

**M4020**

**M4036**

**M4005**

MODBUS IMPLEMENTATION DESCRIPTION

## 1 Revision

Version	Date	Name	Description
V1.00	25.02.2019	IC	<b>Valid from M3118/M3329/M3136 FW-Version: 2.30</b> Initial revision.
V1.10	21.06.2019	FH	<b>Valid from M3118/M3329/M3136/M3329LW/M3120/M3320 FW-Version: 2.40</b> Implementation of M3120/M3320
V1.20	13.04.2023	IC	<b>Valid from M4020/M4036/M4005 FW Version: 1.00.16</b> Introduce M4020/M4036/M4005

## 2 Device Scope

This document is valid for the following devices from Mostec AG.

- M4020 / M4036 / M4005

Hereafter referred to Mxxxx.

## Contents

1	Revision .....	2
2	Device Scope .....	2
3	General Modbus Information .....	4
3.1	Documentation of the Modbus protocol .....	4
3.2	Modbus testing tool .....	4
3.3	Protocol definitions, as implemented in Mxxxx .....	4
3.4	Modbus RTU function codes implemented in Mxxxx .....	4
3.5	Data representation .....	4
3.6	Addressing scheme .....	5
3.7	Error handling .....	6
3.8	User levels, Password protection .....	7
3.9	Writing registers, data retention .....	7
4	Implemented Modbus registers in the Mxxxx .....	8
4.1	Mxxxx Modbus registers .....	8
4.1.1	Device Information Page .....	8
4.1.2	USB Logger Setting Page .....	8
4.1.3	Relay/Alarm Setting Page .....	9
4.1.4	Measuring Setting Page .....	11
4.1.5	Measuring Value Page .....	13
4.1.6	Modbus Slave Setting Page .....	13
5	Detailed description of the implemented Modbus registers .....	14
5.1	Communication Setup Page .....	14
5.1.1	User levels and passwords .....	14
5.1.2	Modbus register offset .....	14
5.1.3	Configuration of the RS485 interface .....	15

## 3 General Modbus Information

### 3.1 Documentation of the Modbus protocol

For detailed information about Modbus specifications, please refer to the following documents:

- Modbus\_over\_serial\_line\_V1\_02.pdf
- Modbus\_Application\_Protocol\_V1\_1b3.pdf

These documents are available on the Modbus website: [www.modbus.org](http://www.modbus.org)

Go to tab „Technical Resources“.

### 3.2 Modbus testing tool

On the web, there are plenty of Modbus testing tools or Modbus libraries available for C++, Python or other programming languages.

For manually accessing the Mxxxx over Modbus, for instance to predefine the Modbus address, baud rate, or other items, the tool „Modbus Poll“ is a feasible choice. It can be purchased at <http://www.modbustools.com>

### 3.3 Protocol definitions, as implemented in Mxxxx

Modbus mode:	RTU
Start bits:	1
Data bits:	8
Stop bits:	1 (default), 2
Parity:	None (default), Odd, Even
Baud rate:	4800, 9600, 19200, 38400(default), 57600, 115200
Device address:	1 to 247 (default)

### 3.4 Modbus RTU function codes implemented in Mxxxx

- #3 Read Holding Registers
- #4 Read Input Registers
- #6 Write Single Register
- #16 Write Multiple Registers

For detailed description of these functions please consult the document „Modbus\_Application\_Protocol\_V1\_1b3.pdf“.

With the Mxxxx, reading any register is performed by either command #3 or #4. There is no difference in handling the information between these two commands.

### 3.5 Data representation

Each Modbus register contains two bytes, the data length of a command and an answer is always a multiple of two registers.

The high byte (first byte) of a register contains the last digit of a value or string, the first digit of a value or string is found on the low byte (second byte) of the last register of the interesting register chain.

The first byte of a register always contains the higher order bits, the second byte contains the lower order bits.

**Decimal values:**

Integer Decimal values are translated to hexadecimal numbers.

Non-integer decimal values are represented as single precision float values.

See below for examples.

**For integer 16 bit values:**

Example: A 16-bit value of 22'354.

Converted to hex: 5752

Register: Value (bytes 1, 2): 0x5752

When using Modbus Poll, select „Signed / Unsigned“ to correctly interpret values.

**For integer 32 bit values:**

Example: A 32-bit value of 12'345'678.

Converted to hex: BC614E

First register: Higher bytes of the value (bytes 1, 2): 0x00BC

Second register: Lower bytes of the value (bytes 3, 4): 0x614E

When using Modbus Poll, select „Long ABCD“ to correctly interpret long values.

**For float values:**

The mantissa of the value is stored on the second register, its exponent in the first register.

The float data format is implemented according to IEEE 754, single precision.

Example: 2.5, converted to a 32-bit float value → (Hex value 0x40200000).

First register: 0x4020

Second register: 0x0000

When using Modbus Poll, select „float ABCD“ to correctly interpret float values.

**For ASCII-text strings:**

Example: Text sample: „Text“. ASCII-code is: 0x54 0x65 0x78 0x74

First register: 0x5465

Second register: 0x7874

Encoding: UTF8

Character set: ISO/IEC 8859-1:1998 (Unicode 0x20-0xFF)

### 3.6 Addressing scheme

The addressing scheme of the Mxxxx is „Base 0“ (first register number is 0).

A register offset is available on register number 0000.

Using this register offset, one can adjust the absolute starting point of the register bank to fit for instance already existing implementations.

The register offset is signed with a range of -32768...32767.

For instance by setting the offset to 1, the device is becoming „Base 1“.

By default, the register offset is set to 0, thus the first user register is on number 100.

Please note: The register offset is always found on register number 0000, independent of its value.

The offset affects only register numbers 0002 and up.

The register numbers given on the following pages are always relative numbers.

The absolute number of a register is calculated by adding the register offset to the relative address.

### 3.7 Error handling

Transmission errors (corrupt telegrams) are detected by the Mxxxx. Corrupt telegrams are discarded and the device is waiting for a next, correct telegram.

Errors on application layer are answered with an error message. In case the answer consists of an error code, the leading bit (0x80) of the function code is set, signaling the error condition. The following error codes are implemented in the Mxxxx:

Error code, hex	Error type
0x00	No error
0x01	Illegal function code was sent to the sensor
0x02	Illegal data address (invalid register number, access denied)
0x03	Illegal data value (value out of range)
0x04	Slave device error (operation not successfully completed)

Error code 0x01 is returned when a function code other than #3, #4, #6, #16 is sent to the Mxxxx.

Error code 0x02 is returned in the following cases:

- Any attempts to undefined registers
- Any attempts to registers on a higher operator level than actually selected (access denied)
- When reading too many registers, so undefined registers would be attempted
- When writing too many or not enough registers at once, or on a wrong starting address

Error code 0x03 is returned when writing invalid data to a register. Invalid data means any value out of the range of the specific register (value below or above limits, value not part of a list of possible values).

In this case, the last valid data is restored on the specific Modbus register and no change is active.

Error code 0x04 is typically returned when trying to log-in to a higher user level with a wrong password or to an inexistent user level. In these cases, the log-in fail, the operation is not successfully completed.

### 3.8 User levels, Password protection

Mxxxx have implemented three user levels, level 0, 1 and 2.

Reading registers is possible on any user level, except some specific registers.

To prevent any unwanted configuration changes, most writing attempts are possible only on user level 1 or 2.

For all user levels, default passwords are stored in the Mxxxx. These passwords can be changed by the user. Changed passwords are stored in the non-volatile memory of the device.

User levels and default passwords of the Mxxxx:

User level	Code, hex	Default password, hex
0	0x03	0x00000000
1	0x0C	0x01145DEA
2	0x30	0x00F479CE

After each power-up, the device is reset to user level 0.

When trying to change the user level to an invalid level or using a wrong password, the Mxxxx remains on the last valid user level, error code 0x04 is returned.

### 3.9 Writing registers, data retention

It is a well-known fact, that FLASH memories only allows about 100'000 write attempts. By exceeding this limit, the FLASH memory might get damaged; resulting in data lost or corrupted data. A device with a damaged FLASH is no longer operable.

Almost all writable registers are write protected. For persistent change of these registers unlock the Mxxxx EEPROM first. To unlock write 0x5752 to register 3999. Otherwise changed values will be lost after the next power cycle. For non-persistent changes let the EEPROM locked.

## Attention!!!

**The Modbus Master controller must make sure to write any configuration data only upon change and only during the commissioning phase of a system!**

**Automatic, periodic writes of data during normal operation with unlocked EEPROM must be prohibited!**

## 4 Implemented Modbus registers in the Mxxxx

### 4.1 Mxxxx Modbus registers

Except register 0000, all register addresses are relative to the offset stored in register 0000.

Example:

Register offset is 999. Register 200 shall be read.

The controller must read from register 1199. Default Register offset is 0.

Registers sorted in ascending register number order:

#### 4.1.1 Device Information Page

Register			Access levels		Comments
Start-Register	Name	Count	Read	Write	
0000 fix	Register offset	1	0	2 (*)	Starting point of Modbus registers
<b>Device Information Page</b>					
30	Device type	1	0	-	uInt 16bit equal to 0x4020/4036
31	Hardware version	1	0	-	uInt 16bit
32	User end firmware version	8	0	-	String Ex:"1.0.6 "
42	User end firmware version	2	0	-	uInt 32bit Ex:10006
44	Serial number	1	0	-	uInt 16bit
<b>Device Setup Page</b>					
60	Language	1	0	2(*)	0 = English, 1= German, 2 = French

#### 4.1.2 USB Logger Setting Page

Register			Access levels		Comments
Start-Register	Name	Count	Read	Write	
<b>USB Logger Setting Page</b>					
200	Date	2	0	1	uInt 32bit 0x00YY'MMDD
202	Time	1	0	1	uInt 16bit 0xHHMM
210	Log interval	1	0	1(*)	uInt16 [s]



### 4.1.3 Relay/Alarm Setting Page

Register			Access levels		Comments
Start-Register	Name	Count	Read	Write	
	<b>Relay/Alarm Setting Page</b>				
	<b>Relay settings K1:</b>				
300	Set point	2	0	1(*)	Float 32bit [Display Unit]
302	Hysteresis	2	0	1(*)	Float 32bit [Display Unit] (***)
304	Relay setting	1	0	1(*)	<b>bRelInverse:1</b> (LSB) <b>bLedInverse:1</b> <b>bUpperLimit:1</b> <b>bMeasUnit:1</b> <b>0=Main Display Unit</b> <b>1=Temperature</b> <b>reserved:13</b>
305	Relay status	1	0	-	<b>bRelActive:1</b> (LSB) <b>reserved:15</b>
306	Relay turn on delay	1	0	1(*)	uInt16 [s] Range 0...3600s
307	Relay turn off delay	1	0	1(*)	uInt16 [s] Range 0...3600s
308	Relay minimal on time	1	0	1(*)	uInt16 [1/100s] Range 0...10.00s
310	<b>Relay settings K2</b>	10	0	1(*)	Same data structure as relay settings K1 from Reg 300-309
320	<b>Relay settings K3</b>	10	0	1(*)	Same data structure as relay settings K1 from Reg 300-309
330	<b>Relay settings K4</b>	10	0	1(*)	Same data structure as relay settings K1 from Reg 300-309

(\*\*\*) Hysteresis for K1 / K2 are the same. As well for K3/K4 and SP1/SP2 of alarm relay

Register			Access levels		Comments
Start-Register	Name	Count	Read	Write	
	<b>Alarm relay settings</b>				
380	Alarm set point 1	2	0	1(*)	Float 32bit [Display Unit]
382	Alarm set point 2	2	0	1(*)	Float 32bit [Display Unit]
384	Relay setting	1	0	1(*)	<b>bRelInverse:1</b> (LSB) <b>bSP1Enable:1</b> <b>bSP1UpperLimit:1</b> <b>bSP2Enable:1</b> <b>bSP2UpperLimit:1</b> <b>bAlarmOnManualMode:1</b> <b>bBrokenWireSensor:1</b> <b>bUsbDiskFull:1</b> <b>bLedInverse:1</b> <b>bSP1MeasUnit:1</b> <b>bSP2MeasUnit:1</b> <b>bBrokenWirePT100:1</b> <b>reserved:5</b>
385	Alarm status	1	0	-	<b>bSP1Active:1</b> (LSB) <b>bSP2Active:1</b> <b>bManuelModeActive:1</b> <b>bPT100_BorkenWireActive:1</b> <b>bUSB_DiskFullActive:1</b> <b>bNo1WireSensorActive:1</b> <b>bPumpMonitoringActive:1</b> <b>reserved:9</b>
386	SP1 turn on delay	1	0	1(*)	uInt16 [s] Range 0...3600s
387	SP2 turn on delay	1	0	1(*)	uInt16 [s] Range 0...3600s
388	Pump monitoring	1	0	1(*)	uInt16 [s] Range 0...10'800s
389	SP1 & SP2 hysteresis	1	0	1(*)	uInt16 [Digit]

Register	Access levels	Comments	Register	Access levels	Comments
Name	Count	Read	Name	Count	Read
	<b>Relay status page</b>				
490	Relay summary status	1	0	-	<b>bRel1Active:1</b> (LSB) <b>bRel2Active:1</b> <b>bOpticalLimitActive:1</b> * <b>bRel3Active:1</b> ** <b>bRel4Active:1</b> ** <b>bRel5/AlarmActive:1</b> ** <b>reserved:10</b> * Only M3118/M3329 ** Only M4020/M4036/M4005

#### 4.1.4 Measuring Setting Page

Register			Access levels		Comments
Start-Register	Name	Count	Read	Write	
	<b>Measuring Setting Page</b>				
	<b>Conductivity Devices</b>				M4036
501	Range	1	0	1(*)	uInt16 0 = Auto 1 = 2µS 2 = 20µS 3 = 200µS 4 = 2mS 5 = 20mS 6 = 200mS
502	Correction Factor	2	0	1(*)	Float 32bit Range 0.0000...10.0000
504	Temperature Measuring Manually/Pt100	1	0	1(*)	uInt16 0 = Pt100 1 = Manual 2* = Digital Sensor (Modbus)
505	Relais Manually/Auto	1	0	1	uInt16 0 = Auto 1 = Manual
506	Manual Temperature	2	0	1(*)	Float 32bit [°C] Range: 0.0...100.0 °C
508	Temperature Slope	2	0	1(*)	Float32 [%/°C] Range: 0.00...8.00 %/°C
511	Broken Wire Threshold	1	1	1	uC8 [1/10%]
	<b>Display Units</b>				M4005
520	Input Zero	2	0	1(*)	Float 32bit [%] Range: 0...100 %
522	Input Gain	2	0	1(*)	Float 32bit [%] Range: 0...100 %
528	Display Value Min	1	0	1(*)	Int16 [Digit] Range: -9999...9999
529	Display Value Max	1	0	1(*)	Int16 [Digit] Range: -9999...9999
	<b>pH/rH Devices</b>				M4020
540	Range	1	0	1	uInt16 0 = mV 1 = pH
541	Relais Manually/Auto	1	0	1	uInt16 0 = Auto 1 = Manual
542	Temperature Measuring Manually/Pt100	1	0	1	uInt16 0 = Pt100 1 = Manual 2* = Digital Sensor (Modbus/1Wire) *Only for M4020

544	Manual Temperature	2	0	1	Float 32bit [°C] Range: 0.0...100.0 °C
546	User Calibration Date	2	0	-	uInt 32bit 0x00YY'MMDD
548	User Calibration Time	2	0	-	uIn 32bit 0xHHMMSSUS
554	4-20mA Input Min	1	0	1	Int16 4mA = x mV Input Range: -2000...2000 mV
555	4-20mA Input Max	1	0	1	Int16 20mA = x mV Input Range: -2000...2000 mV

Register		Access levels		Comments	
Start-Register	Name	Count	Read	Write	
	<b>Current Output Settings</b>				
600	Output #1 Input Range Min (Primary Unit)		0	1(*)	Float 32bit [Display Unit] Range: -2.00...16.00 pH -1000...1000 mV 0...2000 uS,mS -9999...9999 Digit
602	Output #1 Input Range Max (Primary Unit)		0	1(*)	
604	Output #1 Output Range Min (Primary Unit)		0	1(*)	Float 32bit [mA] Range: 0.00...20.000 mA
606	Output #1 Output Range Max (Primary Unit)		0	1(*)	
615	Output #1 Setup Register		0	1(*)	<b>bMeasUnit:1</b> (LSB) 0=Main Display Unit 1=Temperature <b>bspare:1</b> <b>I2_AlarmMode:2</b> 0 = Off, 1 = 3.6mA, 2= 22mA <b>reserved:12</b>
620	<b>Output #2</b>	20	0	1(*)	Same data structure as output settings #1 from Reg 600-619

#### 4.1.5 Measuring Value Page

Register			Access levels		Comments
Start-Register	Name	Count	Read	Write	
	<b>Measuring Value Page</b>				
1000	Measuring value	2	0	-	Float 32bit
1002	Measuring unit	4	0	-	String 8 char
1006	Temperature	2	0	-	Float 32bit [°C]
1008	Measuring value based on $\mu\text{S}$ , for conductivity devices only	2	0	-	Float 32bit Unit = [ $\mu\text{S}$ ]
1010	Measuring value based on mV, for pH/rH devices only	2	0	-	Float 32bit Unit = [mV]
1012	Measuring value based on the 2-wire loop powered mA input of the instrument	2	0	-	Float 32bit Unit = [mA]

#### 4.1.6 Modbus Slave Setting Page

Register			Access levels		Comments
Start-Register	Name	Count	Read	Write	
	<b>Modbus Slave Setting Page</b>				
3096	Device address	2	0	2 (*)	
3098	Address limit minimum	2	0	-	
3100	Address limit maximum	2	0	-	
3102	Baud rate	2	0	2 (*)	
3104	Baud rate limit minimum	2	0	-	
3106	Baud rate limit maximum	2	0	-	
3108	Uart modus modbus	1	0	2 (*)	
3288	User level	4	0	0	
3292	User level passwords	4	-	2 (*)	

Special Register Page					
3998	Reboot device	1	-	2	Rebootcode = 0xC5C5
3999	Unlock EEPROM	1	0	0	Unlockcode = 0x5752



(\*) For persistent change of these registers unlock the Mxxx EEPROM first. To unlock write 0x5752 to register 3999. Otherwise changed values will be lost after the next power cycle.  
 (\*\*) EEPROM must be unlocked. Write 0x5752 to register 3999 to unlock the EEPROM.

**Note:** An unlocked EEPROM becomes automatically locked after 2 minutes.

## Detailed description of the implemented Modbus registers

### 4.2 Communication Setup Page

#### 4.2.1 User levels and passwords

After power-up, the Mxxxx is set to user level 0.  
 User levels 1 or 2 can be selected by logging in with password.  
 The password of each access level can be changed by the user.

##### Set user level

To change or check the user level, write or read relative register number 3288:

Register		Register usage		Access user level	
Start	Count	Register 1 / 2	Register 3 / 4	Read	Write
3288	4	User level code	Password	0	0
Example		0x0000, 0x0030	0x00F4, 0x79CE		

The selected user level stays active until next power-down of the sensor. After power-up, user level 0 is active. Invalid login trials are discarded and user level 0 is activated.

##### Change passwords for user levels

To change the password of a user level, write relative register number 3292:

Register		Register usage		Access user level	
Start	Count	Register 1 / 2	Register 3 / 4	Read	Write
3292	4	User level code (hex)	Password (hex)	-	2
Example		0x0000, 0x0030	0x1905, 0x0202		

Invalid user level settings are discarded and no password will be changed.  
 Checking the valid passwords is performed by reading the user level.

#### 4.2.2 Modbus register offset

By default, the Modbus register offset is defined to 0. If necessary, this offset can be changed to any number in the range of -32768...32767.

To change or check the Modbus register offset, write or read absolute register number 0000:

Register		Register usage		Access user level	
Start	Count	Register 1		Read	Write
0000	1	Modbus register offset (signed integer)		0	2
Example		999 (hex-value on register #0: 0x03E7)			

### 4.2.3 Configuration of the RS485 interface

The factory settings of the RS485 interface are mentioned in chapter „Protocol definitions“. The device address, as well as the baud rate and the UART Mode can be adjusted to fit the needs of your installation. **Please verify the new settings by reading them back before powering the unit off. After the next power cycle, the settings will be in effect and if wrong, no further communication will be possible.**

#### Device address

By default, the device address is set to 1. By reading relative register 3098 and 3100, the valid address range can be evaluated. The device address can be changed to any number within this range by writing register 3096:

Register		Register usage		Access user level	
Start	Count	Register 1 / 2		Read	Write
3096	2	Device address (unsigned int)		0	2
3098	2	Min. Address (unsigned int)		0	-
3100	2	Max. Address (unsigned int)		0	-

#### Baudrate

By default, the baudrate is set to 38400. Relative register 3104 and 3106 reports the baudrate limits. The baudrate can be changed to any number within this range by writing register 3102:

Register		Register usage		Access user level	
Start	Count	Register 1 / 2		Read	Write
3102	2	Baudrate code (unsigned int)		0	2
3104	2	Min. Baudrate code (unsigned int)		0	-
3106	2	Max. Baudrate code (unsigned int)		0	-

The Baudrate is represented as a decimal code:

Baudrate	4800	9600	19200	<b>38400</b>	57600	115200
Code	2	3	4	<b>5</b>	6	7

#### Mode

By default, the Mode is set to 8bit data, no parity, 1 stop bit (8,None,1).

Register		Register usage		Access user level	
Start	Count	Register 1		Read	Write
3108	1	UART Mode		0	2

Possible Values:

<b>0x0000</b>	0x0001	0x0002	0x0003	0x0004	0x0005
<b>8,None,1</b>	8,None,2	8,Even,1	8,Even,2	8,Odd,1	8,Odd,2