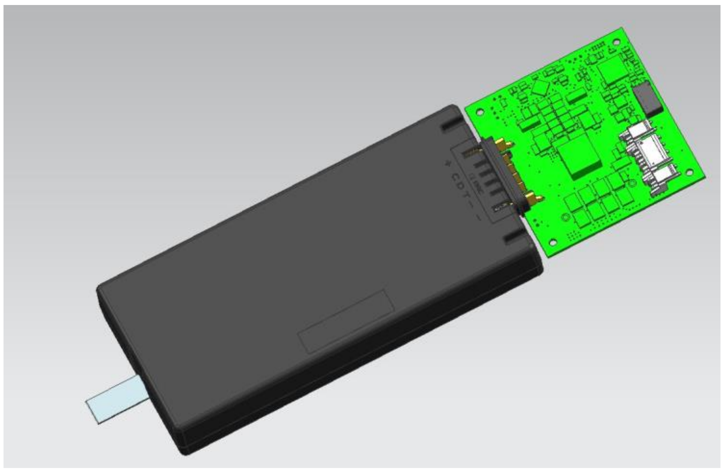


Figure 13 - Correct direction of connection

11. Documentation Support

For related documentation, see the following:

- System Management Bus Specification (Rev 1.1, Dec 11, 1998)
- Smart Battery Data Specification (Rev 1.1, Dec 15, 1998)
- Smart Battery Charger Specification (Rev 1.1, December 1, 1998)

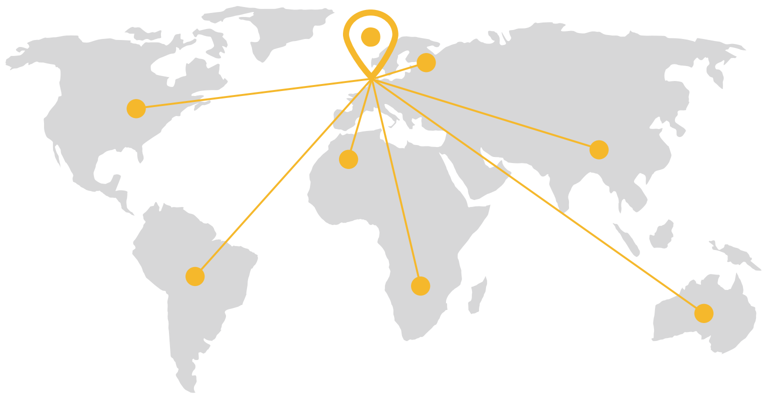
If you cannot follow any of these links, or if they do not work anymore, please contact your local RRC sales-person.

12. Revision History

Revision	Valid from	Revisions	Author
B	03Feb2025	Release for series production	Andreas Warms-bach

Germany/HQ	USA	Hong Kong	China	Vietnam
RRC power solutions GmbH	RRC power solutions Inc.	RRC power solutions Ltd.	RRC power solutions Ltd.	RRC Power Solutions Company Ltd.
Technologiepark 1 66424 Homburg / Saar	18340 Yorba Linda Blvd., # 107-437 Yorba Linda, CA 92886	S-V,6/F, Valiant Industrial Centre 2-12 Au Pui Wan Street Fo Tan, N.T., Hong Kong	Room 1306, C Building, Tianan International Build-ing, Renmin South Road, Luohu District, Shenzhen 518021	Block C, Lot CN4 Dinh Vu – Cat Hai Economic Zone, Dong Hai 2 Ward Hai An District 180000 Hai Phong City
+49 6841 98090 sales@rrc-ps.com	+1 714 777 3604 usa@rrc-ps.com	+852 2376 0106 hkrrc@rrc-ps.cn	+86 755 8374 1908 hkrrc@rrc-ps.cn	vietnam@rrc-ps.com

Texim Europe - contact details



Headquarters & Warehouse

Elektrostraat 17
NL-7483 PG Haaksbergen
The Netherlands

T:	+31 (0)53 573 33 33
E:	info@texim-europe.com
Homepage:	www.texim-europe.com

The Netherlands

Elektrostraat 17
NL-7483 PG Haaksbergen

T: +31 (0)53 573 33 33
E: nl@texim-europe.com

Belgium

Zuiderlaan 14, box 10
B-1731 Zellik

T: +32 (0)2 462 01 00
E: belgium@texim-europe.com

UK & Ireland

St Mary's House, Church Lane
Carlton Le Moorland
Lincoln LN5 9HS

T: +44 (0)1522 789 555
E: uk@texim-europe.com

Germany

Bahnhofstrasse 92
D-25451 Quickborn

T: +49 (0)4106 627 07-0
E: germany@texim-europe.com

Germany

Martin-Kollar-Strasse 9
D-81829 München

T: +49 (0)89 436 086-0
E: muenchen@texim-europe.com

Austria

Warwitzstrasse 9
A-5020 Salzburg

T: +43 (0)662 216 026
E: austria@texim-europe.com

Nordic

Stockholmsgade 45
2100 Copenhagen

T: +45 88 20 26 30
E: nordic@texim-europe.com

Italy

Martin-Kollar-Strasse 9
D-81829 München

T: +49 (0)89 436 086-0
E: italy@texim-europe.com