FAST EnergyCam Modbus Slave Protocol

Table of Contents

Index of Tables

Index of Illustrations

Introduction

FAST EnergyCam is able to communicate via a wired connection in Modbus protocol. General description of Modbus is documented in a separate document “EnergyCam Modbus General” (FAST_EnergyCam-Protocol-MODBUS-General.pdf)

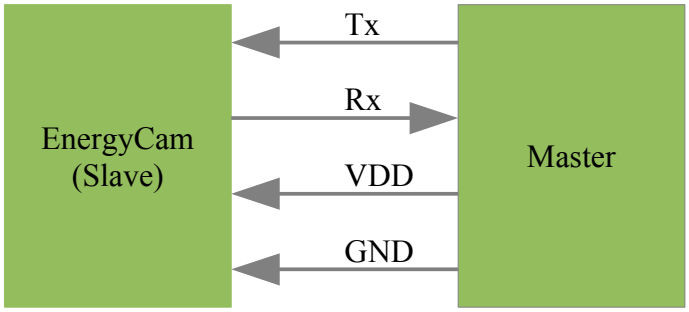
The interface uses 115200 baud, even parity and 1 stop bit.

Point-to-point communication

In case of a point-to-point communication two separate lines (Tx, Rx) are sufficient. EnergyCam acts as slave and waits for commands emitted by the master. Besides these two lines a single power and ground line are needed, too. In this case EnergyCam's Modbus slave address does actually not matter but is assumed to be 0x01.

EnergyCam operates internally with a down converted voltage of 2.1V. This means that master's Rx line will be driven by this lower voltage compared to the supplied VDD.

Illustration 1: Block diagram Point-to-point



Retrieve M-Bus secondary address

A standard M-Bus secondary address which is used for wired (M-Bus) and wireless M-Bus (wM-Bus) communication consists of 8 byte which can be accessed via these Modbus registers.

Name	Access	Modbus Register	Length [byte]	Description
Manufacturer ID	read-only	ManufacturerIdentification	2	Three letter code “FFD” which is coded as 0x18C4.
Identnumber	read-only	MBusIdentNumber	4	Identnumber coded as 8 BCD, internally derived from unique 64 bit DeviceID in a manner that a unique number results
Device type	read-write	OMSConfig[7:0] SetOMSConfig[7:0]	1	Only supported value are: 2: electricity (kWh) 3: gas (m^3) 7: water (m^3)
Version	N/A	N/A	1	Always 0x01

Table 1: M-Bus secondary address

Only the device type is changeable by the user.

Wakeup

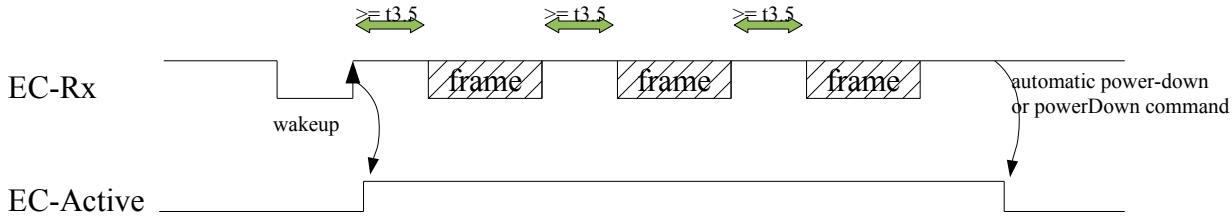
As an addition to Modbus, EnergyCam implements a special wake-up procedure in case the device is in power-down and needs to be waken up by the master.

The paragraph below describes voltage levels received by the internal processor and does not take into account a probably used level-shifter (e.g. for RS232).

The Master-Tx line is connected to EnergyCam's Rx line. This line is expected to be high during idle. During power-down the Rx input has enabled a rising edge input filter. Any rising edge will wakeup EnergyCam which starts after a certain boot time (t.b.d.) receiving Modbus frames. The Master can use any method

which produces a rising edge. E.g. writing a single byte of value 0x00 would only have a single rising edge. When maintaining the correct silent pause t3.5 (and boot time) afterwards the following Modbus frame will be interpreted correctly without CRC errors and a response can be expected from EnergyCam. When a wakeup byte is output during active mode of EnergyCam make sure to maintaining the correct silent pause t3.5 afterwards. This will results in a frame with CRC error and the wakeup byte is purged automatically.

The absolute value of t3.5 is mentioned in document *FAST_EnergyCam-Protocol-MODBUS-General.pdf.*



General Modbus timing

Due to EnergyCam's internal real-time requirements it cannot be guaranteed that every Modbus frame is received or delivered in time. When EnergyCam does not acknowledge a frame the Modbus frame has to be repeated.

Supported actions

EnergyCam supports several actions which can be triggered by Modbus:

- **ActionOCRInstallation**: Starts the EnergyCam installation, which is needed to be done once when put on a meter or EnergyCam got a power-loss.
- **ActionOCR**: Normally EnergyCam is reading the meter every 15 minutes. When an additional reading is needed start this action. Can only be done when EnergyCam is successfully installed. Check for InputReg ResultInstallation equals to OKDIGITSFOUND.
- **ActionPowerDown**: Normally EnergyCam put itself automatically into a power-down mode after a certain timeout (see chapter Automatic behaviors). In order to save power this action can be used to immediately enter power-down.

In order to trigger actions the following control flow has to be maintained:

- First check whether EnergyCam is idle:
  Repeat reading InputReg StatusEnergyCam until value is not equal to ActionOngoing
- Start action:
  Write a 0x0001 to HoldingReg Action...
- Check for action completeness:
  Repeat reading InputReg StatusEnergyCam until value is not equal to ActionOngoing
- Retrieve the result:
  ActionInstallation: Read InputReg ResultInstallation.
  ActionOCR: Read InputReg ResultOCR and following registers at once.

Modbus Register Addresses

Modbus defines register addresses counted from 1 (1-based). However some Modbus master implementations count from 0 (0-based). Please check your implementation and use the according column in tables below. Regardless of addressing the physical transmission is always 0-based.

In order to test a Modbus master the following registers are free to use:

- input registers: Test1[1], Test0
- holding registers: TestReadOnly1, TestReadOnly0, TestReadWrite1 and TestReadWrite0

Any read and write will not influence EnergyCam.

Supported Input Registers

Since Modbus always operates with 16 bit wide registers, EnergyCam defines some consecutive registers as blocks for information units wider than 16 bit.

Modbus address 0-based	Modbus address 1-based	Block name	Register name [16 bits]	Description
0x0000	0x0001	ProtocolVersion	ProtocolVersion	EnergyCam's Modbus protocol version: e.g. 4
0x0001	0x0002	Manufacturer-Identification	Manufacturer-Identification	Three letter code[2] for “FAST FORWARD AG” in MBUS FORMAT: “FFD” = 0x18C4
0x0002	0x0003	DeviceID	DeviceID3	DeviceID [63..48]
0x0003	0x0004		DeviceID2	DeviceID [47..32]
0x0004	0x0005		DeviceID1	DeviceID [31..16]
0x0005	0x0006		DeviceID0	DeviceID [15..0], example 0x4F92F42C109AB502
0x0006	0x0007	AppRevision	AppRevisionMajor	e.g. 2, Revision of Application
0x0007	0x0008		AppRevisionMinor	e.g. 0, meaning whole revision number is “2.0”
0x0008	0x0009	AppType	AppType	Type of app, 0x0100 for Sensor
0x0009	0x000A	Time	Time1	time [31..16], time since epoch in seconds (UTC), 0 means to 1.1.1970 00:00
0x000A	0x000B		Time0	time [15..0]
0x000B	0x000C	AppBuildnumber	AppBuildnumber1	Application build number [31..16]
0x000C	0x000D		AppBuildnumber0	Application build number [15..0]
0x000D	0x000E	EPLDRevision	EPLDRevision	EPLDRevision, e.g. 3
0x000E	0x000F	PCBRevision	PCBRevision	PCBRevision
0x000F	0x0010	OMSConfig	OMSConfig	Meter and wireless M-Bus configuration [7:0] device type (2: electricity; 3: gas; 7: water) [8:12] reserved [13] wM-Bus enabled [14] wM-Bus encrypted [15] wM-Bus installation mode (0: auto; 1: manual)
0x0010	0x0011	AppFirmwareType	AppFirmwareType	Application FirmwareType (like wM-Bus Stack Version T2, S2)
0x0011	0x0012	MBusIdentNumber	MBusIdentNumber1	Identnumber[31:16] in M-Bus format: 8 BCD digits, derived from DeviceID, example 0x12345678

1 Test registers are available with ProtokollVersion 5 and higher
2 According to *http://www.m-bus.com/files/MBDOC48.PDF*

Modbus address 0-based	Modbus address 1-based	Block name	Register name [16 bits]	Description
0x0012	0x0013		MBusIdentNumber0	Identnumber[15:0]
0x0013	0x0014	BLBuildNumber	BLBuildNumber1	Bootloader build number [31..16]
0x0014	0x0015		BLBuildNumber0	Bootloader build number [15..0]
0x0015	0x0016	BLFirmwareType	BLFirmwareType	Bootloader FirmwareType
0x0016	0x0017	Test1[1]	Test1	Test Register for Modbus master tests: Returns 0xABCD = 43981
0x0017	0x0018	Test0[1]	Test0	Test Register for Modbus master tests: Returns 0x1234 = 4660 For 32 Bit accesses (unsigned DWORD) read two16 bit registers starting from Test1. The resulting value should read 0xABCD1234 = 2882343476. In case of 0x1234ABCD = 305441741 the two 16 bit words should be swapped.
gap	**gap**			
0x001F	0x0020	StatusEnergyCam	StatusEnergyCam	Current status of EnergyCam: 0: Invalid 1: PowerDown (was actually in power-down before) 2: ActionOngoing 3: ActionCompletedSuccessfully 4: ActionCompletedWithError
0x0020	0x0021	ResultInstallation	ResultInstallation	Installation result: 0x0000: Invalid 0xIIFF: OKDIGITSFOUND (I = Integer part, F = fraction part) 0xFFFD: ONGOING 0xFFFE: NOTDONE 0xFFFF: ERROR
0x0021	0x0022	ResultOCR	ResultOCRValid	Result of last OCR: 0x0000: Invalid 0x0001: OK (all digits could be interpreted) 0x0003: OK (repeating last value due to Error detection) 0xFFFD: ONGOING 0xFFFE: NOTDONE 0xFFFF: ERROR (not all digits could be interpreted)
0x0022	0x0023	ResultOCRIntChar	ResultOCRIntChar7	May only be read when ResultOCRValid=OK. ResultOCRIntChar[7..0] contains OCR integer result as ASCII string, right aligned, left padded with '-' when fewer than 8 digits found on meter. Just whole integers, i.e. most right digit on meter is in OCRResult0 char
0x0023	0x0024		ResultOCRIntChar6	char
0x0024	0x0025		ResultOCRIntChar5	char
0x0025	0x0026		ResultOCRIntChar4	char
0x0026	0x0027		ResultOCRIntChar3	char
0x0027	0x0028		ResultOCRIntChar2	char
0x0028	0x0029		ResultOCRIntChar1	char
0x0029	0x002A		ResultOCRIntChar0	char
0x002A	0x002B	ResultOCR-	ResultOCRFracChar3	ResultOCRFracChar[3..0] contains OCR fraction result as ASCII

Modbus address 0-based	Modbus address 1-based	Block name	Register name [16 bits]	Description
		FracChar		string, left aligned. Right padded with blanks when fewer than 4 digits found on meter. Examples: "12345678.1234" or "___12345.1___" char
0x002B	0x002C		ResultOCRFracChar2	char
0x002C	0x002D		ResultOCRFracChar1	char
0x002D	0x002E		ResultOCRFracChar0	char
gap	**gap**			
0x003C	0x003D		OCRInfoRed	Horizontal position of detected Red [11:0] horizontal position [15] installed upside down
gap	**gap**			
0x0043	0x0044	ResultOCRInt	ResultOCRInt1	May only be read when ResultOCRValid=OK. Contains OCR result integer part as integer. ResultOCRInt[31..16]
0x0044	0x0045		ResultOCRInt0	ResultOCRInt[15..0]
0x0045	0x0046	ResultOCRFrac	ResultOCRFrac	May only be read when ResultOCRValid=OK. Contains the most significant decimal OCR result fraction part as integer. When more decimals should be retrieved use ResultOCR64 instead. ResultOCRFrac[15..0]
gap	**gap**			
0x004E	0x004F	ResultOCR64	ResultOCR64_3	ResultOCR64 contains the OCR result (integer and fraction) as 64 bit value. The OCR value is scaled by factor 1000, e.g. 12345.1 will be read as 12345100.
0x004F	0x0050		ResultOCR64_2	
0x0050	0x0051		ResultOCR64_1	
0x0051	0x0052		ResultOCR64_0	
gap	**gap**			
0x7FFF	0x8000	UpdateCRCOK		Calculates CRC of currently stored update image and return 0 (CRC wrong) or 1 (CRC OK). When OK firmware update process is started (leads to a reboot of EnergyCam)

Table 2: Supported input registers

Examples

Get AppRevision

Here an example for a reading access of two consecutive input registers starting at address 0x0006 (0-based, "AppRevision") which return the revision as e.g. "2.0":

0x0006 → 2
0x0007 → 0

```
Master-Tx:01 04 00 06 00 02 91 CA
```

Value [hex]	Description
01	EnergyCam's slave address
04	Code for read multiple input registers
00	Address of first input register (higher byte)
06	Address of first input register (lower byte)
00	Number of registers to read from (higher byte)
02	Number of registers to read from (lower byte)
91	CRC (higher byte)
CA	CRC (lower byte)

Table 3: Example request for read from input registers "AppRevision"

In case of success EnergyCam responds as follows:

```
Master-Rx:01 04 04 00 02 00 00 5A 44
```

Value [hex]	Description
01	EnergyCam's slave address
04	Code for read multiple
04	Byte count for data following
00	Data for first register (higher byte)
02	Data for first register (lower byte)
00	Data for second register (higher byte)
00	Data for second register (lower byte)
5A	CRC (higher byte)
44	CRC (lower byte)

Table 4: Example response for successful read from input registers "AppRevision"

Get ResultOCR

Here an example for a reading access of three consecutive input registers starting at address 0x0043 (0-based, "ResultOCRInt" and "ResultOCRFrac" are read at once). Return values are

0x0043 → 0x0001 (= 1_{10}, higher byte of ResultOCRInt)
0x0044 → 0x0D66 (= 3430_{10}, lower byte of ResultOCRInt)
0x0045 → 0x0001 (= 1_{10}, ResultOCRFrac)

This OCR value is calculated as: $1*65536 + 3430 + 1*10^{-1} = 68966.1$

```
Master-Tx:01 04 00 43 00 03 41 DF
```

Value [hex]	Description
01	EnergyCam's slave address
04	Code for read multiple input registers
00	Address of first input register (higher byte)
43	Address of first input register (lower byte)
00	Number of registers to read from (higher byte)
03	Number of registers to read from (lower byte)
41	CRC (higher byte)
DF	CRC (lower byte)

Table 5: Example request for read from input registers ResultOCR

In case of success EnergyCam responds as follows:

```
Master-Rx: 01 04 06 00 01 0D 66 00 01 7E 20
```

Value [hex]	Description
01	EnergyCam's slave address
04	Code for read multiple
06	Byte count for data following
00	Data for first register (higher byte)
01	Data for first register (lower byte)
0D	Data for second register (higher byte)
66	Data for second register (lower byte)
00	Data for third register (higher byte)
01	Data for third register (lower byte)
7E	CRC (higher byte)
20	CRC (lower byte)

Table 6: Example response for successful read from input registers ResultOCR

Supported Holding Registers

Despite that Modbus defines holding registers as read-write, most of them are write-only. Reading from write-only holding registers result in a Modbus exception response. Readable holding registers are marked as read-only or read-write.

Modbus address 0-based	Modbus address 1-based	Block name	Block access	Register name [16 bits]	Description
0x0000	0x0001	SetTime	write-only	SetTime1	SetTime[31..16], Time since epoch in seconds (UTC), 0 means to 1.1.1970 00:00:00, 1360751350 means 13. Feb 2013 10:29:10
			write-only	SetTime0	SetTime[15..0]
0x0002	0x0003	SetOMSCfg	write-only	SetOMSCfg	Meter and wireless M-Bus configuration [7:0] Device type (2: electricity; 3: gas; 7: water) [8:12] Reserved, write 0 [13] wM-Bus enabled[3] [14] wM-Bus encrypted [15] wM-Bus installation mode[3] (0: auto; 1: manual) Only device types 2, 3 and 7 are supported.
0x0003	0x0004	ExpIfc	read-write	ExpIfc1	Expansion Interface serial configuration [3:0] Baud rate, coded as: 0 115200, 1 600 2 1200, 3 2400 4 4800, 5 9600 6 14400, 7 19200 8 38400, 9 57600 10 115200

3 Changes of these items are effective with the next installation cycle

Modbus address 0-based	Modbus address 1-based	Block name	Block access	Register name [16 bits]	Description
					[5:4] Reserved, write 0 [7:6] Parity 0 Even 1 None 2 Odd [11:8] Reserved, write 0 [13:12] Stopbits 0 One 1 Two [15:14] Reserved, write 0
				ExpIfc0	Expansion Interface protocol [7:0] Slave address (primary address) Modbus 1..247 M-Bus 0..250 (0 reacts on any address) [8] AutoPowerDown (when active wakeup is needed) [11:9] Reserved, write 0 [15:12] Protocol, coded as: 0 Modbus 2 M-Bus ExpIfc1 and ExpIfc0 have to be written by a single Modbus access. A Modbus response is returned with current serial settings and they are changed immediately afterwards.
gap	**gap**				
0x0007	0x0008	TestReadOnly1[4]	read-only	TestReadOnly1	Test register for Modbus master tests: Returns 0xDEAD = 57005
0x0008	0x0009	TestReadOnly0[4]	read-only	TestReadOnly0	Test register for Modbus master tests: Returns 0xBEEF = 48879. For 32 Bit accesses (unsigned DWORD) read two16 bit registers starting from Test1. The resulting value should read 0xDEADBEEF = 3735928559. In case of 0xBEEFDEAD = 3203391149 the two 16 bit words should be swapped.
0x0009	0x000A	TestReadWrite1[4]	read-write	TestReadWrite1	Test register for Modbus master tests: Returns 0xFA51 = 64081. Can be written, too. Value is lost after reboot.
0x000A	0x000B	TestReadWrite0[4]	read-write	TestReadWrite0	Test register for Modbus master tests: Returns 0xFFDD=65501. Can be written, too. Value is lost after reboot. For 32 Bit accesses (unsigned DWORD) read two16 bit registers starting from Test1. The resulting value should read 0xFA51FFDD = 4199677917. In case of 0xFFDDFA51 = 4292737617 the two 16 bit words should be swapped.

4 Test registers are available with ProtokollVersion 5 and higher

Modbus address 0-based	Modbus address 1-based	Block name	Block access	Register name [16 bits]	Description
gap	**gap**				
0x001F	0x0020	ActionOCR-Installation	write-only	ActionOCR-InstallationTO	timeout in [s] after installation is treated as failed unless success
0x0020	0x0021		write-only	ActionOCR-Installation	0: Command Ignored 1: Installation is started, end of operations when EnergyCamStatus = ActionCompleted
0x0021	0x0022	ActionOCR	write-only	ActionOCR	0: Command Ignored 1: Photo is taken and OCR is started, end of operations when EnergyCamStatus = ActionCompleted
gap	**gap**				
0x0024	0x0025	ActionPowerDown	write-only	ActionPowerDown	0: Command Ignored 1: Unless no action is ongoing PowerDown is immediately entered, end of operations cannot be checked since this would wake EnergyCam again. It should checked before StatusEnergyCam != ActionOngoing otherwise ActionPowerDown is rejected by Modbus
gap	**gap**				
0x0034	0x0035	OCRConfig	read-write	OCRConfig	OCR configuration (perform a read-modify-write to preserve reserved values) [0] OCR option: read decimal[3] [5:1] reserved: do not change, keep current values [10:6] OCR reading timer: value in minutes[3] [1..15] [15:11] OCR max increment per reading
gap	**gap**				
0x0038	0x0039	OCRIncrConfig	read-write	OCRIncrConfig	OCR max increment per hour [11:0] increment per hour [15:12] = 0x0 on read [15:12] must be set to 0xA on write
0x0039	0x003A	OCRReadIncrConfig	read-write	OCRReadIncrConfig	OCR max increment per reading (used if OCRConfig [15:11] = 0x1F)
0x003A	0x003B	OCRReadReadTimer	read-write	OCRReadTimer-Config	OCR Reading Timer (used if OCRConfig[6:10] = 0x1F)
gap	**gap**				
0x003C	0x003D	OCRConfigMeter	read-write	OCRConfigMeter	OCR Config special Meter [7:0] reduced window width 0: 100% 1: 90% ... [15:8] special meter 0: default 1: black on white with gaps ...

Modbus address 0-based	Modbus address 1-based	Block name	Block access	Register name [16 bits]	Description
gap	**gap**				
0x7FFF	0x8000	UpdateChunk	write-only	UpdateChunkStart-Addr1	Update chunk. Chunk has to be written in a single frame consisting of up to 122 words. Byte address (32 bit) of first binary data following relative to first byte in update image. Very first byte of update image has address 0x00000000. Amount of ChunkData in bytes have to be a multiple of 4, except for very last chunk! Maximum is 120 words = 240 bytes (240 mod 4=0)
				UpdateChunkStart-Addr0	
				UpdateChunkData0	
0x8078	0x8079			UpdateChunkData... UpdateChunk-Data119	

Table 7: Supported holding registers

Examples

Get OCRConfig

Here an example for a reading access of one holding register starting at address 0x0034 (0-based, “OCRConfig”) which return the OCR configuration as e.g. read decimal and 15 minutes reading timer:

0x0034 → 0x43C9 → 01 00 00 11 11 00 10 01 → read decimal, 15 minutes reading timer

```
Master-Tx: 01 03 00 34 00 01 C5 C4
```

Value [hex]	Description
01	EnergyCam's slave address
03	Code for read multiple holding registers
00	Address of first holding register (higher byte)
34	Address of first holding register (lower byte)
00	Number of registers to read from (higher byte)
01	Number of registers to read from (lower byte)
C5	CRC (higher byte)
C4	CRC (lower byte)

Table 8: Example request for read from a holding register

In case of success EnergyCam responds as follows:

```
Master-Rx: 01 03 02 43 C9 49 08 22
```

Value [hex]	Description
01	EnergyCam's slave address
03	Code for read multiple holding registers
02	Byte count for data following
43	Data for first register (higher byte)
C9	Data for first register (lower byte)
08	CRC (higher byte)
22	CRC (lower byte)

Table 9: Example response for successful read from holding registers

Set ActionPowerDown

Here is an example for a single writing of a Modbus holding register at address 0x0024 (0-based, “ActionPowerDown”). This command will send EnergyCam in power down.

0x0024 ← 0x01

```
Master-Tx: 01 06 00 24 00 01 08 01
```

Value [hex]	Description
01	EnergyCam's slave address
06	Code for write single holding registers
00	Address of holding register (higher byte)
24	Address of holding register (lower byte)
00	Data (higher byte)
01	Data (lower byte)
08	CRC (higher byte)
01	CRC (lower byte)

Table 10: Example request for write to single holding registers

In case of success EnergyCam responds as follows:

```
Master-Rx: 01 06 00 24 00 01 08 01
```

Value [hex]	Description
01	EnergyCam's slave address
06	Code for write single holding registers
00	Address of holding register (higher byte)
24	Address of holding register (lower byte)
00	Data (higher byte)
01	Data (lower byte)
08	CRC (higher byte)
01	CRC (lower byte)

Table 11: Example response for successful write to single holding register

Set ActionOCRInstallation

Here is an example for a multiple writing of a Modbus holding register starting at address 0x001F (0-based, “ActionOCRInstallation”). This command will trigger an OCR installation (timeout 100=0x64 seconds).

0x001F ← 0x64
0x0020 ← 0x01

```
Master-Tx: 01 10 00 1F 00 02 04 00 64 00 01 32 FC
```

Value [hex]	Description
01	EnergyCam's slave address
10	Code for write multiple holding registers
00	Address of first holding register (higher byte)
1F	Address of first holding register (lower byte)
00	Number of registers to write to (higher byte)
02	Number of registers to write to (lower byte)
04	Byte count for data following
00	Data for first register (higher byte)
64	Data for first register (lower byte)
00	Data for second register (higher byte)
01	Data for second register (lower byte)
32	CRC (higher byte)
FC	CRC (lower byte)

Table 12: Example request for write to multiple holding registers

In case of success EnergyCam responds as follows:

```
Master-Rx: 01 10 00 1F 00 02 70 0E
```

Value [hex]	Description
01	EnergyCam's slave address
10	Code for write multiple holding registers
00	Address of first holding register (higher byte)
1F	Address of first holding register (lower byte)
00	Number of registers to write to (higher byte)
02	Number of registers to write to (lower byte)
70	CRC (higher byte)
0E	CRC (lower byte)

Table 13: Example response for successful write to multiple holding registers

Error Case

Example for EnergyCam's response to an unknown address, here a try to write to two consecutive holding registers starting at 0x00C8 which does not exist:

0x00C8 ← 1
0x00C9 ← 2

```
Master-Tx:01 10 00 C8 00 02 04 00 01 00 02 2E 58
```

```
Master-Rx:01 90 02 CD C1
```

Value [hex]	Description
01	EnergyCam's slave address
90	Error code [7] Always 1 [6..0] Function code of failed access (here 0x10 for write multiple holding registers)
02	Exception code according to Modbus Exception Codes 1 illegal function **2 illegal register address** 3 illegal data value 4 Slave device failure (can be caused by illegal data tried to be written)
CD	CRC (higher byte)
C1	CRC (lower byte)

Table 14: Example response for write to unknown holding registers

Automatic behaviors

Usually EnergyCam does not start actions autonomously, it just reacts on commands sent by the master.

Exceptions to this are:

- OCR is started every $t_{OCRDefault}$ unless not triggered in this interval by Modbus host. This is needed for internal error corrections.
- Power-down is entered automatically after $t_{modbusTO}$.This period is (re-)started on wakeup (when EnergyCam was really in power-down) or when a valid Modbus frame has been received (e.g. no CRC error occurred).

Symbol	time	Description
$t_{modbusTO}$	5 seconds	After this time without communication on Modbus EnergyCam enters power-down automatically
$t_{OCRDefault}$	15 minutes	After this time a OCR reading will take place unless not triggered in this interval by Modbus host

Table 15: Timings for automatic behaviors

History

Date	Author	Version	Changes
8th May 12	SPR	0.1	Initial, reviewed by CRG
8th Sep 12	SPR	0.2	Fixed “Table 3: Example request for read from input registers”. Removed wrong row “Byte count” Added to chapter “Point-to-point communication” master's Rx line voltage. Changed format of holding registers ResultOCRIntx and ResultOCRFracx
21th Sep 12	SPR	0.3	Swapped ActionInstallationTO and ActionInstallation to have a change to write them with a single access Renamed ActionInstallationx to ActionOCRInstallationx, changed register addresses; changed return codes of result registers, added time, build number, EPLDRevision, PCBRevision
25th Oct 12	SPR	0.4	Added EPLDRevision, PCBRevision
11th Dec 12	SPR	0.5	Added number to Modbus ProtocolVersion (currently 2). Changes to wakeup, automatic behaviors
14th Jan 13	SPR	0.6	Better text contrast, added serial interface definition, corrected address of HoldingReg ActionPowerDown (from 0x23 to 0x25). Added chapters General Modbus timing, Supported actions.
30th Jan 13	SPR	0.7	Renamed input registers ResultOCRInt to ResultOCRIntChar and ResultOCRFrac to ResultOCRFracChar.

Date	Author	Version	Changes
			Added input registers ResultOCRInt and ResultOCRFrac.
13th Feb 13	SPR	0.8	Valid for AppBuildNumbers >= 8162: Added InputRegisters: OMSConfig, AppFirmwareType, MBusIdentNumber, BLBuildNumber, BLFirmwareType; Renamed InputRegisters: Revision to AppRevision, BuildNumber to AppBuildNumber, Serialnumber to DeviceID
17th May 13	SPR	0.9	Input register: ResultOCR: Added new encoding (repeating last value due to Error detection)
10th Jun 13	SPR	1.0	Added registers for firmware update: input register UpdateCRCOK, holding register UpdateChunk
19th Jun 13	SPR	1.1	Added documentation for input register OMSConfig, added holding registers SetOMSCfg and ExpIfc, added chapter “M-Bus secondary address”
27th Jun 13	SPR	1.2	Added footnote when a write to holding registers item takes effect. Added holding register OCRConfig
24th Oct 13	SPR	1.3	Added parity and stop bits configuration in Holding Register ExpIfc1
31th Oct 13	SPR	1.4	Fixed some typos. Moved History to end of document
28th Nov 13	FBL	1.5	Min Val for OCR Reading Timer set to 2
2nd Dec 13	SPR	1.6	Renamed wM-Bus installation mode from “OMS auto” to “auto”
4th Mar 14	CHZ	1.7	Added chapters Set ActionPowerDown and Set ActionOCRInstallation.
5th Mar 14	SPR	1.8	Fixed wrong Table: Example request for read from input registers
5th Mar 14	CHZ	1.9	Added chapter Get OCRConfig: Example request for read from holding registers
17th Mar 14	CHZ	1.10	Cosmetic
17th Apr 14	SPR	1.11	Typo fixed in table 3
22th Apr 14	CRG	1.14	Cosmetic
28th Apr 14	SPR	1.15	Unification of naming
28th Jul 14	SPR	1.16	Added example for read of ResultOCR
31th Jul 14	SPR	1.17	Changed “OCR reading timer” from [2..15] to [1..15]
15th Sep 14	SPR	1.18	Changed “ProtocolVersion” from 2 to 4
29th Oct 14	SPR	1.19	Added 0-based Modbus register addresses
30th Oct 14	SPR	1.20	Added test registers
25th Mar 14	SPR	1.21	Corrected Filenames of referenced .pdf
20th Apr 15	SPR	1.22	Added Exception codes
27th Apr 15	SPR	1.23	Added input registers ResultOCR64
30th Apr 15	CRG	1.24	Added Input Register OCRInfoRed, Holding Register OCRIncrConfig, OCRReadIncrConfig
08th Jul 15	SPR	1.25	Added missing exception code in table 14
19th Nov 15	SPR	1.26	Limitation of ResultOCRFrac commented. Not used characters in ResultOCRIntChar are filled with '-'

Table 16: History