MIC488 MODBUS-RTU Protocol (v1.23)

Contents

1. Introduction

The MIC488 controller provides RS232 and RS485 serial interfaces which ensure communication with external devices via MODBUS-RTU protocol.
Connection with RS485 port between controllers and MASTER device, especially at bigger distances and speed transmissions (>38400bps, >10m) should be made using twisted pair cable (shielded twisted pair as best solution). It should be noticed to add terminator (120Ω…470Ω resistor, connected between A and B lines) at the beginning and the ending of RS485 bus.

The controller provides communication with Master devices with data rate up to 100 frames per second (fps).

TIP!
Register addresses could be different in newer software versions. Software version should be the same as described in this documentation.

WARNING!
COM1 (RS232) and COM2 (RS485) ports of MIC488 controller are not opt isolated. The potential between controller/s and the other devices on this same bus (common GND) have to be the same. In the other case communication problems could appear or device could be damaged.

2. Transmission parameters and MODBUS functions

Transmission parameters

- Default address: 1
- Default baudrate: **38400 b/s**
- Stop bits: **1**, Parity: **none**
- Timeout: **750µs** (maximum time interval between next bytes in frame)

Provided MODBUS functions

Function No. (hex)	Description
0x01	Read output
0x02	Read input
0x03	Read N registers (for WORD, INT, DINT, REAL)
0x05	Write input
0x06	Write 1 register (for WORD, INT)
0x10	Write N registers (for DINT, REAL)

Description of data types used in MIC488's MODBUS-RTU

Data type	Description	Byte size / registers	Range
BYTE	8 bit (1-byte)	1 / 1	0-255
WORD	16 bit (2-byte)	2 / 1	0…32768
INT	Signed integer (2-byte)	2 / 1	-32768…32767
DINT	Double integer (4-byte)	4 / 2	-2^{31} …$(2^{31}-1)$
REAL	Floating points	4 / 2	$1.18*10^{-38}$…$3.40*10^{38}$, 0, $-3.40*10^{38}$ … $-1.18*10^{-38}$

Values entered into registers are not saved - values will return to default after re-turning on power supply.

3. List of registers

3.1 Write and read inputs, outputs and bit registers

Address	Name	Variable type	Mode (MODBUS function No.)	Description
6000 6000…6007	OUT	BYTE	R (0x01), W (0x05)	Read outputs OUT1…OUT8 Set/reset outputs OUT1…OUT8
5000…5019	IN	BYTE	R (0x02)	Read inputs IN1…IN20
2000	M1_EN	BYTE	W (0x05)	Enable/disable M1 drive (output M1_EN)
2001	M2_EN	BYTE	W (0x05)	Enable/disable M2 drive (output M2_EN)
2002	M3_EN	BYTE	W (0x05)	Enable/disable M3 drive (output M3_EN)
2003	M4_EN	BYTE	W (0x05)	Enable/disable M4 drive (output M4_EN)
2004	M1_STOP	BYTE	W (0x05)	Stop M1 drive (STOP)
2005	M2_STOP	BYTE	W (0x05)	Stop M2 drive (STOP)
2006	M3_STOP	BYTE	W (0x05)	Stop M3 drive (STOP)
2007	M4_STOP	BYTE	W (0x05)	Stop M4 drive (STOP)
2008	M1_JOG_PLUS	BYTE	W (0x05)	JOG+ mode of M1 drive
2009	M1_JOG_MINUS	BYTE	W (0x05)	JOG- mode of M1 drive
2010	M2_JOG_PLUS	BYTE	W (0x05)	JOG+ mode of M2 drive
2011	M2_JOG_MINUS	BYTE	W (0x05)	JOG- mode of M2 drive
2012	M3_JOG_PLUS	BYTE	W (0x05)	JOG+ mode of M3 drive
2013	M3_JOG_MINUS	BYTE	W (0x05)	JOG- mode of M3 drive
2014	M4_JOG_PLUS	BYTE	W (0x05)	JOG+ mode of M4 drive
2015	M4_JOG_MINUS	BYTE	W (0x05)	JOG- mode of M4 drive

NOTE: In HMI devices, write/read functions of bit inputs/outputs are usually labeled as **x0**.

Table example of data for read input IN1 state (address = 5000)

Request (MASTER -> MIC488)		Response (MIC488 -> MASTER)	
Device address	0x01	Device address	0x01
Function	0x02	Function	0x02
Initial HI address	**0x13**	No. of bytes	0x01
Initial LO address	**0x88**	Inputs state	BYTE
No. of HI inputs	0x00	CRC HI	BYTE
No. of LO inputs	**0x08**	CRC LO	BYTE
CRC HI	FD		
CRC LO	62		

Table example of data for output OUT2 settings (address = 6001)

Request (MASTER -> MIC488)		Response (MIC488 -> MASTER)	
Device address	0x01	Device address	0x01
Function	0x05	Function	0x05
Initial HI address	**0x17**	Initial HI address	**0x17**
Initial LO address	**0x71**	Initial LO address	**0x71**
No. of HI inputs	0x00	No. of HI inputs	0x00
No. of LO inputs	**0x00**	No. of LO inputs	**0x00**
CRC HI	98	CRC HI	98
CRC LO	65	CRC LO	65

3.2 Write and read numerical registers

User registers for general purpose

Address	Name	Variable type	Mode (MODBUS function No.)	Description
0…498	USER_REGISTER	WORD DIN REAL	R (0x03), W (0x06) R (0x03), W (0x10) R (0x03), W (0x10)	User registers.

Control registers WORD

Address	Name	Variable type	Mode (MODBUS function No.)	Description
1000	DATA_TYPE	WORD	R (0x03), W (0x06)	Data type transferred by MODBUS: 0 – REAL, 1 - DINT
1001	PROG_BANK_SEL	WORD	R (0x03), W (0x06)	Selection of program bank number to run
1002	POS_BANK_SEL	WORD	R (0x03), W (0x06)	Selection of position bank number to run
1003	PROGRAM_CTRL	WORD	R (0x03), W (0x06)	Program control: 0 – STOP, 1 – START, 2 - PAUSE
1004	AXIS_CTR_MODE	WORD	R (0x03), W (0x06)	Velocity/position control mode for drives: 0 (DIRECT) – velocity/position achieved immediately after write into proper register 1 (TRIGGER) – motion trigger occur after write into AXIS_POS_TRIG (position trigger) or AXIS_VEL_TRIG (velocity trigger) register
1005	AXIS_VEL_ABS_TRIG	WORD	R (0x03), W (0x06)	For mode AXIS_CTR_MODE = 1 Motion trigger from absolute velocity registers Bit 0 – M1, Bit 1 – M2, Bit 2 – M3, Bit 4 – M4
1006	AXIS_VEL_REL_TRIG	WORD	R (0x03), W (0x06)	For mode AXIS_CTR_MODE = 1 Motion trigger from relative velocity registers Bit 0 – M1, Bit 1 – M2, Bit 2 – M3, Bit 4 – M4
1007	AXIS_POS_ABS_TRIG	WORD	R (0x03), W (0x06)	For mode AXIS_CTR_MODE = 1 Motion trigger from absolute position registers Bit 0 – M1, Bit 1 – M2, Bit 2 – M3, Bit 4 – M4
1008	AXIS_POS_REL_TRIG	WORD	R (0x03), W (0x06)	For mode AXIS_CTR_MODE = 1 Motion trigger from relative position registers Bit 0 – M1, Bit 1 – M2, Bit 2 – M3, Bit 4 – M4
1009	M_ALL_STATUS	WORD	R (0x03)	Not available
1010 1011 1012 1013	M1_STATUS M2_STATUS M3_STATUS M4_STATUS	WORD	R (0x03)	Drive state: 0 – drive turned off (EN signal inactive) 1 – drive turned on, no motion (EN signal active) 2 – drive in set velocity mode 3 – drive in motion to set position mode 4 – drive achieved the set position 5 – error of achieving set position (for operation with encoder) 6 – drive in homing mode 8 – drive in position correction mode (for operation with encoder) 9 - drive achieved limit position L while motion towards negative position value (by program or proximity sensor signal KL) 10 - drive achieved limit position R while motion towards positive position value (by program or proximity sensor signal KR)
1014	M_ENABLE	WORD	W (0x06)	Turning on drive's EN output Bit 0 – M1, Bit 1 – M2, Bit 2 – M3, Bit 4 – M4
1015	M_DISABLE	WORD	W (0x06)	Turing off drive's EN output Bit 0 – M1, Bit 1 – M2, Bit 2 – M3, Bit 4 – M4
1016	M_STOP	WORD	W (0x06)	Drive stop

				Bit 0 – M1, Bit 1 – M2, Bit 2 – M3, Bit 4 – M4

NOTE:

- In HMI devices, write/read functions of numerical values are usually labeled as **x4**.
- For registers which control 4 drives at the same time (AXIS_VEL_ABS_TRIG, AXIS_VEL_REL_TRIG, AXIS_POS_ABS_TRIG, AXIS_POS_REL_TRIG, M_ENABLE, M_DISABLE, M_STOP), 4 least significant bits of register define number of drive which have to be controlled: Bit 0 – M1, Bit 1 – M2, Bit 2 – M3, Bit 4 – M4. For instance, to simultaneous stop M1, M2, M3, M4 drives, M_STOP register value should be set as 15 (bits 0,1,3,4 set).

Control registers DINT, REAL

Address	Name	Variable type	Mode (MODBUS function No.)	Description
Current values and ramp parameters				
1017	JOG_SPEED	REAL	W (0x06)	Velocity of manual drives control mode.
1020 1022 1024 1026	M1_POS_ACT M2_POS_ACT M3_POS_ACT M4_POS_ACT	REAL / DINT*	R (0x03), W (0x10)	Read current drive position. Save current position to drive. **Set 0 value causes reset of current drive position.**
1028 1030 1032 1034	M1_VEL_ACT M2_VEL_ACT M3_VEL_ACT M4_VEL_ACT	REAL / DINT*	R (0x03)	Read current drive velocity.
1036 1038 1040 1042	M1_ACC M2_ACC M3_ACC M4_ACC	REAL / DINT*	R (0x03), W (0x10)	Write/read acceleration in position mode and acceleration | braking in velocity mode.
1044 1046 1048 1050	M1_DEC M2_DEC M3_DEC M4_DEC	REAL / DINT*	R (0x03), W (0x10)	Write/read braking in velocity mode.
1052 1054 1056 1058	M1_VMAX M2_VMAX M3_VMAX M4_VMAX	REAL / DINT*	R (0x03), W (0x10)	Write/read maximum velocity for position mode.
Set in motion. IMPORTANT: When register AXIS_CTR_MODE (1004) = 0 (DIRECT mode) motion occur immediately after write into proper register. When AXIS_CTR_MODE (1004) = 1 (TRIGGER mode) motion occur until when write into trigger register (properly 1005..1008 registers).				
1060 1062 1064 1066	M1_HOME M2_HOME M3_HOME M4_HOME	REAL / DINT*	W (0x10)	Homing execution. Value of register determines homing velocity.
1068 1070 1072 1074	M1_VEL_ABS M2_VEL_ABS M3_VEL_ABS M4_VEL_ABS	REAL / DINT*	W (0x10)	Set absolute velocity (drive velocity will be the same as entered value).
1076 1078 1080 1082	M1_VEL_REL M2_VEL_REL M3_VEL_REL M4_VEL_REL	REAL / DINT*	W (0x10)	Set relative velocity (drive velocity will be equal to current and entered velocity).
1084 1086 1088 1090	M1_POS_ABS M2_POS_ABS M3_POS_ABS M4_POS_ABS	REAL / DINT*	W (0x10)	Set absolute position (drive motion occur until achieved set position).
1092 1094 1096 1098	M1_POS_REL M2_POS_REL M3_POS_REL M4_POS_REL	REAL / DINT*	W (0x10)	Set relative position (drive motion occur until position will be equal to current + set value).
Registers for encoders				
1100 1102 1104 1106	ENC1_IMP ENC2_IMP ENC3_IMP ENC4_IMP	DINT	R (0x03), W (0x10)	Counter value of pulses from encoder.

1108 1110 1112 1114	ENC1_XPOS ENC2_XPOS ENC3_XPOS ENC4_XPOS	REAL / DINT*	R (0x03), W (0x10)	Counter value of pulses from encoder converted into drive rotation.
1120 1122 1124 1126	M1_POSLIM_L M2_POSLIM_L M3_POSLIM_L M4_POSLIM_L	REAL / DINT*	R (0x03), W (0x10)	Program position limitation negative shift (L)
1128 1130 1132 1134	M1_POSLIM_R M2_POSLIM_R M3_POSLIM_R M4_POSLIM_R	REAL / DINT*	R (0x03), W (0x10)	Program position limitation positive shift (R)

The MIC488 is addressing registers starts with 0. For MASTER devices which addressing starts with 1, register values should be entered with shift by 1 e.g. JOG_SPEED = 1017 + 1 = 1018

* Variable type depends on data type settings in DATA_TYPE (1000) register. REAL data type values are default, scaled on proper motion units.

After writing 1 value into ***DATA_TYPE*** register, controller assigns and returns values in pulses (DINT type) which are not scaled. Values in pulses are equivalent to number of steps generated by controller.

Table example of settings for device M1 absolute velocity. Function: 0x10, Register address: 1068 (M1_VEL_ABS)

Write (MASTER -> MIC488)		Response (MIC488 -> MASTER)	
Device address	0x01	Device address	0x01
Function	0x10	Function	0x10
Register Hi address	**0x04**	Initial HI address	0x04
Register Lo address	**0x2C**	Initial LO address	**0x2C**
No. of HI registers	0x00	No. of HI registers	0x00
No of LO registers	0x02	No. of LO registers	**0x02**
No. of bytes	0x04	CRC	16 bit
Register 0x06 HI	REAL/DINT* (Byte 1)		
Register 0x06 LO	REAL/DINT* (Byte 0)		
Register 0x06 +1 HI	REAL/DINT* (Byte 3)		
Register 0x06 +1 LO	REAL/DINT* (Byte 2)		
CRC	**16 bit**		

Table example with read current position of M1 drive. Function: 0x03, Register address: 1020 (M1_POS_ACT)

Request (MASTER -> MIC488)		Response (MIC488 -> MASTER)	
Device address	0x01	Device address	0x01
Function	0x03	Function	0x03
Register HI address	**0x03**	No. of bytes	**0x04**
Register LO address	**0xFC**	Register 0x03 HI	REAL/DINT* (Byte 1)
No. of HI registers	0x00	Register 0xFC LO	REAL/DINT* (Byte 0)
No of LO registers	0x02	Register 0x03+1 HI	REAL/DINT* (Byte 3)
CRC HI	0x04	Register 0xFC+1 LO	REAL/DINT* (Byte 2)
CRC LO	0x7F	CRC	16 bit

All 4-bytes types: **DINT**, **DWORD**, **REAL** are always included in **two registers**. Furthermore for DINT, first register include least significant part while second register – most significant part. For example, to read current position of M1 drive, 1020 and 1021 registers should be read and then make proper conversion (if there is not proper Modbus function in MASTER driver).

2 registers conversion (4 bytes) to 32-byte type (DINT, DWORD, FLOAT).

RegisterX HI <-> Byte1
RegisterX LO <-> Byte0
RegisterX+1 HI <-> Byte3
RegisterX+1 LO <-> Byte2

32_bit_integer = **Byte3**<<24 + **Byte2**<<16 + **Byte1**<<8 + **Byte0,**

or
32_bit_integer = RegisterX + Register(X + 1)<<16

4. Modbus control description

4.1 Program control in controller memory

Program selection from memory of the controller which have to be controlled (run, stop, holdup) is possible with ***PROG_BANK_SEL*** (1001) register. The corresponding value of selected program (0…7 value) need to be write into register.

Using ***POS_BANK_SEL*** (1002) register, bank with positions could be selected and then used in program (when program uses positions from position table).

PROGRAM_CTRL (1003) register controls program running:

- Write 1 value causes run program
- Write 2 value causes hold current program
- Write 0 value causes stop program

If running program comes from empty program or position bank, controller will signals error by lighting red ERR diode.

4.2 Turning on/off and stopping drive

Write into control registers of selected drives at the same time
Registers:

- ***AXIS_VEL_ABS_TRIG*** (1005)
- ***AXIS_VEL_REL_TRIG*** (1006),
- ***AXIS_POS_ABS_TRIG*** (1007),
- ***AXIS_POS_REL_TRIG*** (1008),
- ***M_ENABLE*** (1014)
- ***M_DISABLE*** (1015)
- ***M_STOP*** (1016)

ensure selected drives controlling (e.g. turning on/off selected drives etc., motion trigger for TRIGGER mode) at the same time. The corresponding bit setting value of selected drives should be write into selected register, where least significant bit 0 – M1 drive, bit 1 – M2, bit 2 – M3, bit 3 – M4.
Table below includes list of all bit combination. The plus „+" symbol means that drive is selected.

selected drive				value wrote into register
M1	M2	M3	M4	
-	-	-	-	0
+	-	-	-	1
-	+	-	-	2
+	+	-	-	3
-	-	+	-	4
+	-	+	-	5
-	+	+	-	6
+	+	+	-	7
-	-	-	+	8
+	-	-	+	9
-	+	-	+	10
+	+	-	+	11

-	-	+	+	12
+	-	+	+	13
-	+	+	+	14
+	+	+	+	15

Output ENABLE control

- Independent control of each output with ***M1_EN***…***M4_EN*** (2000…2003) bit registers. One register controls only one EN output. Write 0xFF00 value into register turn EN output on, 0x00 value turn output off.
- Simultaneous control of selected outputs with ***M_ENABLE*** (1014) register which turns selected EN outputs on and ***M_DISABLE*** (1015) which turns them off.

Drive stopping

- Independent stop of each drive with ***M1_STOP***…***M4_STOP*** (2004…2007) bit registers. Write 0xFF00 value into register stops the drive.
- Simultaneous stop of selected drives with ***M_STOP*** (1016) register. For instance, when 15 value is write into, the drives are stopped.

4.3 Motion parameters setting (ramp)

Motion parameters setting which control ramp is released using registers:

- ***M1_ACC***…***M4_ACC*** (1036…1042) – write/read acceleration in position mode and acceleration | braking in velocity mode
- ***M1_DEC***…***M4_DEC*** (1044…1050) – write/read braking in velocity mode
- ***M1_VMAX***…***M4_VMAX*** (1052…1058) – write/read maximal velocity for position mode

Motion parameters should be entered as positive values, greater than 0!

4.4 Drive homing

Drives homing uses ***M1_HOME***…***M4_HOME*** (1060…1066) registers. Entered value into register defines homing velocity. Homing occur immediately after write into register. When homing is end, drive position is automatically reset.

Homing to **the limit switch** (KL) is release by write velocity as **negative value**.

4.5 Velocity and position setting

Absolute and relative velocity/position

Velocity and position could be set **absolutely** (e.g. drive achieve velocity/position value the same as entered register value) or **relatively** (drive will increase or decrease velocity/position by entered register value). Registers which contain **ABS** in their names set absolute values, while **REL** – relatively values.

Motion registers

- ***M1_VEL_ABS***… ***M4_VEL_ABS*** (1068…1074) – set absolute velocities
- ***M1_VEL_REL*** … ***M1_VEL_REL*** (1076…1082) – set relative velocities

- ***M1_POS_ABS*** … ***M4_POS_ABS*** (1084…1090) – set absolute positions
- ***M1_POS_REL*** … ***M1_POS_REL*** (1092…1098) – set relative positions

Velocity / position setting mode

Mode is changing by write 1 value into ***AXIS_CTR_MODE*** (1004) register (TRIGGER mode) or 0 value (DIRECT mode).

Direct mode (DIRECT), (when register AXIS_CTR_MODE = 0)
The controller performs in direct speed/position control mode (**DIRECT** mode) by default. The controller execute motion immediately after write into proper motion register.

Trigger mode (TRIGGER), (when register AXIS_CTR_MODE = 1)
In this mode writing into motion registers do not causes release signal motion to drive by controller. Motion is released after writing equivalent value to trigger registers of selected drives numbers:

- AXIS_VEL_ABS_TRIG (1005) – for release absolute velocities
- AXIS_VEL_REL_TRIG (1006) – for release relative velocities
- AXIS_POS_ABS_TRIG (1007) – for release absolute positions
- AXIS_POS_REL_TRIG (1008) – for release relative positions

It is useful when a few drives should operate at the same time to perform synchronous motion.

For instance, for M1, M2 and M3 drives, position should be set properly on 10, 15 and 20:

1) Set TRIGGER mode by write into ***AXIS_CTR_MODE*** (1004) register 1 value (0 value to return to DIRECT mode),
2) Write into ***M1_POS_ABS***, ***M2_POS_ABS*** and ***M3_POS_ABS*** registers properly 10, 15 and 20 values,
3) Write into trigger register ***AXIS_POS_ABS_TRIG*** (1007) 7 value (which is equivalent to 0111 bit value, because motion is triggered for M1, M2 and M3 drives)
   IMPORTANT: EN signals have to be turned on before, if they control drives work.

4.6 Position reset, position and velocity read

Read current velocity and position

Current velocity is available in ***M1_VEL_ACT***…. ***M4_VEL_ACT*** registers.
Current position in available in ***M1_POS_ACT***…. ***M4_POS_ACT*** registers.

Current position reset

In order to reset current position for selected drive, 0 value need to be write into proper ***M1_POS_ACT***… ***M4_POS_ACT*** register. Write non-zero value causes overwrite current position by entered value.

4.7 Drive status

Drive operation monitoring is possible with ***M1_STATUS*** (1010)…***M4_STATUS*** (1013) state registers.

MX_STATUS register value	Description
0	Drive turned off (EN signal inactive)
1	Drive turned on, no motion (EN signal active)
2	Drive in set velocity mode
3	Drive in motion to set position mode
4	Drive achieved the set position
5	Error of achieving set position (for operation with encoder)

6	Drive in homing mode
7	-
8	Drive in position correction mode (for operation with encoder)
9	Drive achieved limit position L while motion towards negative position value (by program or proximity sensor signal KL)
10	Drive achieved limit position R while motion towards positive position value (by program or proximity sensor signal KR)

Drive status could be used i.e. to define: is drive achieved set position before next position setting.

4.8 JOG control mode

The JOG mode could be used for manual drive position control with e.g. HMI touch panel. Motion of the drive occur by setting proper JOG register. Motion velocity in JOG mode is defined in ***JOG_SPEED*** (1017) register (REAL type).

Motion for each drive is triggered by bit registers:

- ***M1_JOG_PLUS*** (2008) - motion towards positive of M1 drive
- ***M1_JOG_MINUS*** (2009) – motion towards negative M1 drive
- ***M2_JOG_PLUS*** (2010) - motion towards positive M2 drive
- ***M2_JOG_MINUS*** (2011) – motion towards negative M2 drive
- ***M3_JOG_PLUS*** (2012) - motion towards positive M3 drive
- ***M3_JOG_MINUS*** (2013) - motion towards negative M3 drive
- ***M4_JOG_PLUS*** (2014) - motion towards positive M4 drive
- ***M4_JOG_MINUS*** (2015) - motion towards negative M4 drive

4.9 Read / set parameters in pulses (DINT data type)

The controller ensure control of drives without units conversion (configurable in controller) with **DINT** type values. It is useful when direct setting of position in pulses is required or master driver do not support register conversion into floating point **REAL** integer type.

To change data type to DINT, 1 value should be write into ***DATA_TYPE*** (1000) register. After that, all registers, whose type is defined as **REAL / DINT*** will assign/return DINT type values.

Example

Drive: stepper motor 200 imp./rev. with 1/64 step resolution.
Units calculation: 200 * 64 = 12800 pulses / motor revolution.

Velocity setting 2,5 rev./sec for M1 drive:
Write into ***M1_VEL_ABS*** (1068) register 2,5 * 12800 = 32000 [imp]

Read current position of M1 drive:
Read **M1_POS_ACT** (1020) register, which contains (for example) 57600 value.
Current position of drive in [rev] = 57600/12800 = 4,5 [rev]

4.10 User registers

The MIC488 contains 500 general-purpose registers (**0-499 addresses**). User can store in these registers values, which could be read or write by current controller program. Values type as INT, DINT and REAL could be write into registers. In order to write DINT and REAL value type, which are always included in two adjacent registers, they should be write into even addresses.

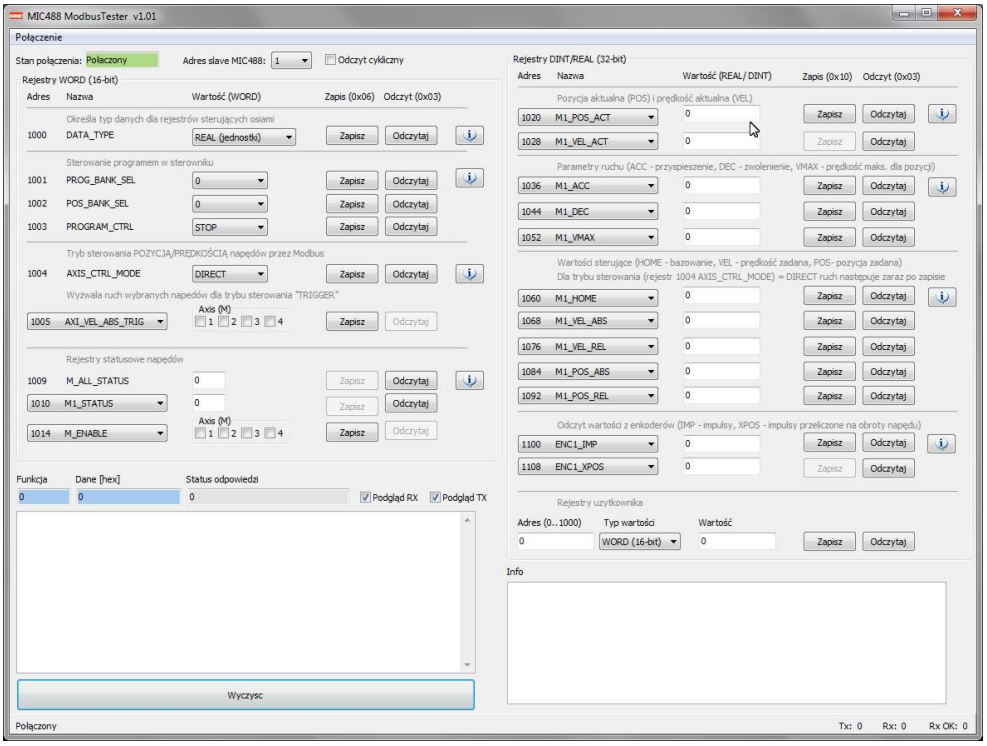
Register	0	1	2	3	…	498	499

	INT 0	INT 1	INT 2	INT 3		INT 498	INT 499
	DINT 0		DIN 2			DIN 498	
	REAL 0		REAL 2			REAL 498	

In order to reference to user registers in WBC language, commands which should be used are:
$I**X** – reference to register with INT type value
$D**X** – reference to register with DINT type value (X only as even value)
$R**X** – reference to register with REAL type value (X only as even value)
where X is register address 0..499

5. MIC488 ModbusTester program

MIC488 ModbusTester application provides test each controller registers and preview Modbus protocol frames. Communication with application is released by converter USB<->RS232/RS485 with FTDI chipset.



Program may not include all registers available in device.

6. Documentation revision

v1.01:
- initial version

v1.23:
- register of limit positions MX_POSLIM_L/ MX_POSLIM_R added