

Table of Contents

Chapter I	ALLEN-BRADLEY DF1 FULL DUPLEX PROTOCOL FOR PLC3	1
1	Introduction	1
2	Numeric gates	1
3	Digital gates	3
4	String gates	4
5	Note	4
6	Configuration	5
Chapter II	ALLEN-BRADLEY DF1 FULL DUPLEX PROTOCOL FOR PLC5	5
1	Introduction	5
2	Numeric gates	6
3	Digital gates	8
4	String gates	10
5	Note	10
6	Configuration	12
Chapter III	ALLEN-BRADLEY DF1 FULL DUPLEX PROTOCOL FOR SLC500 / MicroLogix	12
1	Introduction	12
2	Numeric gates	13
3	Digital gates	14
4	String gates	16
5	Note	16
6	Configuration	17
Chapter IV	ALLEN-BRADLEY Ethernet	17
1	Introduction	17
2	Numeric gates	17
3	Digital gates	19
4	String gates	21
5	Note	21
6	Configuration	22
Chapter V	AVEBus	23
1	Introduction	23
2	Configuration	25

Chapter VI	BACnet	25
	1 Introduction	25
	2 Numeric gates	26
	3 Digital gates	29
	4 String gates	29
	5 Configuration	29
Chapter VII	DATA STREAM (CR Magnetics)	31
	1 Introduction	31
	2 Configuration	31
Chapter VIII	EUROTHERM BISYNCH ASCII	31
	1 Introduction	31
	2 Numeric gates	32
	3 Digital gates	32
	4 String gates	32
	5 Configuration	33
Chapter IX	EV2001 (Bilanciai)	33
	1 Introduction	33
	2 Numeric gates	33
	3 Digital gates	34
	4 String gates	34
	5 Configuration	34
Chapter X	GEFRAN - CENCAL	34
	1 Introduction	34
	2 Numeric gates	34
	3 Digital gates	35
	4 String gates	35
	5 Configuration	35
Chapter XI	IDEC IZUMI FA	35
	1 Introduction	35
	2 Numeric gates	35
	3 Digital gates	36
	4 String gates	36
	5 Configuration	37
Chapter XII	KLOCKNER MOELLER SUCOM - A	37

	1	Introduction	37
	2	Numeric gates	37
	3	Digital gates	39
	4	String gates	39
	5	Configuration	40
Chapter XIII		KLOCKNER MOELLER SUCOM - A per PS4	40
	1	Introduction	40
	2	Numeric gates	40
	3	Digital gates	41
	4	String gates	41
	5	Configuration	42
Chapter XIV		KNX (Falcon Library)	42
	1	Introduction	42
	2	Numeric gates	43
	3	Digital gates	50
	4	String gates	52
	5	Configuration	55
Chapter XV		M-BUS (METER-BUS)	56
	1	Introduction	56
	2	Numeric gates	57
	3	Digital gates	76
	4	String gates	76
	5	Configuration	80
Chapter XVI		mitsubishi Computer Link FX	81
	1	Introduction	81
	2	Numeric gates	81
	3	Digital gates	83
	4	String gates	84
	5	Configuration	84
Chapter XVII		MITSUBISHI FR-CU03	85
	1	Introduction	85
	2	Numeric gates	85
	3	Digital gates	86
	4	String gates	86
	5	Configuration	86

Chapter XVIII	MITSUBISHI MC PROTOCOL (1E frame)	87
1	Introduction	87
2	Numeric gates	87
3	Digital gates	91
4	String gates	92
5	Configuration	93
Chapter XIX	MODBUS TCP - MODBUS RTU - MODBUS ASCII	94
1	Introduction	94
2	Numeric gates	94
3	Digital gates	97
4	String gates	97
5	Configuration	101
Chapter XX	ODBC Client	103
1	Introduction	103
2	Numeric gates	104
3	Digital gates	105
4	String gates	106
5	Configuration	107
Chapter XXI	OMRON FINS	110
1	Introduction	110
2	Numeric gates	111
3	Digital gates	113
4	String gates	114
5	Configuration	115
Chapter XXII	OMRON FINS in Host Link Protocol	116
1	Introduction	116
2	Numeric gates	116
3	Digital gates	118
4	String gates	119
5	Configuration	120
Chapter XXIII	OMRON SYSMAC	121
1	Introduction	121
2	Numeric gates	121
3	Digital gates	122

	4	String gates	123
	5	Configuration	123
Chapter XXIV		OPC Client	123
	1	Introduction	123
	2	Numeric gates	125
	3	Digital gates	125
	4	String gate	125
	5	Configuration	125
Chapter XXV		RED LION PAXI-1/8 DIN COUNTER/RATE METER	126
	1	Introduction	126
	2	Numeric gates	126
	3	Digital gates	126
	4	String gates	126
	5	Configuration	126
Chapter XXVI		SAIA P800	127
	1	Introduction	127
	2	Numeric gates	127
	3	Digital gates	128
	4	String gates	128
	5	Configuration	128
Chapter XXVII		SAIA S-BUS	129
	1	Introduction	129
	2	Numeric gates	129
	3	Digital gates	130
	4	String gates	130
	5	Configuration	130
Chapter XXVIII		SIEMENS MPI	131
	1	Introduction	131
	2	Numeric gates	132
	3	Digital gates	135
	4	String gates	136
	5	Configuration	137
Chapter XXIX		TUTONDO	140
	1	Introduction	140

	2	Configuration	142
Chapter XXX		PANASONIC (MATSUSHITA) MEWTOCOL - COM	142
	1	Introduction	142
	2	Numeric gates	142
	3	Digital gates	144
	4	String gates	145
	5	Configuration	145
Chapter XXXI		PPI S7 200 (PPI Adapter)	146
	1	Introduction	146
	2	Numeric gates	147
	3	Digital gates	148
	4	String gates	149
	5	Configuration	149
Chapter XXXII		Raw ASCII Output	150
	1	Introduction	150
	2	Numeric gates	150
	3	Digital gates	150
	4	String gates	150
	5	Configuration	150
Chapter XXXIII		Winlog TCP Protocol	151
	1	Introduction	151
	2	Numeric gates	152
	3	Digital gates	153
	4	String gates	155
	5	Configuration	156

1 ALLEN-BRADLEY DF1 FULL DUPLEX PROTOCOL FOR PLC3

1.1 Introduction

The PC is seen as a node of the network Data Highway Plus and can communicate with all PLC 3 of the network

Contact the provider of the PLC to select the proper interface board (ex. 1770-KF2 Interface Module)
The communication between the PC and the Interface Board is through the RS232 serial channel of the PC.

1.2 Numeric gates

This specification applies to gates which belong to files: **Integer, Float, BCD, Bit, ASCII, Long, Status:**

General format:

T.FFF.EEEE

where

T : file identifier of the gate

FFF : file number

EEEE : word.

Address	Description	Type	FFF	EEEE	Read gate	Write gate	Read block
T.FFF.EEEE	Integer file	N	0...999	0...9999	Yes	Yes	Yes
T.FFF.EEEE	Float file	F	0...999	0...9998 must be an even number	Yes	Yes	Yes
T.FFF.EEEE	BCD file	D	0...999	0...9999	Yes	Yes	Yes
T.FFF.EEEE	Bit file	B	0...999	0...9999	Yes	Yes	Yes
T.FFF.EEEE	ASCII file	A	0...999	0...9999	Yes	Yes	Yes
T.FFF.EEEE	Long file	L	0...999	0...9998 must be an even number	Yes	Yes	Yes
T.FFF.EEEE	Status file	S	0...999	0...9999	Yes	Yes	Yes

Data format is DWORD (4 bytes) for Float and Long gates and WORD (2 bytes) for other gates.

Example:

S.001.0003 : Status - File 001 – Number Word 0003.

B.121.0303 : Bit - File 121 – Number Word 0303.

N.100.0120 : Integer - File 100 – Number Word 0120.

D.050.0020 : BCD - File 050 – Number Word 0020.

A.007.0023 : ASCII - File 007 – Number Word 0023.

This specification applies to gates which belong to file: **Output, Input**

General format:

T.O000

where

T : file type

O000 : word number (octal)

Address	Description	Type	O000	Read gate	Write gate	Read block
T.O000	Output file	O	0...7777 octal	Yes	Yes	Yes
T.O000	Input file	I	0...7777 octal	Yes	Yes	Yes

Data format for these gates is WORD (2 Bytes).

Example:

O.0017 : Output – Word number 0017 (octal).

I.1234 : Input – Word number - 1234 (octal).

This specification applies to gates which belong to file: **Timer, Counter**

General format:

T.SSSS.Q

where

T : file type

SSSS : Address of structure Timer / Counter .

Q : Timer / Counter Sub Element (0,1,2).

Address	Description	Type	SSSS	Q	Read gate	Write gate	Read block
T.SSSS.Q	Timer file	T	0...9999	0...2	Yes	Yes	Yes
T.SSSS.Q	Counter file	C	0...9999	0...2	Yes	Yes	Yes

Data format for these gates is WORD (2 Bytes).

Example:

T.0015.0 : Timer - Structure 0015 - Sub-Element 0

C.0007.1 : Counter - Structure 0007 - Element 120 – Sub-Element 1

Blocks of numeric gates

In case of gates which belong to file **Status, Bit, Integer, BCD, ASCII**, the block must be made of gates which belong to the same type and have the same file number and a sequential word number. The maximum block length is 119 gates.

Numeric gates block	
	T.FFF.EEE
	T.FFF.EEE+1
	T.FFF.EEE+2
	T.FFF.EEE+3
	T.FFF.EEE+4

In case of gates which belong to file **Float** and **Long**, the block must be made of gates which belong to the same type and have the same file number and a sequential even word number. The maximum block length is 59 gates.

Numeric gates block	
	T.FFF.EEE
	T.FFF.EEE+2
	T.FFF.EEE+4
	T.FFF.EEE+6
	T.FFF.EEE+8

In case of gates which belong to file **Timer** and **Counter**, the block must be made of gates which belong to the same structure and have a sequential sub-element number. The maximum block length is 3 gates.

Numeric gates block	
	T.SSSS.Q.00
	T.SSSS.Q.01
	T.SSSS.Q.02

1.3 Digital gates

In case of gates which belong to file **Status**, **Bit**, and **Integer** the address is :

T.FFF.EEEE.BB

where

T : file identifier

FFF : file number

EEEE: element number

BB : bit number (octal value from 00 to 17)

Address	Description	T	FFF	EEEE	BB	Gate read	Gate write	Block read
T.FFF.EEE.B B	Status file	S	0...999	0...9999	00...17 octal	Yes	Yes	Yes
T.FFF.EEE.B B	Bit file	B	0...999	0...9999	00...17 octal	Yes	Yes	Yes
T.FFF.EEE.B B	Integer file	N	0...999	0...9999	00...17 octal	Yes	Yes	Yes

Example:

S.001.0003.00 : Status - File 001 - Element 0003 – Bit 00.

B.121.0303.17 : Bit - File 121 - Element 0303 – Bit 17.

N.100.0120.12: Integer - File 100 - Element 0120 – Bit 12.

In case of gates which belong to file **Output** and **Input** the address is:

T.O000.BB

where

T : file identifier

O000: word number (octal value)

BB : bit number (octal value from 00 to 17)

Address	Description	T	O000	BB	Gate read	Gate write	Block read
T.O000.BB	Output file	O	0...7777 octal	00...17 octal	Yes	Yes	Yes
T.O000.BB	Input file	I	0...7777 octal	00...17 octal	Yes	Yes	Yes

Fields **O000** and **BB** must be defined with octal numbers

Example:

O.0001.17 : Output – Word 0001 (octal) – Bit 17 (octal).

I.0001.10 : Input - Word 001 (octal) – Bit 10 (octal).

In case of gates which belong to file **Timer** e **Counter** the address is:

T.SSSS.Q

where

T : file identifier

SSSS : structure address Timer / Counter .

Q : sub-element (only 0).

BB : bit number (octal value from 00 to 17)

Address	Description	T	SSSS	Q	BB	Gate read	Gate write	Block read
T.SSSS.Q.B B	Timer file	T	0...9999	0	00...17 octal	Yes	Yes	Yes
T.SSSS.Q.B B	Counter file	C	0...9999	0	00...17 octal	Yes	Yes	Yes

Example:

T.0015.0.01 : Timer - Structure 0015 - Sub-Element 0 – Bit 01 (octal).

C.0007.0.17 : Counter – Structure 0007 - Sub-Element 0 – Bit 17 (octal).

Blocks of digital gates

In case of gates which belong to file **Status**, **Bit**, **Integer**, the block must be made of gates which

belong to the same type and have the same file number and the same or a sequential word number. In case of gates which belong to file **Output** and **Input**, the block must be made of gates which belong to the same type and have the same or a sequential word number.

In case of gates which belong to file **Timer** and **Counter**, the block must be made of gates which belong to the same type, structure and sub-element.

The length of a block depends from the block composition. A block made of 16 gates belonging to the same word has a smaller dimension than a block made of 16 gates belonging to different words. The right dimensioning of a block is controlled by the driver itself during the first scanning; if the block length is too high, an error message will be displayed.

Example of valid block	Example of valid block	Example of NOT valid block	Example of NOT valid block
B.012.0003.01	T.0006.0.00	B.012.0003.01	T.0006.0.00
B.012.0003.03	T.0006.0.01	B.012.0003.03	T.0006.0.01
B.012.0004.14	T.0006.0.02	B.012.0004.14	T.0006.0.02
B.012.0005.01	T.0006.0.03	N.012.0005.01	T.0007.0.16
B.012.0006.01	T.0006.0.05	N.012.0006.01	T.0007.0.05
B.012.0006.07	T.0006.0.12	B.012.0006.07	T.0007.0.12
B.012.0006.10	T.0006.0.13	B.012.0006.10	T.0008.0.06
B.012.0007.00	T.0006.0.14	B.012.0007.00	T.0008.0.07

1.4 String gates

String gates are not allowed in this protocol.

1.5 Note

TIMER

Timer Sub Element	Description
00	Control bits
01	Preset
02	Acc

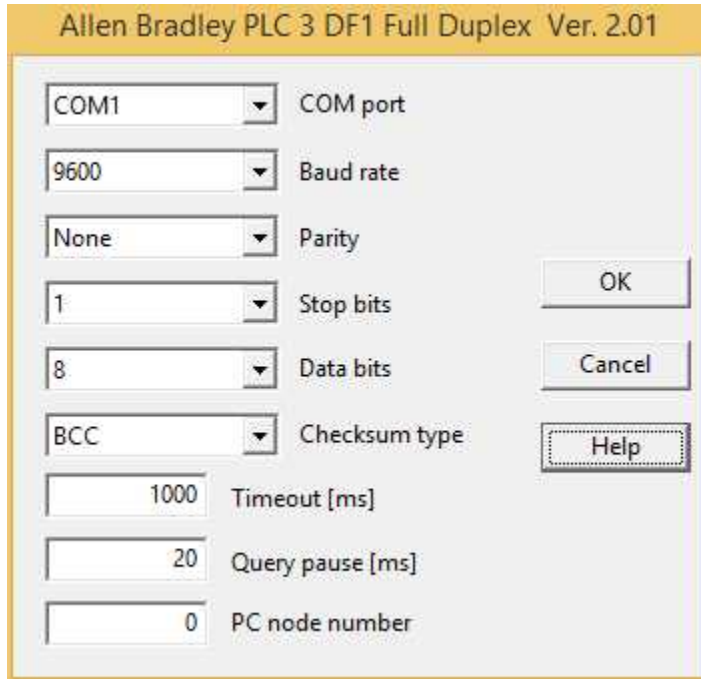
Timer Sub Element Detail	Bit number	Description
00	13	Done
00	14	Timing
00	15	Enable

COUNTER

Timer Sub Element	Description
00	Control bits
01	Preset
02	Accumulator

Timer Sub Element Detail	Bit Number	Description
00	11	Underflow
00	12	Overflow
00	13	Done
00	14	Count down
00	15	Count up

1.6 Configuration



Allen Bradley PLC 3 DF1 Full Duplex Ver. 2.01

COM1	COM port
9600	Baud rate
None	Parity
1	Stop bits
8	Data bits
BCC	Checksum type
1000	Timeout [ms]
20	Query pause [ms]
0	PC node number

OK
Cancel
Help

Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages.
- **Checksum type :** BCC or CRC16.
- **PC node number :** node number of the PC

2 ALLEN-BRADLEY DF1 FULL DUPLEX PROTOCOL FOR PLC5

2.1 Introduction

This protocol applies to series 5 PLC's (excluded PLC 5-250).

The PC is seen as a node of the network Data Highway Plus and can communicate with all PLC 5 of the network

Contact the provider of the PLC to select the proper interface board (ex. 1770-KF2 Interface Module)
The communication between the PC and the Interface Board is through the RS232 serial channel of the PC.

2.2 Numeric gates

General format:

T.FFF.EEE.SS or **T.FFF.EEE**

where:

T : file identifier

FFF : file number

EEE : element number

SS : sub-element number

Specification which applies to gates: **Status, Bit, Integer, BCD, ASCII**:

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block
T.FFF.EEE	Status file	S	0...999	0...999		Yes	Yes	Yes
T.FFF.EEE	Bit file	B	0...999	0...999		Yes	Yes	Yes
T.FFF.EEE	Integer file	N	0...999	0...999		Yes	Yes	Yes
T.FFF.EEE	BCD file	D	0...999	0...999		Yes	Yes	Yes
T.FFF.EEE	ASCII file	A	0...999	0...999		Yes	Yes	Yes

The format of data is: WORD (2 Byte)

Example:

S.001.003 : Status - File 001 - Element 003.

B.121.303 : Bit - File 121 - Element 303.

N.100.120 : Integer - File 100 - Element 120.

D.050.020 : BCD - File 050 - Element 020.

A.007.023 : ASCII - File 007 - Element 023.

Specification which applies to gates which belong to file **Output** and **Input** :

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block
T.FFF.EEE	Output file	O	0...999	0...277 octal		Yes	Yes	Yes
T.FFF.EEE	Input file	I	0...999	0...277 octal		Yes	Yes	Yes

The format of data is: WORD (2 Byte)

The field **EEE** must be specified with an octal number

Example:

O.001.010 : Output - File 001 - Element 008 (010 octal).

I.001.021 : Input - File 001 - Element 017 (021 octal).

Specification which applies to gates which belong to file **Float** :

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block
T.FFF.EEE	Float file	F	0...999	0...998 must be an even number		Yes	Yes	Yes

The format of data is FLOAT (4 Byte).

The field **EEE** must be an even number

Example:

F.015.110 : Float - File 015 - Element 110.

Specification which applies to gates which belong to file **Timer, Counter** and **Control** :

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block
T.FFF.EEE.SS	Timer file	T	0...999	0...999	0...2	Yes	Yes	Yes
T.FFF.EEE.SS	Counter file	C	0...999	0...999	0...2	Yes	Yes	Yes
T.FFF.EEE.SS	Control file	R	0...999	0...999	0...2	Yes	Yes	Yes

The format of data is: WORD (2 Byte)

Example:

T.015.110.00 : Timer - File 015 - Element 110 – Sub-Element 00

C.007.120.01 : Counter - File 007 - Element 120 – Sub-Element 01

R.050.011.02 : Control - File 050 - Element 011 – Sub-Element 02

Specification which applies to gates which belong to file **PID** :

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block
T.FFF.EEE.SS	PID file	P	0...999	0...999	0...48	Si	Si	Si

If the sub-element (SS) is 00 or 01, then the data format is WORD (2 Byte); otherwise is FLOAT (4 byte).

If the sub-element (**SS**) is greater than 1, then it must be an even number.

Example:

P.015.110.00 : PID - File 015 - Element 110 – Sub-Element 00

P.005.010.46 : PID - File 005 – Element 010 – Sub-Element 46

Blocks of numeric gates

In case of gates which belong to file **Status, Bit, Integer, BCD, ASCII, Output, Input** the block must be made of gates which belong to the same type and have the same file number and a sequential word number.

The maximum block length is 119 gates.

Numeric gates block	
T.FFF.EEE	
T.FFF.EEE+1	
T.FFF.EEE+2	
T.FFF.EEE+3	
T.FFF.EEE+4	

In case of gates which belong to file **Float**, the block must be made of gates which belong to the same type and have the same file number and a sequential even word number.

The maximum block length is 59 gates.

Numeric gates block	
T.FFF.EEE	
T.FFF.EEE+2	
T.FFF.EEE+4	
T.FFF.EEE+6	
T.FFF.EEE+8	

In case of gates which belong to file **Timer, Counter** and **Control** the block must be made of gates which belong to the same file type, file number and element and have a sequential sub-element number.

The maximum block length is 3 gates.

Numeric gates block	
T.SSSS.Q.00	
T.SSSS.Q.01	

T.SSSS.Q.02

In case of gates which belong to file **PID**, the block must be made of gates which belong to the same file type, file number and element and have a sequential sub-element number.

The same block cannot be made of gates which have sub-element number less or equal than 1 and gates which have sub-element number greater than 1.

There are two possible blocks of gates PID:

1st block type : gates PID with sub-element equal to 1	
T.FFF.EEE.00	
TFFF.EEE.01	
2nd block type : gates PID with sub-element greater than 1	
T.FFF.EEE.SS	
T.FFF.EEE.SS+2	
T.FFF.EEE.SS+4	
T.FFF.EEE.SS+6	

In the second block the sub-element is always even because the gate is FLOAT (4 bytes).

2.3 Digital gates

The address of a digital gate is as follows:

T.FFF.EEE.SS.BB or **T.FFF.EEE.BB**

where

T : file identifier

FFF : file number

EEE : element number

SS : sub-element

BB : bit number

Specification which applies to gates which belong to file **Status, Bit, and Integer** :

Address	Description	T	FFF	EEEE	SS	BB	Gate read	Gate write	Block read
T.FFF.EEE.BB	Status file	S	0...999	0...999		00...15	Yes	Yes	Yes
T.FFF.EEE.BB	Bit file	B	0...999	0...999		00...15	Yes	Yes	Yes
T.FFF.EEE.BB	Integer file	N	0...999	0...999		00...15	Yes	Yes	Yes

Example:

S.001.003.00 : Status - File 001 - Element 003 – Bit 00.

B.121.303.15 : Bit - File 121 - Element 303 – Bit 15.

N.100.120.12: Integer - File 100 - Element 120 – Bit 12.

Specification which applies to gates which belong to file **Output and Input** :

Address	Description	T	FFF	EEE	SS	BB	Gate read	Gate write	Block read
T.FFF.EEE.BB	Output file	O	0...999	0...277 octal		00...17 octal	Yes	Yes	Yes
T.FFF.EEE.BB	Input file	I	0...999	0...277 octal		00...17 octal	Yes	Yes	Yes

Fields **EEE** e **BB** must be octal numbers

Example:

O.001.010.17 : Output - File 001 - Element 008 (010 octal) – Bit 15 (17 octal).

I.001.021.10 : Input - File 001 - Element 017 (021 octal) – Bit 08 (10 octal).

Specification which applies to gates which belong to file **Float** :

Address	Description	T	FFF	EEE	SS	BB	Gate read	Gate write	Block read
T.FFF.EEE.BB	Float file	F	0...999	0...998 must be an even number		0...31	Yes	Yes	Yes

Because FLOAT gates are made of 4 bytes, the bit number is from 0 to.

Field **EEE** must be an even number

Example:

F.015.110.29 : Float - File 015 - Element 110 – Bit 29.

Specification which applies to gates which belong to file **Timer, Counter** and **Control** :

Address	Description	T	FF	EEE	SS	BB	Gate read	Gate write	Block read
T.FFF.EEE.00.BB	Timer file	T	0...999	0...999	00	0...15	Yes	Yes	Yes
T.FFF.EEE.00.BB	Counter file	C	0...999	0...999	00	0...15	Yes	Yes	Yes
T.FFF.EEE.00.BB	Control file	R	0...999	0...999	00	0...15	Yes	Yes	Yes

Example:

T.015.110.00.01 : Timer - File 015 - Element 110 – Sub-Element 00 – Bit 01.

C.007.120.00.15 : Counter - File 007 - Element 120 – Sub-Element 00 – Bit 15.

Specification which applies to gates which belong to file **PID** :

Address	Description	T	FF	EEE	SS	BB	Gate read	Gate write	Block read
T.FFF.EEE.00.BB	PID file	T	0...999	0...999	00..01	0...15	Yes	Yes	Yes

Example:

P.011.015.00.01 : PID - File 011 - Element 015 – Sub-Element 00 – Bit 01.

Blocks of digital gates

In case of gates which belong to file **Status, Bit, Integer, Output, Input, Float** the block must be made of gates which belong to the same type and have the same file number and the same or a sequential word number.

In case of gates which belong to file **Timer, Counter, Control, Pid** the block must be made of gates which belong to the same type and have the same file number and the same word number, and have the same or a sequential sub-element number.

The length of a block depends from the block composition. A block made of 16 gates belonging to the same word has a smaller dimension than a block made of 16 gates belonging to different words. The right dimensioning of a block is controlled by the driver itself during the first scanning; if the block length is too high, an error message will be displayed.

Example of valid block	Example of valid block	Example of NOT valid block	Example of NOT valid block
B.012.0003.01	T.001.006.00.00	B.012.003.01	T.001.006.00.00
B.012.0003.03	T.001.006.00.01	B.012.003.03	T.001.006.00.01
B.012.0004.14	T.001.006.00.02	B.012.004.14	T.001.006.00.02
B.012.0005.01	T.001.006.00.16	I.012.005.01	T.001.007.00.16
B.012.0006.01	T.001.006.00.05	I.012.006.01	T.001.007.01.05
B.012.0006.07	T.001.006.00.12	B.012.006.07	T.001.007.01.12
B.012.0006.08	T.001.006.00.06	B.012.006.10	T.001.008.02.06
B.012.0007.00	T.001.006.00.07	B.012.007.00	T.001.008.02.07

2.4 String gates

String gates are not allowed in this protocol.

2.5 Note

TIMER

Timer Sub Element	Description
00	Control bits
01	Preset
02	Acc

Timer Sub Element Detail	Bit number	Description
00	13	Done
00	14	Timing
00	15	Enable

COUNTER

Timer Sub Element	Description
00	Control bits
01	Preset
02	Accumulator

Timer Sub Element Detail	Bit Number	Description
00	11	Underflow
00	12	Overflow
00	13	Done
00	14	Count down
00	15	Count up

CONTROL

Timer Sub Element	Description
00	Control bits
01	Length
02	Position

Timer Sub Element Detail	Bit number	Description
00	08	Found
00	09	Inhibit
00	10	Unload
00	11	Error
00	12	Empty
00	13	Done
00	14	Enable unload
00	15	Enable

PID

PID Sub Element	Descrizione
00	Control bits 0
01	Control bits 1
02	Set point
04	Proportional gain / Controller gain
06	Integral gain / reset term
08	Derivative gain / Rate term
10	Feedforward or bias
12	Maximum scaling
14	Minimum scaling
16	Dead band
18	Set output
20	Maximum output limit
22	Minimum output limit
24	Loop update time
26	Scaled PV value
28	Scaled error value
30	Output
32	PV high alarm value
34	PV low alarm value
36	Error high alarm value
38	Error low alarm value
40	PV alarm dead band
42	Error alarm dead band
44	Maximum input value
46	Minimum input value
48	Tieback value for manual control

PID sub element	Numero Bit	Descrizione
00	00	Equation
00	01	Mode
00	02	Control
00	04	Set output
00	06	Derivative action
00	07	Process variable tracking
00	08	Cascade loop
00	09	Cascade selection
00	15	Enable
01	00	PV is alarm high
01	01	PV is alarm low
01	02	Error is alarmed high
01	03	Error is alarmed low
01	08	Set when error is DB
01	09	Output alarm, upper limit
01	10	Output alarm, lower limit
01	11	Set point out of range
01	12	PID initialized

2.6 Configuration

Allen Bradley PLC 5 DF1 Full Duplex Ver. 2.01

COM1	COM port
9600	Baud rate
None	Parity
1	Stop bits
8	Data bits
BCC	Checksum type
1000	Timeout [ms]
20	Query pause [ms]
0	PC node number

Buttons: OK, Cancel, Help

Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages.
- **Checksum type :** BCC or CRC16.
- **PC node number :** node number of the PC

3 ALLEN-BRADLEY DF1 FULL DUPLEX PROTOCOL FOR SLC500 / MicroLogix

3.1 Introduction

This protocol applies to series SLC 500 and MicroLogix PLC's.

The PC is seen as a node of the network DH 485 and can communicate with all SLC 500 and MicroLogix PLC of the network

Contact the provider of the PLC to select the proper interface board (ex. 1770-KF3 Interface Module)
The communication between the PC and the Interface Board is through the RS232 serial channel of the PC.

3.2 Numeric gates

General format:

T.FFF.EEE.SS or **T.FFF.EEE**

where

T : file identifier

FFF : file number

EEE : element number

SS : sub-element number

Specification which applies to gates: **Status, Bit, Integer, BCD, ASCII**:

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block
T.FFF.EEE	Status file	S	0...999	0...999		Yes	Yes	Yes
T.FFF.EEE	Bit file	B	0...999	0...999		Yes	Yes	Yes
T.FFF.EEE	Integer file	N	0...999	0...999		Yes	Yes	Yes
T.FFF.EEE	BCD file	D	0...999	0...999		Yes	Yes	Yes
T.FFF.EEE	ASCII file	A	0...999	0...999		Yes	Yes	Yes

The format of data is: WORD (2 Byte)

Example:

S.001.003 : Status - File 001 - Element 003.

B.121.303 : Bit - File 121 - Element 303.

N.100.120 : Integer - File 100 - Element 120.

D.050.020 : BCD - File 050 - Element 020.

A.007.023 : ASCII - File 007 - Element 023.

Specification which applies to gates: **Output and Input** :

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block
T.FFF.EEE	Output file	O	0...999	0...277 octal		Yes	Yes	Yes
T.FFF.EEE	Input file	I	0...999	0...277 octal		Yes	Yes	Yes

The format of data is: WORD (2 Byte)

Field **EEE** is an octal number

Example:

O.001.010 : Output - File 001 - Element 008 (010 octal).

I.001.021 : Input - File 001 - Element 017 (021 octal).

Specification which applies to gates which belong to file **Float** :

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block
T.FFF.EEE	Float file	F	0...999	0...998 must be an even number		Yes	Yes	Yes

The format of data is FLOAT (4 Byte).

The field **EEE** must be an even number

Example:

F.015.110 : Float - File 015 - Element 110.

Specification which applies to gates which belong to file **Float reverse mode** :

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block
---------	-------------	------	-----	-----	----	-----------	------------	------------

T.FFF.EEE	Float file	f	0...999	0...998 must be an even number		Yes	Yes	Yes
-----------	------------	---	---------	-----------------------------------	--	-----	-----	-----

The format of data is FLOAT (4 Byte).

The field **EEE** must be an even number

Example:

f.015.110 : Float - File 015 - Element 110.

Specification which applies to gates which belong to file **Timer, Counter** and **Control** :

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block
T.FFF.EEE.SS	Timer file	T	0...999	0...999	0...2	Yes	Yes	Yes
T.FFF.EEE.SS	Counter file	C	0...999	0...999	0...2	Yes	Yes	Yes
T.FFF.EEE.SS	Control file	R	0...999	0...999	0...2	Yes	Yes	Yes

The format of data is: WORD (2 Byte)

Example:

T.015.110.00 : Timer - File 015 - Element 110 – Sub-Element 00

C.007.120.01 : Counter - File 007 - Element 120 – Sub-Element 01

R.050.011.02 : Control - File 050 - Element 011 – Sub-Element 02

Blocks of numeric gates

In case of gates which belong to file **Status, Bit, Integer, BCD, ASCII, Output, Input** the block must be made of gates which belong to the same type and have the same file number and a sequential word number.

The maximum block length is 119 gates.

Numeric gates block	
T.FFF.EEE	
T.FFF.EEE+1	
T.FFF.EEE+2	
T.FFF.EEE+3	
T.FFF.EEE+4	

In case of gates which belong to file **Timer, Counter** and **Control** the block must be made of gates which belong to the same file type, file number and element and have a sequential sub-element number.

The maximum block length is 3 gates.

Numeric gates block	
T.FFF.EEE.00	
T.FFF.EEE.01	
T.FFF.EEE.02	

Numeric gates block	
T.FFF.EEE.SS+4	
T.FFF.EEE.SS+6	

3.3 Digital gates

The address of a digital gate is as follows:

T.FFF.EEE.SS.BB oppure **T.FFF.EEE.BB**

where

T : file identifier

FFF : file number

EEE : element number

SS : sub-element
BB : bit number

Specification which applies to gates which belong to file **Status, Bit, and Integer** :

Address	Description	T	FFF	EEEE	SS	BB	Gate read	Gate write	Block read
T.FFF.EEE.BB	Status file	S	0...999	0...999		00...15	Yes	Yes	Yes
T.FFF.EEE.BB	Bit file	B	0...999	0...999		00...15	Yes	Yes	Yes
T.FFF.EEE.BB	Integer file	N	0...999	0...999		00...15	Yes	Yes	Yes

Example:

S.001.003.00 : Status - File 001 - Element 003 – Bit 00.

B.121.303.15 : Bit - File 121 - Element 303 – Bit 15.

N.100.120.12: Integer - File 100 - Element 120 – Bit 12.

Specification which applies to gates which belong to file **Output e Input** :

Address	Description	T	FFF	EEE	SS	BB	Gate read	Gate write	Block read
T.FFF.EEE.BB	Output file	O	0...999	0...277 octal		00...17 octal	Yes	Yes	Yes
T.FFF.EEE.BB	Input file	I	0...999	0...277 octal		00...17 octal	Yes	Yes	Yes

Fields **EEE** e **BB** must be octal numbers

Example:

O.001.010.17 : Output - File 001 - Element 008 (010 octal) – Bit 15 (17 octal).

I.001.021.10 : Input - File 001 - Element 017 (021 octal) – Bit 08 (10 octal).

Specification which applies to gates which belong to file **Timer, Counter and Control** :

Address	Description	T	FF	EEE	SS	BB	Gate read	Gate write	Block read
T.FFF.EEE.00.BB	Timer file	T	0...999	0...999	00	0...15	Yes	Yes	Yes
T.FFF.EEE.00.BB	Counter file	C	0...999	0...999	00	0...15	Yes	Yes	Yes
T.FFF.EEE.00.BB	Control file	R	0...999	0...999	00	0...15	Yes	Yes	Yes

Example:

T.015.110.00.01 : Timer - File 015 - Element 110 – Sub-Element 00 – Bit 01.

C.007.120.00.15 : Counter - File 007 - Element 120 – Sub-Element 00 – Bit 15.

Blocks of digital gates

In case of gates which belong to file **Status, Bit, Integer, Output, Input, Float** the block must be made of gates which belong to the same type and have the same file number and the same or a sequential word number.

In case of gates which belong to file **Timer, Counter, Control** the block must be made of gates which belong to the same type and have the same file number and the same word number, and have the same or a sequential sub-element number.

The length of a block depends from the block composition. A block made of 16 gates belonging to the same word has a smaller dimension than a block made of 16 gates belonging to different words. The right dimensioning of a block is controlled by the driver itself during the first scanning; if the block length is too high, an error message will be displayed.

Example of valid block	Example of valid block	Example of NOT valid block	Example of NOT valid block
B.012.0003.01	T.001.006.00.00	B.012.003.01	T.001.006.00.00
B.012.0003.03	T.001.006.00.01	B.012.003.03	T.001.006.00.01
B.012.0004.14	T.001.006.00.02	B.012.004.14	T.001.006.00.02
B.012.0005.01	T.001.006.00.16	I.012.005.01	T.001.007.00.16
B.012.0006.01	T.001.006.00.05	I.012.006.01	T.001.007.01.05
B.012.0006.07	T.001.006.00.12	B.012.006.07	T.001.007.01.12
B.012.0006.08	T.001.006.00.06	B.012.006.10	T.001.008.02.06

B.012.0007.00	T.001.006.00.07	B.012.007.00	T.001.008.02.07
---------------	-----------------	--------------	-----------------

3.4 String gates

String gates are not allowed in this protocol.

3.5 Note

TIMER

Timer Sub Element	Description
00	Control bits
01	Preset
02	Acc

Timer Sub Element Detail	Bit number	Description
00	13	Done
00	14	Timing
00	15	Enable

COUNTER

Timer Sub Element	Description
00	Control bits
01	Preset
02	Accumulator

Timer Sub Element Detail	Bit Number	Description
00	11	Underflow
00	12	Overflow
00	13	Done
00	14	Count down
00	15	Count up

CONTROL

Timer Sub Element	Description
00	Control bits
01	Length
02	Position

Timer Sub Element Detail	Bit number	Description
00	08	Found
00	09	Inhibit
00	10	Unload
00	11	Error
00	12	Empty
00	13	Done
00	14	Enable unload

00	15	Enable
----	----	--------

3.6 Configuration

Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages.
- **Checksum type :** BCC or CRC16.
- **PC node number :** node number of the PC

4 ALLEN-BRADLEY Ethernet

4.1 Introduction

This protocol applies to series SLC500 and MicroLogix PLC's via Ethernet.

4.2 Numeric gates

General format:
T.FFF.EEE.SS or T.FFF.EEE

With:

T : file identifier
FFF : file number
EEE : element number
SS : sub-element number

Specification which applies to gates: **Status, Bit, Integer, BCD, ASCII:**

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read Block	PLC
T.FFF.EEE	Status file	S	0...999	0...999		Yes	Yes	Yes	SLC500 MicroLogix
T.FFF.EEE	Bit file	B	0...999	0...999		Yes	Yes	Yes	SLC500 MicroLogix
T.FFF.EEE	Integer file	N	0...999	0...999		Yes	Yes	Yes	SLC500 MicroLogix
T.FFF.EEE	BCD file	D	0...999	0...999		Yes	Yes	Yes	SLC500
T.FFF.EEE	ASCII file	A	0...999	0...999		Yes	Yes	Yes	SLC500

The format of data is: WORD (2 Byte)

Example:

S.002.003 : Status - File 002 - Element 003.
B.003.303 : Bit - File 003 - Element 303.
N.007.120 : Integer - File 007 - Element 120.
D.050.020 : BCD - File 050 - Element 020.
A.007.023 : ASCII - File 007 - Element 023.

Specification which applies to gates: **Output and Input :**

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block	PLC
T.FFF.EEE	Output file	O	0...999	0...277 octal		Yes	Yes	Yes	SLC500 MicroLogix
T.FFF.EEE	Input file	I	0...999	0...277 octal		Yes	Yes	Yes	SLC500 MicroLogix

The format of data is: WORD (2 Byte)

Field **EEE** is an octal number

Example:

O.000.010 : Output - File 000 - Element 008 (010 octal).
I.001.021 : Input - File 001 - Element 017 (021 octal).

Specification which applies to gates which belong to file **Float :**

Address	Description	Type	FFF	EEE	SS	Read gate	Write gate	Read block	PLC
T.FFF.EEE	Float file	F	0...999	0...999		Yes	Yes	Yes	SLC500 MicroLogix
T.FFF.EEE	Float file (reverse mode)	f	0...999	0...999		Yes	Yes	Yes	SLC500 MicroLogix

The format of data is FLOAT (4 Byte).

The field **EEE** must be an even number

Example:

F.008.110 : Float - File 008 - Element 110.
f.008.110 : Float reverse mode - File 008 - Element 110.

Specification which applies to gates which belong to file **Timer,Counter and Control :**

Address	Description	Typr	FFF	EEE	SS	Read gate	Write gate	Read block	PLC
T.FFF.EEE.SS	Status file	T	0...999	0...999	0..2	Yes	Yes	Yes	SLC500 MicroLogix
T.FFF.EEE.SS	Bit file	C	0...999	0...999	0..2	Yes	Yes	Yes	SLC500 MicroLogix
T.FFF.EEE.SS	Integer file	R	0...999	0...999	0..2	Yes	Yes	Yes	SLC500 MicroLogix

The format of data is: WORD (2 Byte)

Example:

T.004.110.00 : Timer - File 004 - Element 110 – Sub-Element 00

C.005.120.01 : Counter - File 005 - Element 120 – Sub-Element 01

R.006.011.02 : Control - File 006- Element 011 – Sub-Element 02

Blocks of numeric gates

In case of gates which belong to file **Status, Bit, Integer, BCD, ASCII, Output, Input** the block must be made of gates which belong to the same type and have the same file number and a sequential word number.

The maximum block length is 119 gates.

Block of numeric gates
T.FFF.EEE
T.FFF.EEE+1
T.FFF.EEE+2
T.FFF.EEE+3
T.FFF.EEE+4

In case of gates which belong to file **Timer, Counter** and **Control** the block must be made of gates which belong to the same file type, file number and element and have a sequential sub-element number.

The maximum block length is 3 gates.

Block of numeric gates
T.FFF.EEE.00
T.FFF.EEE.01
T.FFF.EEE.02

4.3 Digital gates

The address of a digital gate is as follows:

T.FFF.EEE.SS.BB oppure **T.FFF.EEE.BB**

With:

T : file identifier
FFF : file number
EEE : element number
SS : sub-element
BB : bit number

Specification which applies to gates which belong to file **Status, Bit, and Integer** :

Address	Description	Type	FFF	EEE	BB	Read gate	Write gate	Read block	PLC
T.FFF.EEE.BB	Status file	S	0...999	0...999	0..15	Yes	Yes	Yes	SLC500 MicroLogix
T.FFF.EEE.BB	Bit file	B	0...999	0...999	0..15	Yes	Yes	Yes	SLC500 MicroLogix

Address	Description	Type	FFF	EEE	BB	Read gate	Write gate	Read block	PLC
T.FFF.EEE.BB	Integer file	N	0...999	0...999	0..15	Yes	Yes	Yes	SLC500 MicroLogix

Example:

S.004.003.00 : Status - File 004 - Element 003 – Bit 00.

B.005.303.15 : Bit - File 005 - Element 303 – Bit 15.

N.006.120.12: Integer - File 006 - Element 120 – Bit 12.

Specification which applies to gates which belong to file **Output e Input** :

Address	Description	Type	FFF	EEE	BB	Read gate	Write gate	Read block	PLC
T.FFF.EEE.BB	Output file	O	0...999	0...277 octal	0..17 octal	Yes	Yes	Yes	SLC500 MicroLogix
T.FFF.EEE.BB	Input file	I	0...999	0...277 octal	0..17 octal	Yes	Yes	Yes	SLC500 MicroLogix

Fields **EEE** e **BB** must be octal numbers

Example:

O.000.010.17 : Output - File 000 - Element 008 (010 octal) – Bit 15 (17 octal).

I.001.021.10 : Input - File 001 - Element 017 (021 octal) – Bit 08 (10 octal).

Specification which applies to gates which belong to file **Timer, Counter and Control** :

Address	Description	Type	FFF	EEE	SS	BB	Read gate	Write gate	Read block	PLC
T.FFF.EEE.SS.BB	Timer file	T	0...999	0...999	0..2	0..15	Yes	Yes	Yes	SLC500 MicroLogix
T.FFF.EEE.SS.BB	Counter file	C	0...999	0...999	0..2	0..15	Yes	Yes	Yes	SLC500 MicroLogix
T.FFF.EEE.SS.BB	Control file	R	0...999	0...999	0..2	0..15	Yes	Yes	Yes	SLC500 MicroLogix

Esempio:

T.004.110.00.07 : Timer - File 004 - Element 110 – Sub-Element 00 - Bit 07

C.005.120.01.12 : Counter - File 005 - Element 120 – Sub-Element 01 - Bit 12

R.006.011.02 .03: Control - File 006 - Element 011 – Sub-Element 02 - Bit 03

Blocks of digital gates

In case of gates which belong to file **Status, Bit, Integer, Output, Input, Float** the block must be made of gates which belong to the same type and have the same file number and the same or a sequential word number.

In case of gates which belong to file **Timer, Counter, Control** the block must be made of gates which belong to the same type and have the same file number and the same word number, and have the same or a sequential sub-element number.

The length of a block depends from the block composition. A block made of 16 gates belonging to the same word has a smaller dimension than a block made of 16 gates belonging to different words. The right dimensioning of a block is controlled by the driver itself during the first scanning; if the block length is too high, an error message will be displayed.

Example of valid block	Example of valid block	Example of NOT valid block	Example of NOT valid block
B.012.003.01	T.001.006.00.00	B.012.003.01	T.001.006.00.00
B.012.003.03	T.001.006.00.01	B.012.003.03	T.001.006.00.01
B.012.004.14	T.001.006.00.02	B.012.004.14	T.001.006.00.02
B.012.005.01	T.001.006.00.16	I.012.005.01	T.001.007.00.16
B.012.006.01	T.001.006.00.05	I.012.006.01	T.001.007.01.05

B.012.006.07	T.001.006.00.12	B.012.006.07	T.001.007.01.12
B.012.006.08	T.001.006.00.06	B.012.006.08	T.001.008.02.06
B.012.007.00	T.001.006.00.07	B.012.007.00	T.001.008.02.07

4.4 String gates

String gates are not allowed in this protocol.

4.5 Note

TIMER

Timer Sub Element	Description
00	Control bits
01	Preset
02	Acc

Timer Sub Element Detail	Bit number	Description
00	13	Done
00	14	Timing
00	15	Enable

COUNTER

Timer Sub Element	Description
00	Control bits
01	Preset
02	Accumulator

Timer Sub Element Detail	Bit Number	Description
00	11	Underflow
00	12	Overflow
00	13	Done
00	14	Count down
00	15	Count up

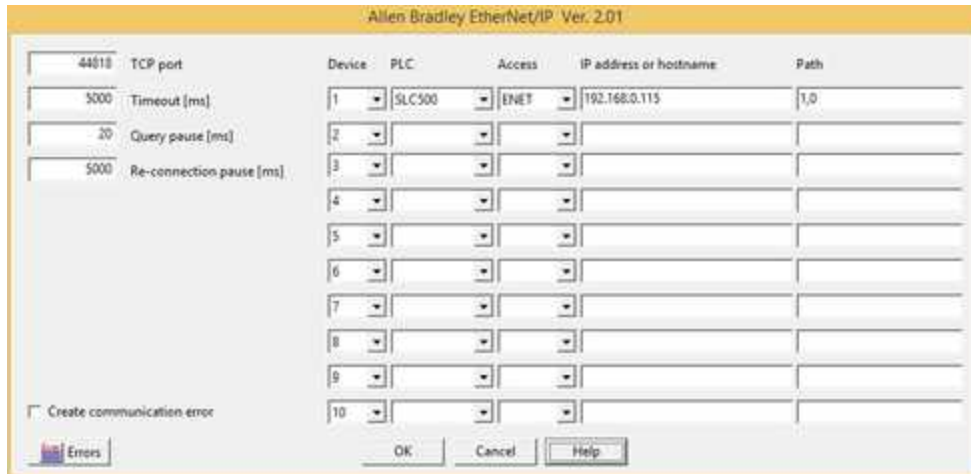
CONTROL

Timer Sub Element	Description
00	Control bits
01	Length
02	Position

Timer Sub Element Detail	Bit number	Description
00	08	Found
00	09	Inhibit
00	10	Unload
00	11	Error
00	12	Empty

00	13	Done
00	14	Enable unload
00	15	Enable

4.6 Configuration



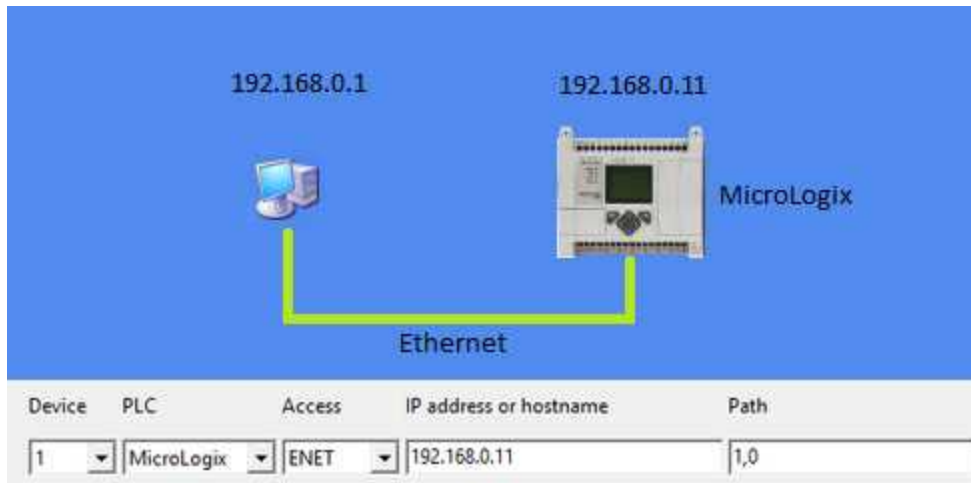
Protocol configuration window.

- **Port number:** ethernet communication port.
- **Timeout [ms]:** maximum wait time for an answer message
- **Query pause [ms]:** tempo di attesa fra due richieste.
- **Re-connection pause [ms]:** pause between an answer and the next question.
- **Save communication error file :** if it is checked, a communication error message will be saved in a file on the disk every time that a communication error happens. A list of the last 100 communication error messages can be viewed (also in Runtime mode), by clicking on "Errors" button.

It is possible to have till to 10 PLC connection on a unique channel

- **Device:** it is the logical address associated to the PLC and must agree with the "Device" field in the Gate Builder for all the gates referred to the PLC selected.
- **PLC:** PLC model
- **Access:** PLC access mode (via Ethernet or ControlNet)
- **IP address:** PLC IP address
- **Path:** it is expressed as a sequence of "*port / link address*": identical to the syntax of RSLogix 5000 - Message configuration - Communication path. *Port* is a way out of a device : it can be a backplane or a network. *Link address* is a destination node: if the corresponding port is a backplane then the link address represents the slot number, else it represents a network address.

Example:



Backplane=1
 Slot = 0 (logic controller)
 Path =[Backplane],[Slot]=1,0

5 AVEBus

5.1 Introduction

Communication protocol for AVE devices on Domotic Bus.

Hardware interface PC-AveBus **BSA-RS232** o **BSA-USB** is needed.

Supported devices (family grouped) :

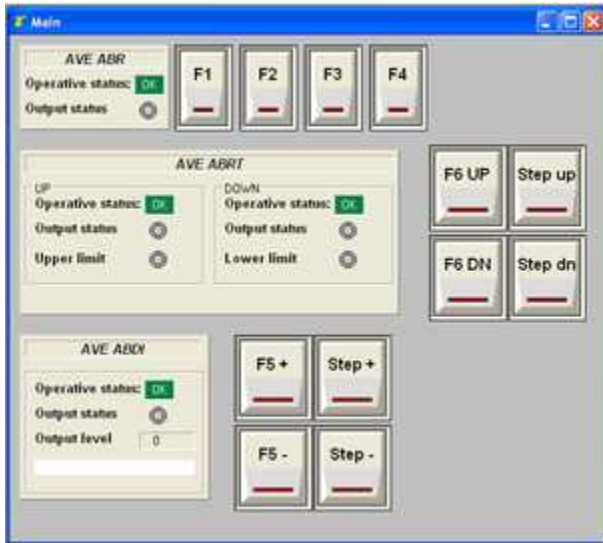
AVE_ABR : receiver for ON/OFF output.

AVE_ABRT: receiver for blinds output.

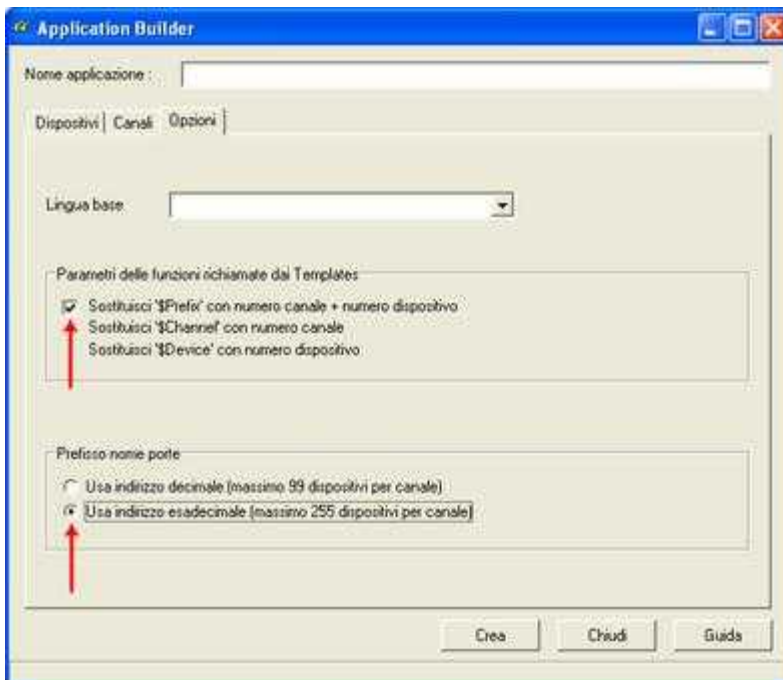
AVE_ABDI: receiver for analog output.

AVE_ABT: transmitter device.

They are also already available in the Library, in order to allow to build a working application (like in the figure below) in few time.

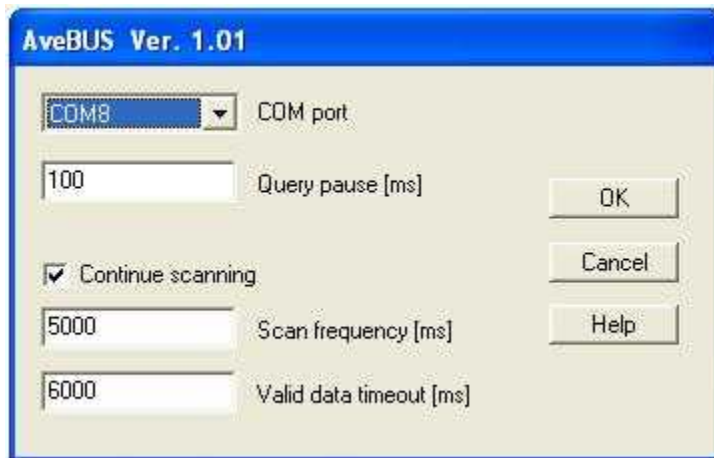


To build applications with AVE devices, use **Application Builder** (accessible from **Project Manager**).



Set the "Parameters of functions called from Templates" and the "Gate name prefix" as reported in the picture above.

5.2 Configuration



Protocol configuration window.

- **COM port:** COM port number associated to the BSA-RS232 or BSA-USB adapter.
- **Query pause [ms]:** wait time between two devices sampling. Expressed in milliseconds, must be ≥ 100 ms.
- **Continue scanning:** if this checkbox is enabled then all the AVE devices present in the application will be scanned every "Scan frequency" time. With this flag enabled, it is possible to monitor continuously the bus status and trap device or bus failure.
- **Scan frequency [ms]:** in case of "Continue scanning" enabled, this parameter specify the sample frequency of each device.
- **Valid data timeout [ms]:** in case of "Continue scanning" enabled, this parameter tell to the software to consider the data invalid if it is not refreshed between "Valid data timeout" milliseconds. This parameter must be greater than "Scan frequency" parameter otherwise you can receive a wrong communication error status from the driver. Note that if you have a lot of device to scan, you may have to increase "Valid data timeout" because the complete scanning time may be greater than "Scan frequency" parameter.

6 BACnet

6.1 Introduction

BACnet, defined by ASHRAE (American Society of Heating and Air-conditioning Engineers) is a protocol designed specifically for building automation and control systems such as heating, ventilation, air conditioning, lighting, access control and fire detection .

The version implemented in this driver is BACnet over IP.

Supported properties:

Let's consider "*Simple properties*" of an object, those properties whose result can be associated directly to a single numeric, digital or string gate.

Let's consider "*Structured Properties*" of an object, those properties whose result can be associated with a group of numeric, digital or string gates, suitably arranged in sequence among themselves.

Simple properties:

a property of this type can be defined directly in the GateBuilder by specifying the appropriate address in the gate.

Are supported all properties that the result of which is one of the following Datatypes:

- **Boolean**
- **Unsigned Integer**
- **Signed Integer** (2's complement notation)
- **Real** (ANSI/IEEE-754 floating point)
- **Double** (ANSI/IEEE-754 double precision floating point)
- **Octet String**
- **Character String** (ANSI_X3_4 and ISO_8859_1 formats are supported)
- **Bit String**
- **Enumerated**
- **BACnetObjectIdentifier**

Structured properties:

as the use of this type of property requires the definition of a group of gates appropriately arranged in sequence with each other, it is necessary to resort to the use of ApplicationBuilder (*) to introduce them in the application: in this way it will be automatically created the list of gates with their templates ready to be used.

The following properties are supported:

- **DateList** (library object BACnet_Property_DateList)
- **EffectivePeriod** (library object BACnet_Property_EffectivePeriod)
- **ExceptionSchedule** (library object BACnet_Property_ExceptionSchedule)
- **WeeklySchedule** (library object BACnet_Property_WeeklySchedule)

Notes:

* ApplicationBuilder can be called from the menu item "*Project-> New-> Project using ApplicationBuilder ...*" of the ProjectManager.

6.2 Numeric gates

The gate address is specified in the following order:

Host, DeviceInstance, ObjectType, ObjectInstance, Property, [COV], [COV increment], [Priority]

Where:

Host: is the identifying name of the IP address specified in the protocol configuration, through which to reach the device.

DeviceInstance: instance number of the device

ObjectType: type of the object inside the device.

ObjectInstance: Instance number of that object type.

Property: property number of the object (*Note 2*).

COV: optional parameter that specifies whether the COV (Change Of Value) function must be enabled for this property.

COV increment: optional parameter that specifies the minimum variation of Present_Value necessary to cause a COVNotification by the device.

Priority: optional parameter that specifies the write priority level (1...16) (*Note 3*).

Example 1:

Host1, 1, 2, 0, 85

Where :

Host = **Host1**

DeviceInstance = **1**

ObjectType = **2** (Analog value)

ObjectInstance = **0**

Property = **85**

COV = **not specified**

COV increment = **not specified**

Priorità di scrittura = **not specified**

Notes:

As the COV is not specified, the property will be cyclically sampled according to the sampling frequency specified in the gate.

Since no writing priority is specified, the writes in the property will be made without the priority parameter.

Example 2:

Host1, 1, 2, 0, 85, COV

Where :

Host = **Host1**

DeviceInstance = **1**

ObjectType = **2** (Analog value)

ObjectInstance = **0**

Property = **85**

COV = **COV** (enabled)

COV increment = **not specified**

Priority write = **not specified**

Notes:

As the COV parameter is specified, the supervisor will send a SubscribeCOV request to the device specifying a subscription time equal to that set in the protocol configuration window (*Lifetime* parameter): if the device accepts the subscription, the supervisor will no longer interrogate it, but will wait for the latter to send him notifications of variation in the value of the specified property. The COV increment used is the one specified in the device.

Shortly before the subscription time expires, the supervisor will automatically make a new subscription. If, however, the device does not support the COV function, the software will cyclically interrogate the device according to the sampling frequency specified in the gate.

Since no writing priority is specified, the writes in the property will be made without the priority parameter.

Example 3:

Host1, 1, 2, 0, 85, COV, .6

Where :

Host = **Host1**

DeviceInstance = **1**

ObjectType = **2** (Analog value)

ObjectInstance = **0**

Property = **85**

COV = **COV** (enabled)

COV increment = **.6**

Priority write = **non specificato**

Notes:

As far as the COV parameter is concerned, the considerations given in Example 2 apply, however the COVIncrement used is the one specified in the example ie .6.

Since no writing priority is specified, the writes in the property will be made without the priority parameter.

Example 4:

Host1, 1, 2, 0, 85, COV, .6, 15

Where :

Host = **Host1**

DeviceInstance = **1**

ObjectType = **2** (Analog value)

ObjectInstance = **0**

Property = **85**

COV = **COV** (enabled)

COV increment = **.6**

Priority write = **15**

Notes:

During the writing of the value, the specified priority will also be sent, ie 15.

Example 5:

Host1, 1, 2, 0, 85,,, 15

Where :

Host = **Host1**

DeviceInstance = **1**

ObjectType = **2** (Analog value)

ObjectInstance = **0**

Property = **85**

COV = **not enabled**

COV increment = **not enabled**

Priority write = **15**

Notes:

The example shows how to specify the address if there is a parameter that follows unspecified parameters ie ,,15.

Note 1:

A block of gates can only consist of gates with the same *Host*, *DeviceInstance*, *ObjectType* and *ObjectInstance*.

For the grouped gates, it is not possible to specify the COV function.

Note 2:

All properties of the following type can be associated with a numeric gate:

Boolean, UnsignedInteger, SignedInteger, Float, Double, BitString, Enumerated.

All properties of the following type can be associated with a digital gate:

Boolean, Enumerated.

All properties of the following type can be associated with a string gate:

Boolean, UnsignedInteger, SignedInteger, Float, Double, OctetString, CharacterString, BitString, Enumerated, ObjectIdentifier.

Note 3:

It is possible to relinquish writing a value in the specified priority level only through the use of a string

gate having the same address as the numerical gate and specifying an empty string as value.

6.3 Digital gates

Refer to Numeric gates.

6.4 String gates

Refer to Numeric gates.

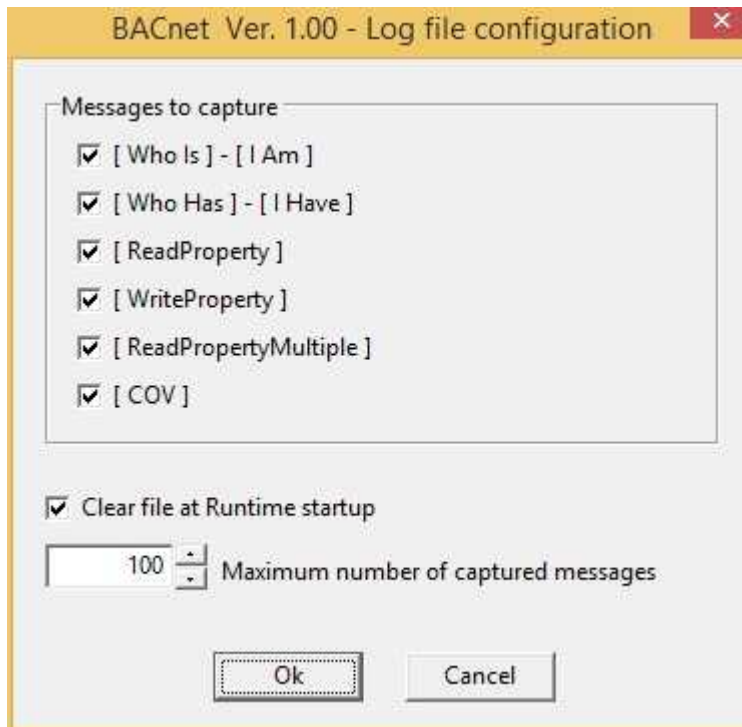
6.5 Configuration

ID	IP address
Host1	192.168.10.10
Host2	192.168.10.11
Host3	192.168.10.12

Protocol configuration window.

- **Port:** communication port.
- **Wait answer timeout [ms]:** maximum time (expressed in milliseconds) within which a response is expected from the device (for messages that require an answer).
- **Query pause [ms]:** time (expressed in milliseconds) of waiting between two requests.
- **COV - Subscriber Process Identifier :** process identifier assigned to the supervisor during use of the COV function.
- **COV - Lifetime (hh:mm:ss) :** COV function subscription time.
- **IP address :** device IP address.

- **Save Log file:** enabling the saving of the Log communication file (for diagnostic purposes).
- **Config:** invoke the configuration dialog of the Log file.
- **Log:** show the Log file.



Log file configuration window.

- **[Who Is] - [I Am]** : if activated, all messages "Who Is" and "I Am" will be recorded in the Log file.
- **[Who Has] - [I Have]** : if activated, all messages "Who Has" and "I Have" will be recorded in the Log file.
- **[ReadProperty]** : if activated, all "ReadProperty" messages will be recorded in the Log file.
- **[WriteProperty]** : if activated, all "WriteProperty" messages will be recorded in the Log file.
- **[ReadPropertyMultiple]** : if activated, all "ReadPropertyMultiple" messages will be recorded in the Log file.
- **[COV]** : if activated, all messages related to the VOC functions will be recorded in the Log file.
- **Clear file at Runtime startup** : if activated, the Log file will be deleted each time the Runtime is restarted.
- **Maximum number of captured messages:** maximum number of messages saved in the Log file, beyond which the saving will be deactivated until the next restart of the Runtime.

7 DATA STREAM (CR Magnetics)

7.1 Introduction

Communication protocol for **Data Stream (CR Magnetics)** devices.

Supported devices:

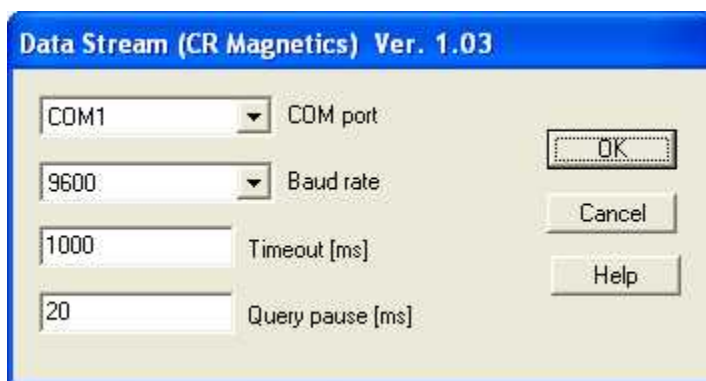
Multifunction Digital Transducer: CRD5110,CRD5150,CRD5170

Digital Current Transducer: CRD4110,CRD4150,CRD4170

Digital Voltage Transducer: CRD4510,CRD4550,CRD4570

Use **Application Builder** tool to build an application with the devices above.

7.2 Configuration



Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Timeout [ms]:** timeout for answer message.
- **Query pause [ms]:** timeout between two request messages

8 EUROTHERM BISYNCH ASCII

8.1 Introduction

Communication protocol for **Eurotherm** devices.

8.2 Numeric gates

The address of the gate is indicated by a mnemonic code; for a directory of all mnemonic codes refer to the manual of the Eurotherm device.

Mnemonic code	Read gate	Write gate	Read block
XX	Yes	Yes	No

Example: the following are examples of gate address:

PV : Process Value

OP : Output Power.

VP: Output Position.

Note:

It is not possible to have blocks of numeric gates.

8.3 Digital gates

The address of the gate is indicated by a mnemonic code; for a directory of all mnemonic codes refer to the manual of the Eurotherm device.

Mnemonic code	Read gate	Write gate	Read block
XX	Yes	Yes	No

Example: the following are examples of gate address:

FR : Fast Run

Z1 : Logic 1 output

ut : Segment synchronisation

Note:

It is not possible to have blocks of digital gates.

8.4 String gates

The address of the gate is indicated by a mnemonic code; for a directory of all mnemonic codes refer to the manual of the Eurotherm device.

Mnemonic code	Read gate	Write gate	Read block
XX	Yes	Yes	No

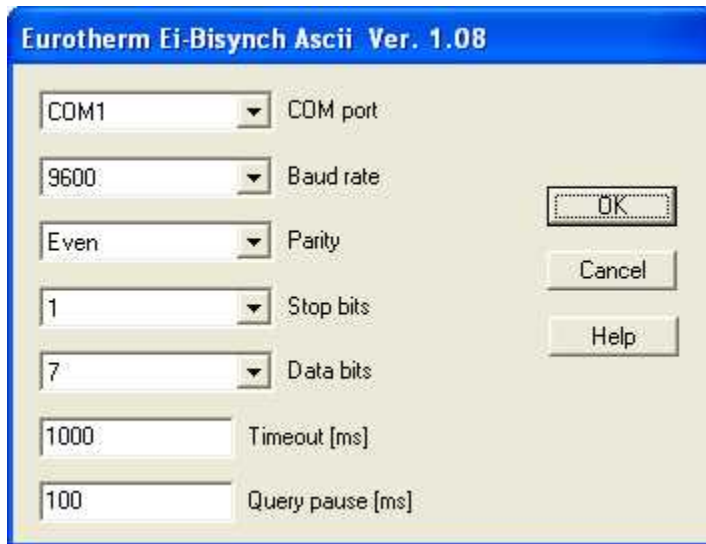
Esempio: the following are examples of gate address:

II : Instrument identity

Note:

It is not possible to have blocks of string gates.

8.5 Configuration



Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages.

9 EV2001 (Bilanciali)

9.1 Introduction

Communication protocols for **EV2001 (Bilanciali)** scales.

9.2 Numeric gates

A **block** of 2 numeric gates must be defined using "Gate Builder" tool.

The first numeric gate must have the following address: **EV2001_WEIGHT_TYPE**
It contains the stability value:

- 0: *Weight stabilized*
- 1: *Weight not stabilized*
- 2: *Invalid weight (under 0 or overflow)*

The second numeric gate must have the following address : **EV2001_WEIGHT**
It contains the net weight .

9.3 Digital gates

Digital gates are not allowed in this protocol.

9.4 String gates

String gates are not allowed in this protocol.

9.5 Configuration



Protocol configuration window.

- **COM port:** serial port name.

10 GEFRAN - CENCAL

10.1 Introduction

This communication protocol is used by Gefran devices.

10.2 Numeric gates

The gate address is specified adding the fields Function and Number of bytes of the following table.

Function code	Number of bytes	Read gate	Write gate	Read block
CC (Hex value)	N (decimal value)	Yes	Yes	Yes
CCCC (Hex value)	N (Decimal value)	Yes	Yes	Yes

Example: here are some examples of numeric gates:

6F-2 : Input value (2 bytes) of the Gefran 3300 device.

A6-1 : Control type (1 byte) of the Gefran 3300 device.

8002-1 : Number of cycle to execute of the Gefran 3500 device.

Code function must be compound only by 2 or 4 characters.

Note:

A block of numeric gates must be made only by gates with consecutive Function code.

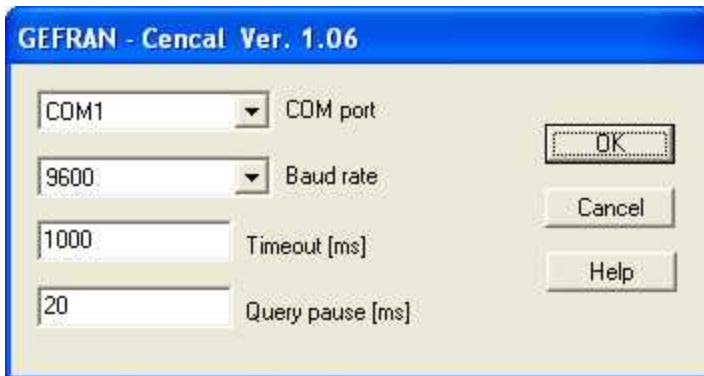
10.3 Digital gates

Digital gates are not allowed in this protocol.

10.4 String gates

String gates are not allowed in this protocol.

10.5 Configuration



Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages

11 IDEC IZUMI FA

11.1 Introduction

This protocol is used for communication with IZUMI PLCs type FA

The communication between the PC and the PLCs is through the PC serial interface and requires a RS232/RS422 converter or a RS232/Optical Fiber converter; up to 255 devices can be connected on the same serial link

11.2 Numeric gates

The gate address is specified adding the fields Function and Address of the following table.

Function	Description	Address	Gate read	Gate write	Block read
T	TIMER (only 14 less significant bit)	XX 00...79	Yes	No	No
C	COUNTER (only 14 less significant bit)	XX 00...47	Yes	No	No
CT	COMPLETE TIMER (16 bit)	XX 00...79	Yes	No	No

CC	COMPLETE COUNTER (16 bit)	XX 00...47	Yes	No	No
PT	PRESET TIMER	XX 00...79	Yes	Yes	No
PC	PRESET COUNTER	XX 00...47	Yes	Yes	No
10T	10 ms TIMER COUNTER	XXXX 1100...1179	Yes	No	No
DR	DATA REGISTER	XXX 800...899	Yes	Yes	No
EDR	EXTENDED DATA REGISTER	XXXX 1500...1799	Yes	Yes	No

Example: the following are some examples of numeric gates:

T1 : Timer 01.

PC47 : Preset counter 47.

10T1100 : 10 msec timer counter 1100.

EDR1500: Extended data register 1500.

Note:

This protocol does not allow reading or writing of blocks of numeric gates.

11.3 Digital gates

The gate address is specified adding the fields Function and Word Address and Bit Address of the following table.

Function	Description	Word address	Bit address	Gate read	Gate write	Block read
I	INPUT	XX 00...15	X 0...7	Yes	Yes	No
O	OUTPUT	XX 20...35	X 0...7	Yes	Yes	No
IR	INTERNAL RELAY	XX 40...71	X 0...7	Yes	Yes	No
EI	EXPANS. INPUT	XXX 200...215	X 0...7	Yes	Yes	No
EO	EXPANS. OUTPUT	XXX 220...235	X 0...7	Yes	Yes	No
EIR	EXPANS. INTERNAL RELAY	XXX 240...271	X 0...7	Yes	Yes	No
SFR	SHIFT REGISTER	XXX 000...127		Yes	Yes	No

Example: the following are some examples of digital gates:

I157 : Input word 15 – bit 7.

IR506 : Internal relay word 50 – bit 6.

EIR2503 : Expansion internal relay word 250 – bit 3.

SFR101 : Shift register bit 101.

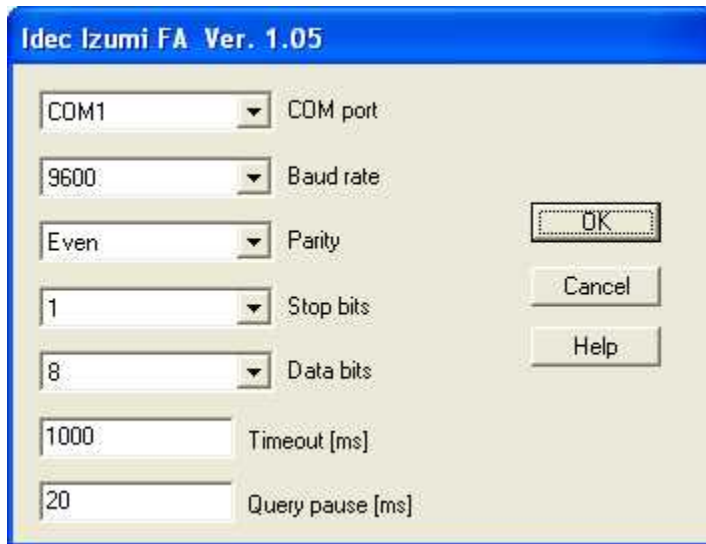
Note:

This protocol does not allow reading or writing of blocks of digital gates.

11.4 String gates

String gates are not allowed in this protocol.

11.5 Configuration



Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages.

12 KLOCKNER MOELLER SUCOM - A

12.1 Introduction

This protocol is used for communication with Klockner Moeller PLCs : **PS32, PS306, PS316-CPU-223, PS416-CPU-223, PS416-CPU-400.**

Communication between PC and PLC is performed via the standard PC serial link but requires a RS232/RS485 converter (Converter Klockner Moeller “UM 1.2” is recommended); only one device can be connected on the serial link

12.2 Numeric gates

The gate address is specified adding the fields Function and Address of the following table.

Numeric gates for PS32, PS306, PS316-CPU-223, PS416-CPU-223

Function	Description	Address	Gate read	Gate write	Block read
IW	WORD DIGITAL INPUTS (16 bit)	XX	Yes	No	Yes
QW	WORD DIGITAL	XX	Yes	Yes	Yes

	OUTPUTS (16 bit)				
MW	WORD MERKER (16 bit)	XXXX	Yes	Yes	Yes
IB	BYTE DIGITAL INPUTS (8 bit)	XX+ ".0" 0...7	Yes	No	Yes
IB	BYTE DIGITAL INPUTS (8 bit)	XX+ ".8" 8...15	Yes	No	Yes
QB	BYTE DIGITAL OUTPUTS (8 bit)	XX+ ".0" 0...7	Yes	Yes	Yes
QB	BYTE DIGITAL OUTPUTS (8 bit)	XX+ ".8" 8...15	Yes	Yes	Yes
MB	BYTE MERKER (8 bit)	XXXX+ ".0" 0...7	Yes	Yes	Yes
MB	BYTE MERKER (8 bit)	XXXX+ ".8" 8...15	Yes	Yes	Yes

Example: here are some examples of numeric gates address:

IW01 :Input word 01.

IB04.0 : Input byte 04 (bit 0..7)

IB05.8 : Input byte 05 (bit 8..15)

MB0123: Merker 0123.

MB0234.8: Merker 0234 (bit 8..15)

Numeric gates for PS416-CPU-400

Function	Description	Address	Gate read	Gate write	Block read
MB	BYTE MERKER (8 bit)	XXXX	Yes	Yes	Yes

Example: here are some examples of numeric gates address:

MB0001 :Merker byte 0001.

MB0301 :Merker byte 0301.

Note:

A block can be made of a maximum of 32 WORD type gates or 64 BYTE type gates

A block of numeric gates must have only gates with the same function and sequential address.

Example of valid numeric gates block	Example of NOT valid numeric gates block
IW01	IW01
IW02	IW06
IW03	QW07
IW04	QW09
IW05	IW10

In case of **PS32**, **PS306**, **PS316-CPU-223**, **PS416-CPU-223** a block of numeric gates with BYTE format must be made of gates with the same Function and sequential Address with alternate "0" and "8".

Example of valid numeric gates block	Example of valid numeric gates block	Example of NOT valid numeric gates block	Example of NOT valid numeric gates block
IB00.0	QI00.8	IB00.0	QI00.8
IB00.8	QI00.0	IB01.0	IB00.0
IB01.0	QI01.8	IB01.8	IB01.0
IB01.8	QI01.0	IB03.0	IB01.8
IB02.0	QI02.8	IB04.0	IB02.0
IB02.8	QI02.0	IB04.8	IB03.0

12.3 Digital gates

The gate address is specified adding the fields Function, Word address and Bit address of the following table.

Digital gates for PS32, PS306, PS316–CPU-223, PS416-CPU-223

Function	Description	Word address	Bit address	Gate read	Gate write	Block read
I	INPUT	XX	XX	Yes	Yes	Yes
Q	OUTPUT	XX	XX	Yes	Yes	Yes
M	MERKER	XX	XX	Yes	Yes	Yes

Example: here are some examples of digital gates address:

I00.01 : digital input word 00 bit 1.

Q03.15 : Digital output word 03 bit 15.

M1234.09 : Merker 1234 bit 9.

Digital gates for PS416 - CPU- 400

Function	Description	Byte address	Bit address	Gate read	Gate write	Block read
M	MERKER	XXXX	X	Yes	Yes	Yes

Example: here are some examples of digital gates address:

M0001.1 : Merker byte 0001 bit 1.

M0034.7 :Merker byte 0034 bit 7.

Note:

A block can be made by a maximum of 64 bytes, but this doesn't mean that the maximum number of digital gates is 64. Let us group in a block the following digital gates: **I00.00**, **I00.01**, **I00.03**, **I00.04**; these 4 gates take the space of only one byte, so other 63 bytes are available

A block is defined as a number of sequential bytes in the PLC: so digital gates which are grouped in a block must refer to the same byte or to the sequential.

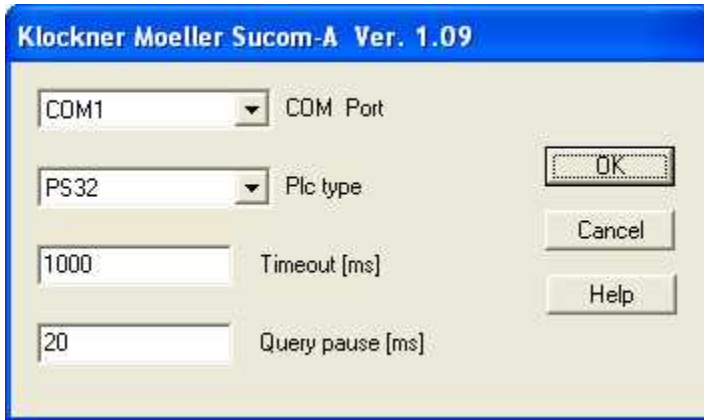
All digital gates of a block must have the same function.

Example of valid block	Example of NOT valid block
I00.00	I00.00
I00.01	I01.00
I00.02	I02.15
I00.09	I03.12
I00.15	I05.14
I01.15	I06.03
I01.14	I06.04
I02.00	I06.05
I02.09	I06.06
I03.07	I06.07
I03.12	I06.09

12.4 String gates

String gates are not allowed in this protocol.

12.5 Configuration



Protocol configuration window.

- **COM port:** serial port name.
- **Plc type:** Plc type: *PS32, PS306, PS316-CPU-223, PS416-CPU-223, PS416-CPU-400.*
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages.

13 KLOCKNER MOELLER SUCOM - A per PS4

13.1 Introduction

This protocol is used for communication with Klockner Moeller PLCs **PS4-141-MM1, PS4-151-MM1, PS4-201-MM1, PS4-341-MM1.**

Communication between PC and PLC is performed via the standard PC RS232 serial link; in case of cable length greater than 10 meters a RS232/RS422 converter is required; only one device can be connected on the serial link

Data are dynamically allocated inside the PLC according to the number of merker byte declared during PLC programming. The first thing done by the driver is to build a map of data addresses; if data in the PLC are reallocated, it is necessary to restart the software to allow the driver to update the address map.

13.2 Numeric gates

The gate address is specified adding the fields Function and Address of the following table.

Function	Description	Address	Gate read	Gate write	Block read
MW	WORD MERKER (16 bit)	XXXXX (only even numbers)	Yes	Yes	Yes
MB	BYTE MERKER (8 bit)	XXXXX	Yes	Yes	Yes

Example: here are some examples of digital gates address:

MW01230: Merker word 01230.

MB02341 : Merker byte 02341

Note:

A block can be made of a maximum of 32 WORD type gates or 64 BYTE type gates

A block of numeric gates must have only gates with the same function and sequential address.

Example of valid block	Example of NOT valid block
MW00000	MW00000
MW00002	MW00003
MW00004	MW00006
MW00006	MW00007
MW00008	MW00010

13.3 Digital gates

The gate address is specified adding the fields Function, Word address and Bit address of the following.

Function	Description	Byte address	Bit address	Gate read	Gate write	Block read
M	MERKER	XXXXX	X 0...7	Yes	Yes	Yes

Example: here are some examples of digital gates address:

M01234.7 : Merker 01234 bit 7.

M11220.0 : Merker 11220 bit 0.

Note:

A block can be made by a maximum of 64 bytes, but this doesn't mean that the maximum number of digital gates is 64. Let us group in a block the following digital gates: **M00000.0**, **M00000.1**, **M00000.3**, **M00000.4**; these 4 gates take the space of only one byte, so other 63 bytes are available

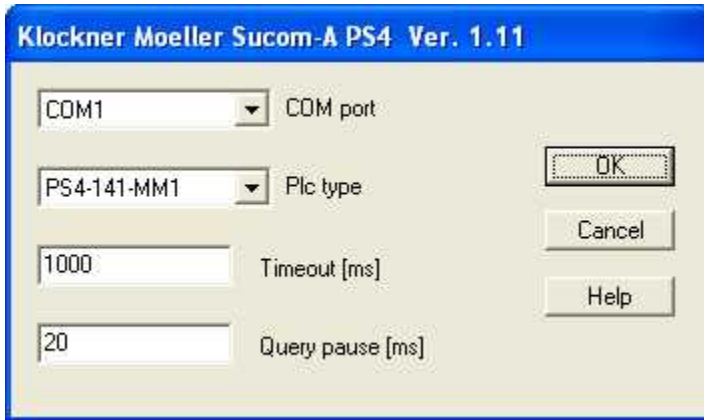
A block is defined as a number of sequential bytes in the PLC: so digital gates which are grouped in a block must refer to the same byte or to the sequential one.

Example of valid block	Example of NOT valid block
M00000.0	M00000.0
M00000.1	M00000.1
M00000.2	M00001.0
M00000.4	M00002.0
M00000.5	M00003.1
M00001.5	M00005.4
M00001.6	M00006.3
M00002.0	M00006.4
M00002.1	M00006.5
M00003.3	M00006.7
M00004.4	M00009.0

13.4 String gates

String gates are not allowed in this protocol.

13.5 Configuration



Protocol configuration window.

- **COM port:** serial port name.
- **Plc type:** Plc type: *PS4-141-MM1*, *PS4-151-MM1*, *PS4-201-MM1*, *PS4-341-MM1*.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages.

14 KNX (Falcon Library)

14.1 Introduction

Protocol designed to allow communication with building automation devices connected through **KNX** bus.

KNX bus interfacing requires a serial or USB BCU (Bus Coupling Unit) or a Ethernet port to connect PC to a KNXnet/IP router. Supervisor interfacing is the same as ETS.

KNX protocol requires *Falcon Library*, the official way to interface a Windows PC to KNX bus. To run this protocol, you need to acquire and install Falcon Library.

Falcon Library (*runtime* version) can be downloaded from KNX official site (<http://www.knx.org/knx-tools/falcon/downloads/>). Run setup and follow on-screen instruction to install Falcon Library. Some Falcon Library HW/SW requirements (<http://www.knx.org/knx-tools/falcon/requirements/>) can prevent use of this communication protocol on some machines. For example Falcon Library version 2.0 can not be run on Microsoft Windows 2000.

From version 2.0, Falcon Library is released free of charge. If you need to use a prior version, you have to follow the procedure described here to unlock Falcon Library using acquired License Key.

KNX devices supervision is based on *group addresses*; group addresses must be defined and downloaded on the system using ETS. So you can read and write group addresses linking it to gates. Furthermore you can set listening group addresses: protocol monitors the bus, whenever some one read, write or transmit one of these group addresses it captures the value and set the linked gate value.

Please note that you can only read, write and listen only devices datapoints linked in some of system group addresses. In addition manufacturers can prevent reading or writing some datapoints even if they are linked to group addresses. For more details, refer to KNX specifications and devices reference manuals.

KNX protocol is multi-master; any device must not abuse the bus usage. So avoid frequent reading of several gates, instead design your system in the manner that you can read once (gate read only at startup) and then stay listening.

14.2 Numeric gates

You can link a gate to:

- one read group address (*RGA*)
- one write group address (*WGA*)
- one or more listening group addresses (*LGA1*, *LGA2*, ...)

RGA is the group address read synchronously from the bus (PC asks the bus); value answered by the bus becomes the gate value. *RGA* must be a readable group address; a gate may not have a read group address.

WGA is the group address written synchronously on the bus (PC sets the value on the bus). *WGA* must be a writeable group address; a gate may not have a write group address.

LGA1, *LGA2*, ... are the listening group addresses. When one of these group address passes through the bus, value is captured and becomes the gate value. This is due to the multi-master architecture of KNX protocol: each device can read, write or transmit group addresses (e.g. a light switch can write the value of a group address, a room sensor can transmit on the bus the changed value of temperature). A gate may have no listening group address and can be linked to a maximum of eight group addresses. Different gates can listen to same group addresses.

Numeric gates address

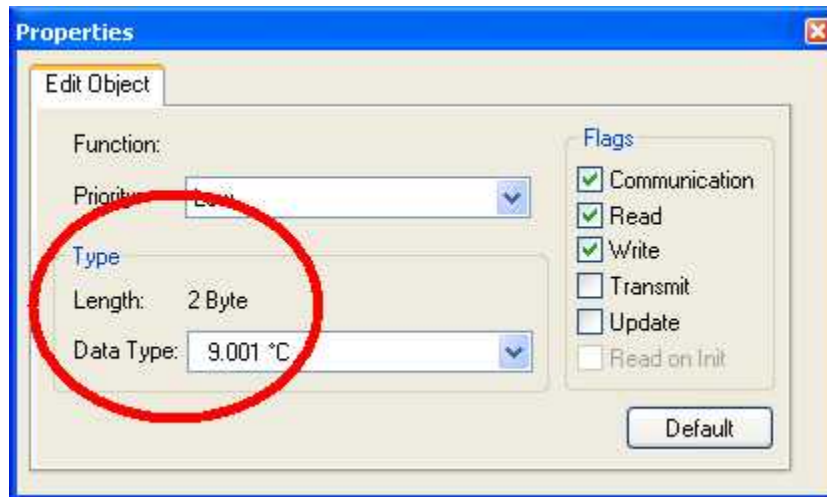
Address field of numeric gates must comply to the following format:

RGA;WGA;LGA1,LGA2,...;type

where

- *RGA*, *WGA*, *LGA1*, *LGA2*, ... are read, write and listening group address described above; none is mandatory, but you need to specify almost one of the three. Group addresses can be the same or can be different (one gate can read from a group address, write to a different group address and listen to different too group addresses). *RGA*, *WGA* and *LGA* list must be separated by *semicolon* character (*:*), each *LGA* is separated from the next with a *comma* character (*,*). Group addresses must be written in the standard 3-level representation: *main_group/middle_group/sub_group*.
- *type* is the data format of group addresses. It is the *main number (Format + Encoding)* of *Datapoint Type* (see chapter 3.7.2 of KNX Handbook). You can know the type of a group address (type of datapoint linked to a group address) referring to device documentation; another way is to see datapoint type in ETS: choose the group address in the left pane of *Group Addresses* window, then double click one of the linked datapoints in the right side; you need only the *main number*, that is the left part before *dot* character (*.*); in the figure below, *type* of datapoint is 9 (nine).

In the next paragraphs are listed all supported types and some examples of gate address field syntax.



Supported KNX Datapoint Types

Following table lists Datapoint Type supported for numeric gates. Reference of this table is chapter 3.7.2 v1.5.00 of KNX Handbook; this part of official documentation is available in *Download* section of KNX web site (<http://www.knx.org/downloads-support/downloads/>).

KNX standard is evolving and often are added new Datapoint Types. So you can correctly interface with a datapoint (group address) even if you can not find his type in the 3rd column of the table.

Datapoint Type main number (type)	Datapoint Type format	Supported Datapoint Types	Notes
1	B ₁	1.001 ... 1.023	<p>Reading and listening: gate value is set to 1 if received 1, 0 otherwise.</p> <p>Writing: 1 is written on the bus if least significant bit of gate is 1, 0 otherwise</p>
2	B ₂	2.001 ... 2.012	<p>Reading and listening: gate value is set according to the two received bits (bit 0 of gate ← v, bit 1 ← c).</p> <p>Writing: the two least significant bits of the gate are written on the bus (v ← bit 0 of gate, c ← bit 1).</p>
3	B ₁ U ₃	3.007, 3.008	<p>Reading and listening: gate value is set according to the four received bits (bit 0</p>

Datapoint Type main number (type)	Datapoint Type format	Supported Datapoint Types	Notes
			<p>... bit 2 of the gate ← <i>step-code</i>, bit 3 ← <i>c</i>).</p> <p>Writing: the four least significant bits of the gate are written on the bus (<i>step-code</i> ← bit 0 ... bit 2 of the gate, <i>c</i> ← bit 3).</p>
4	A ₈ Character set	4.001, 4.002	<p>Reading and listening: gate value is set according ISO-8859-1 code of received character.</p> <p>Writing: the eight least significant bits of the gate are written on the bus. If you are writing a 4.001 Datapoint Type, pay attention not to use values bigger than 127.</p>
5	U ₈ 8-bit Unsigned value	5.001, 5.003 ... 5.006, 5.010	<p>Reading and listening: gate value is set according to received value. No automatic scaling is provided: value is saved always as raw byte value (from 0 to 255). If you need scaling (e.g. for 5.001 and 5.003 Datapoint Type) please set conversion data in gate configuration.</p> <p>Writing: the eight least significant bits of the gate are written on the bus. No range check is provided; if the gate value is out of range (< 0 or > 255) writing results may be unpredictable.</p>
6	V ₈	6.001, 6.010	<p>Reading and listening: gate value is set according to</p>

Datapoint Type main number (type)	Datapoint Type format	Supported Datapoint Types	Notes
			received value. Writing: gate value is written on the bus. Out of range values (< -128 or > 127) generate writing error and the gate is not written.
7	U ₁₆ 2-octet Unsigned value	7.001, 7.002, 7.005, 7.007, 7.010 ... 7.013	Reading and listening: gate value is set according to received value. Writing: the sixteen least significant bits of the gate are written on the bus. No range check is provided; if the gate value is out of range (< 0 or > 65535) writing results may be unpredictable.
8	V ₁₆ 2-octet Signed value	8.001, 8.002, 8.005, 8.007, 8.010, 8.011	Reading and listening: gate value is set according to received value. No automatic scaling is provided: value is saved always as raw word value (from -32768 to 32767). If you need scaling (e.g. for 8.010 Datapoint Type) please set conversion data in gate configuration. Writing: gate value is written on the bus. Out of range values (< -32768 or > 32767) generate writing error and the gate is not written.
9	F ₁₆ 2-octet Float value	9.001 ... 9.008, 9.010, 9.011, 9.020 ... 9.028	Reading and listening: gate value is set according to received value. Writing: gate value is written on the bus. Out

Datapoint Type main number (type)	Datapoint Type format	Supported Datapoint Types	Notes
			of range values (< -671088.64 or > 670433.28) generate writing error and the gate is not written.
10	$N_3U_5r_2U_6r_2U_6$ Time	10.001	<p>Reading and listening: gate value is set according to received value. Received data is roughly copied in the three least significant bytes of the gate as described in KNX documentation.</p> <p>Writing: gate value is written on the bus. Only the three least significant bytes of the gate are transmitted, reserved bits are reset.</p>
11	$r_3U_5r_4U_4r_1U_7$ Date	11.001	<p>Reading and listening: gate value is set according to received value. Received data is roughly copied in the three least significant bytes of the gate as described in KNX documentation.</p> <p>Writing: gate value is written on the bus. Only the three least significant bytes of the gate are transmitted, reserved bits are reset.</p>
12	U_{32} 4-octet Unsigned value	12.001	<p>Reading and listening: gate value is set according to received value.</p> <p>Writing: gate value is written on the bus.</p>
13	V_{32} 4-octet Signed value	13.001, 13.010 ... 13.015	<p>Reading and listening: gate value is set according to</p>

Datapoint Type main number (type)	Datapoint Type format	Supported Datapoint Types	Notes
			received value. Writing: gate value is written on the bus.
14	F ₃₂ 4-octet Float value	14.000 ... 14.079	Reading and listening: gate value is set according to received value. Writing: gate value is written on the bus.
17	r ₂ U ₆ Scene number	17.001	Reading and listening: gate value is set according to received value. Received data is roughly copied in the least significant byte of the gate as described in KNX documentation. Writing: gate value is written on the bus. Only the least significant byte of the gate is transmitted, reserved bits are reset.
18	B ₁ r ₁ U ₆ Scene control	18.001	Reading and listening: gate value is set according to received value. Received data is roughly copied in the least significant byte of the gate as described in KNX documentation. Writing: gate value is written on the bus. Only the least significant byte of the gate is transmitted, reserved bits are reset.
20	N ₈	20.001 ... 20.008, 20.011 ... 20.014, 20.017	Reading and listening: gate value is set according to received value. Writing: the eight least significant bits of the

Datapoint Type main number (type)	Datapoint Type format	Supported Datapoint Types	Notes
			gate are written on the bus. No value check is provided; out of range or restricted values are not rejected.
21	B ₈	21.001, 21.002	<p>Reading and listening: gate value is set according to received value.</p> <p>Writing: the eight least significant bits of the gate are written on the bus. No value check is provided; out of range or restricted values are not rejected.</p>
23	N ₂	23.001 ... 23.003	<p>Reading and listening: gate value is set according to the two received bits; they are saved in the two least significant bits of the gate.</p> <p>Writing: the two least significant bits of the gate are written on the bus</p>
26	r ₁ B ₁ U ₆ Scene control	26.001	<p>Reading and listening: gate value is set according to received value. Received data is roughly copied in the least significant byte of the gate as described in KNX documentation.</p> <p>Writing: gate value is written on the bus. Only the least significant byte of the gate is transmitted, reserved bits are reset.</p>

Examples

Address field of the gate	Description
1/1/12;;;1	Read group address (RGA): 1/1/12 Write group address (WGA): none Listening group addresses (LGA): none KNX type (type): 1 (B ₁)
1/2/7;1/2/7;;5	Read group address (RGA): 1/2/7 Write group address (WGA): 1/2/7 Listening group addresses (LGA): none KNX type (type): 5 (U ₈ 8-bit Unsigned value)
1/2/7;;;1/2/7;5	Read group address (RGA): 1/2/7 Write group address (WGA): none Listening group addresses (LGA): 1/2/7 KNX type (type): 5 (U ₈ 8-bit Unsigned value)
1/2/7;1/2/8;1/2/7,1/2/9;9	Read group address (RGA): 1/2/7 Write group address (WGA): 1/2/8 Listening group addresses (LGA): 1/2/7 and 1/2/9 KNX type (type): 9 (F ₁₆ 2-octet Float value)
;1/2/8;;1	Read group address (RGA): none Write group address (WGA): 1/2/8 Listening group addresses (LGA): none KNX type (type): 1 (B ₁)

Blocks of numeric gates

Blocks of numeric gates are not supported.

14.3 Digital gates

You can link a gate to:

- one read group address (**RGA**)
- one write group address (**WGA**)
- one or more listening group addresses (**LGA1**, **LGA2**, ...)

RGA is the group address read synchronously from the bus (PC asks the bus); value answered by the bus becomes the gate value. **RGA** must be a readable group address; a gate may not have a read group address.

WGA is the group address written synchronously on the bus (PC sets the value on the bus). **WGA** must be a writeable group address; a gate may not have a write group address.

LGA1, **LGA2**, ... are the listening group addresses. When one of these group address passes through the bus, value is captured and becomes the gate value. This is due to the multi-master architecture of KNX protocol: each device can read, write or transmit group addresses (e.g. a light switch can write the value of a group address, a room sensor can transmit on the bus the changed value of temperature). A gate may have no listening group address and can be linked to a maximum of eight group addresses. Different gates can listen to same group addresses.

Digital gates address

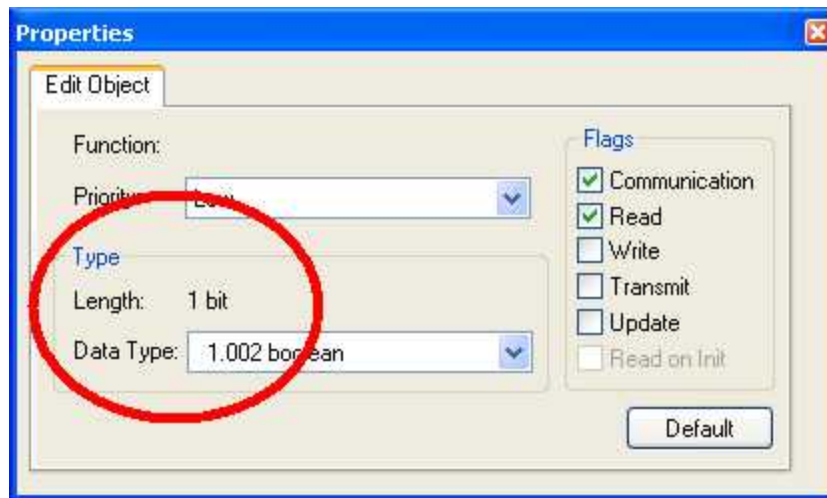
Address field of digital gates must comply to the following format:

$RGA;WGA;LGA1,LGA2,\dots;type$

where

- $RGA, WGA, LGA1, LGA2, \dots$ are read, write and listening group address described above; none is mandatory, but you need to specify almost one of the three. Group addresses can be the same or can be different (one gate can read from a group address, write to a different group address and listen to different too group addresses). RGA, WGA and LGA list must be separated by *semicolon* character ($;$), each LGA is separated from the next with a *comma* character ($,$). Group addresses must be written in the standard 3-level representation: $main_group/middle_group/sub_group$.
- $type$ is the data format of group addresses. It is the *main number (Format + Encoding)* of *Datapoint Type* (see chapter 3.7.2 of KNX Handbook). You can know the type of a group address (type of datapoint linked to a group address) referring to device documentation; another way is to see datapoint type in ETS: choose the group address in the left pane of *Group Addresses* window, then double click one of the linked datapoints in the right side; you need only the *main number*, that is the left part before *dot* character ($.$); in the figure below, $type$ of datapoint is 1 (one).

In the next paragraphs are listed all supported types and some examples of gate address field syntax.



Supported KNX Datapoint Types

Following table lists Datapoint Type supported for digital gates. Reference of this table is chapter 3.7.2 v1.5.00 of KNX Handbook; this part of official documentation is available in *Download* section of KNX web site (<http://www.knx.org/downloads-support/downloads/>).

KNX standard is evolving and often are added new Datapoint Types. So you can correctly interface with a datapoint (group address) even if you can not find his type in the 3rd column of the table.

Datapoint Type main number (<i>type</i>)	Datapoint Type format	Supported Datapoint Types	Notes
1	B ₁	1.001 ... 1.023	Reading and listening: gate value is set according to received value.

Datapoint Type main number (type)	Datapoint Type format	Supported Datapoint Types	Notes
			Writing: gate value is written on the bus.

Examples

Address field of the gate	Description
1/3/173;;;1	Read group address (RG A): 1/3/173 Write group address (W G)A): none Listening group addresses (L G)A): none KNX type (type): 1 (B ₁)
1/3/173;1/3/173;;1	Read group address (RG A): 1/3/173 Write group address (W G)A): 1/3/173 Listening group addresses (L G)A): none KNX type (type): 1 (B ₁)
1/3/173;;;1/3/173;1	Read group address (RG A): 1/3/173 Write group address (W G)A): none Listening group addresses (L G)A): 1/3/173 KNX type (type): 1 (B ₁)
2/1/14;2/1/16;2/1/13,2/1/15;1	Read group address (RG A): 2/1/14 Write group address (W G)A): 2/1/16 Listening group addresses (L G)A): 2/1/13 and 2/1/15 KNX type (type): 1 (B ₁)
;3/2/1;;1	Read group address (RG A): none Write group address (W G)A): 3/2/1 Listening group addresses (L G)A): none KNX type (type): 1 (B ₁)

Blocks of digital gates

Blocks of digital gates are not supported.

14.4 String gates

You can link a gate to:

- one read group address (**RG**A)
- one write group address (**W**G)A)
- one or more listening group addresses (**L**G)A₁, **L**G)A₂, ...)

RGA is the group address read synchronously from the bus (PC asks the bus); value answered by the bus becomes the gate value. **RG**A must be a readable group address; a gate may not have a read group address.

WG)A is the group address written synchronously on the bus (PC sets the value on the bus). **W**G)A

must be a writeable group address; a gate may not have a write group address.

LGA1, LGA2, ... are the listening group addresses. When one of these group address passes through the bus, value is captured and becomes the gate value. This is due to the multi-master architecture of KNX protocol: each device can read, write or transmit group addresses (e.g. a light switch can write the value of a group address, a room sensor can transmit on the bus the changed value of temperature). A gate may have no listening group address and can be linked to a maximum of eight group addresses. Different gates can listen to same group addresses.

String gates address

Address field of string gates must comply to the following format:

RGA;WGA;LGA1,LGA2,...;type

where

- *RGA, WGA, LGA1, LGA2, ...* are read, write and listening group address described above; none is mandatory, but you need to specify almost one of the three. Group addresses can be the same or can be different (one gate can read from a group address, write to a different group address and listen to different too group addresses). *RGA, WGA* and *LGA* list must be separated by *semicolon* character (;), each *LGA* is separated from the next with a *comma* character (,). Group addresses must be written in the standard 3-level representation: *main_group/middle_group/sub_group*.
- *type* is the data format of group addresses. It is the *main number (Format + Encoding)* of *Datapoint Type* (see chapter 3.7.2 of KNX Handbook). You can know the type of a group address (type of datapoint linked to a group address) referring to device documentation; another way is to see datapoint type in ETS: choose the group address in the left pane of *Group Addresses* window, then double click one of the linked datapoints in the right side; you need only the *main number*, that is the left part before *dot* character (.); in the figure below, *type* of datapoint is 16 (sixteen).

In the next paragraphs are listed all supported types and some examples of gate address field syntax.



Supported KNX Datapoint Types

Following table lists Datapoint Type supported for string gates. Reference of this table is chapter 3.7.2 v1.5.00 of KNX Handbook; this part of official documentation is available in *Download* section of KNX web site (<http://www.knx.org/downloads-support/downloads/>).

KNX standard is evolving and often are added new Datapoint Types. So you can correctly interface with a datapoint (group address) even if you can not find his type in the 3rd column of the table.

Datapoint Type main number (type)	Datapoint Type format	Supported Datapoint Types	Notes
4	A ₈ Character set	4.001, 4.002	<p>Reading and listening: gate value is set according to received character.</p> <p>Writing: first character of gate is written on the bus. If you are writing a 4.001 Datapoint Type, pay attention not to use character which code is bigger than 127.</p>
16	A ₁₁₂ String	16.001, 16.002	<p>Reading and listening: gate value is set according to received characters.</p> <p>Writing: first 14 characters of gate are written on the bus. If gate string length is smaller than 14, remaining characters are reset. If you are writing a 16.001 Datapoint Type, pay attention not to use character which code is bigger than 127.</p>
24	A[n]	24.001	<p>Reading and listening: gate value is set according to received characters (maximum 80 characters).</p> <p>Writing: gate string characters are written on the bus.</p>

Examples

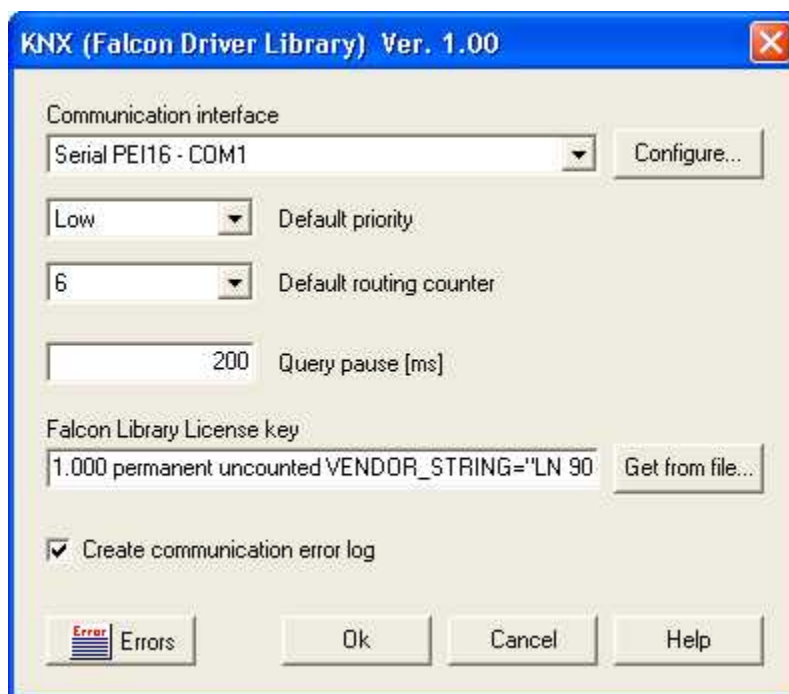
Address field of the gate	Description
4/12/1;;;4	<p>Read group address (RGA): 4/12/1</p> <p>Write group address (WGA): none</p>

Address field of the gate	Description
	Listening group addresses (LGA): none KNX type (type): 4 (A ₈ Character set)
4/12/1;4/12/1;;16	Read group address (RGA): 4/12/1 Write group address (WGA): 4/12/1 Listening group addresses (LGA): none KNX type (type): 16 (A ₁₂ String)
4/12/1;;4/12/1;16	Read group address (RGA): 4/12/1 Write group address (WGA): none Listening group addresses (LGA): 4/12/1 KNX type (type): 16 (A ₁₂ String)
5/1/10;5/1/11;5/1/12,5/1/17;4	Read group address (RGA): 5/1/10 Write group address (WGA): 5/1/11 Listening group addresses (LGA): 5/1/12 e 5/1/17 KNX type (type): 4 (A ₈ Character set)
;3/20/1;;16	Read group address (RGA): none Write group address (WGA): 3/20/1 Listening group addresses (LGA): none KNX type (type): 16 (A ₁₂ String)

Blocks of string gates

Blocks of string gates are not supported.

14.5 Configuration



Protocol configuration window.

Communication interface: choose interface that will be used to communicate with KNX devices.

Listed interfaces are system-defined (same interfaces are available to all KNX software running on the machine, like ETS). *Configure...* button allow to create, edit and remove communication interfaces (it is the same dialog of ETS).

Default priority e Default routing counter: these are parameters used to communicate with KNX bus. Please use default settings (*Low* and *6*) unless you are aware of change effects. For a detailed description of these parameters refer to KNX Handbook.

Query pause: this is the minimum time waited between a bus request and the next. Please use a sensible time; a too short time can cause bus malfunctions.

License key: starting from Falcon Library v2.0 leave this field blank; otherwise insert here Falcon Library license acquired from KNX Association. License key can be obtained from license file downloaded from KNX online shop. Push *Get from file...* button and select your license key file; KNX online shop normally provide a *.zip* file; unzip it to get the license file (which has *.lic* extension).

Alternatively you can manual insert your license key. Open your license file with a text editor (like *Notepad*) and extract the license key; it is a string like this:

```
1.000 permanent uncounted VENDOR_STRING="xxxxxxxx" HOSTID=FLEXID=
yyyyyyyy SIGN=zzzzzzzz
```

Other information about Falcon Library licensing are provided in introduction page.

Create communication error log: if set, during runtime, protocol driver will create an error log file. To display error log press *Errors* button (during runtime as well). Error log is useful to debug projects helping you to find causes of communication errors.

Falcon Library creates and manages other machine-wide log files; they are located in

Program Common Files dir\EIBA sc\Log (normally

C:\Program Files\Common Files\EIBA sc\Log). Please note that these log files contain entries related to all programs accessing KNX bus (ETS too).

15 M-BUS (METER-BUS)

15.1 Introduction

Protocol driver designed to allow communication (read-only) with devices supporting **M-Bus**, according to **M-Bus Usergroup** (<http://www.m-bus.com/>) specification and European Standard **EN13757-2** (physical and link layer) and **EN 13757-3** (application layer).

M-Bus interfacing requires a serial, USB or Ethernet adapter (*level converter*). Using an USB or Ethernet adapter requires a Virtual COM driver, so PC can access them as a standard COM port.

PC communicates with M-Bus devices sending them *REQ_UD2* requests. It supports *SND_UD* replies in the following formats:

- **Fixed Data Structure:** Little endian (Mode 1, CI = 73h) or Big endian (Mode 2, CI = 77h)
- **Variable Data Structure** with 12 bytes long *Data Header* and without data encryption: Little endian (Mode 1, CI = 72h) or Big endian (Mode 2, CI = 76h); *multi telegram readout* is supported

Driver supports both primary addressing and secondary addressing (only *Identification number*). To use primary address, it is enough to specify it in the *Device* field of gates configuration. To use secondary address you should fill table in the protocol configuration dialog.

Writing data to devices is not supported.

Communication between PC and M-Bus devices is based on the *REQ_UD2* request and the *SND_UD* response which contains values of all device variables. To avoid that for each gate reading bytes are transmitted vainly and to reduce communication time, it is recommended to group gates in reading

blocks. Moreover because M-Bus communication baud rate is often very low (2400 and 300 bps).

Some M-Bus devices are powered by batteries and to reduce consumption they can restrict bus access (i.e. they allow a limited number of transaction for each day). In this case it is mandatory using reading blocks and to reduce frequency of data polling.

15.2 Numeric gates

To read data from M-Bus devices, supervisor inquires them using *REQ_UD2* message; devices reply with a *SND_UD* telegram. *SND_UD* message is composed by an header (which contains some device info) and a set of *data blocks* (records), one for each variable. Each record contains the value of the variable and other related info (measure unit, scale, ...)

Header
Device variables
Record 1
Value
Other info
Record 2
Value
Other info
: :
Record n
Value
Other info

Using numeric gates, it is possible to retrieve data from header (*device variables*) and data (value and other info) from each record (*record variables*). To identify the wanted record you need to know the order in which records are arranged in the response message (ask response format to device manufacturer). For example Record 3 is the 3rd record of the response.

Numeric gates address to retrieve *device variables*

To read device variables, *Address* field of numeric gates must comply to the following format:

var_id

where *var_id* can be one of the following

var_id	Description	Notes
idNo	Identification number	Converted from BCD to binary
accessNo	Access number	-
status	Status	-
medium	Medium	-
manufacturer	Manufacturer	16 bit value as retrieved from response message It is not available for devices that reply with Fixed Format Structure message
version	Version/generation	It is not available for devices that reply with Fixed Format Structure message

Numeric gates address to retrieve *record variables*

To read record variables, *Address* field of numeric gates must comply to the following format:

rec_no.var_id

where

- *rec_no* is the number of the record as it appears in response message (for example if you want the 3rd record of the response, type 3). To know the order in which records are arranged in the response message ask format of SND_UD message to device manufacturer. You can insert record no in decimal notation or hexadecimal notation (use prefix 0x or postfix the number with the character h); for example to select 23rd record, type 23 or 0x17 or 17h. For devices that reply with Fixed Format Structure message are allowed only records 1 and 2. If you require a unavailable record, gate will be set in communication error.
- *var_id* is the identification of which information has to be retrieved. Can be one of the following:

var_id	Description	Notes
value	Value	Value of the record as retrieved form response message. Decoding of "complex" value types (<i>Type F, G, I, J, K</i>), must be performed manually (using scripting language). <i>Type L</i> records are unsupported and generate reading errors. Reading of text type records (<i>data field = 1101b</i> and <i>LVAR ≤ BFh</i>) generates error. It is preferred to use normalized value (see normValue).
normValue	Normalized value	"Normalized" value of the record. Value is converted in a predefined measure unit. This is useful for devices which rescale variables dynamically. Records of <i>Type F, G, I, J</i> , when contain a date and time value, are converted in a format compatible with date/time function of scripting language (number of seconds from midnight of 01/01/1901). When they contain only time information, are converted in the number of seconds from midnight. <i>Type K</i> records are not converted. <i>Type L</i> records are unsupported and generate reading errors. Reading of text type records (<i>data field = 1101b</i> and <i>LVAR ≤ BFh</i>) generates error. See tables of next paragraph to see detailed info how records values are normalized.
unit	Measure unit	Measure unit or type of info contained in the record. For devices that reply with Fixed Format Structure message, it is the <i>physical unit</i> field of the record (counter). For other devices, it is a "sum" of the <i>VIF</i> and <i>VIFE</i> of the record. It is a 16 bit value, where most significant byte is:

var_id	Description	Notes
		<ul style="list-style-type: none"> - <i>00h</i> if the record has a <i>Primary VIF</i> (<i>VIF</i> = e0000000b ... e1111010b). Least significant byte is <i>VIF</i> value (extension bit <i>e</i> is reset) - <i>01h</i> if record has a <i>FBh Linear VIF-Extension</i> (<i>VIF</i> = 11111011b). Least significant byte is the value of the first <i>VIFE</i> (extension bit <i>e</i> is reset) - <i>02h</i> if record has a <i>FDh Linear VIF-Extension</i> (<i>VIF</i> = 11111010b). Least significant byte is the value of the first <i>VIFE</i> (extension bit <i>e</i> is reset) - <i>03h</i> if the record uses one the <i>non metric units</i> (<i>VIF</i> = e0000000b ... e1111010b, <i>VIFE</i> = e0111101b). Least significant byte is <i>VIF</i> value (extension bit <i>e</i> is reset). - <i>04h</i> if record has a <i>Plain-text VIF</i> (<i>VIF</i> = e1111010b). Least significant byte is always <i>00h</i>. - <i>FFh</i> if record has a <i>Manufacturer specific VIF</i> (<i>VIF</i> = e1111111b). Least significant byte is always <i>00h</i>. <p>Use string gates to obtain a textual representation of measure unit.</p>
extUnit1 ... extUnit10	Measure unit extensions	<p>All the record <i>VIFE</i>s not used to determine unit. All extension bit <i>e</i> are reset. Unused <i>extUnit_i</i> are set to <i>00h</i>.</p> <p>Use string gates to obtain a textual representation of measure unit extensions. They are not available for devices that reply with Fixed Format Structure message.</p>
storageNo	Storage number	<p>Value of <i>Storage number</i> field contained in the <i>DIF</i> and in the <i>DIFE</i>s of the record. It is not available for devices that reply with Fixed Format Structure message.</p>
tariff	Tariff	<p>Value of <i>Tariff</i> field contained in the <i>DIF</i> and in the <i>DIFE</i>s of the record. It is not available for devices that reply with Fixed Format Structure message.</p>
subunit	Subunit	<p>Value of <i>Subunit</i> field contained in the the <i>DIFE</i>s of the record. It is not available for devices that reply with Fixed Format Structure message.</p>
function	Function	<p>Value of <i>Function</i> field contained in the the <i>DIF</i> of the record. It is not available for devices that reply with Fixed Format Structure message.</p>

Omitting *var_id* (and the separator character) and specifying only *rec_no*, the numeric gate gets the normalized value.

Value normalization for Fixed Format Structure response message

Following table shows how values and measure units are normalized. For each possible *physical unit* (its value is readable via `unit`) it is indicated measure unit, measure unit at which value is normalized and the multiplier factor used in normalization.

Physical unit	Type of quantity	Unità di misura	Normalized measure unit	unit	Multiplier
00 0000	Time	hh:m m:ss	hh:m m:ss	00h	1
00 0001	Date	D/M/Y	D/M/Y	01h	1
00 0010	Energy	Wh	kWh	02h	10 ⁻³
00 0011		10 mWh	kWh	03h	10 ⁻²
00 0100		100 mWh	kWh	04h	10 ⁻¹
00 0101		kWh	kWh	05h	1
00 0110		10 kWh	kWh	06h	10
00 0111		100 kWh	kWh	07h	10 ²
00 1000		MWh	kWh	08h	10 ³
00 1001		10 MWh	kWh	09h	10 ⁴
00 1010		100 MWh	kWh	0Ah	10 ⁻³
00 1011		Energy	kJ	MJ	0Bh
00 1010	10 kJ		MJ	0Ch	10 ⁻¹
00 1011	100 kJ		MJ	0Dh	1
00 1110	MJ		MJ	0Eh	10
00 1111	10 MJ		MJ	0Fh	10 ²
01	100		MJ	10h	10 ³

Physical unit	Type of quantity	Unità di misura	Normalized measure unit	unit	Multiplier
10 0000	Power	MJ/h	MJ/h	20h	1
10 0001		10 MJ/h	MJ/h	21h	10
10 0010		100 MJ/h	MJ/h	22h	10 ²
10 0011		GJ/h	MJ/h	23h	10 ³
10 0100		10 GJ/h	MJ/h	24h	10 ⁴
10 0101		100 GJ/h	MJ/h	25h	10 ⁵
10 0110	Volume	ml	l	26h	10 ⁻³
10 0111		10 ml	l	27h	10 ⁻²
10 1000		100 ml	l	28h	10 ⁻¹
10 1001		l	l	29h	1
10 1010		10 l	l	2Ah	10
10 1011		100 l	l	2Bh	10 ²
10 1100	Volume flow	m ³	l	2Ch	10 ³
10 1101		10 m ³	l	2Dh	10 ⁴
10 1110		100 m ³	l	2Eh	10 ⁵
10 1111		ml/h	l/h	2Fh	10 ⁻³
11		10	l/h	30h	10 ⁻²

0000		MJ				
01 0001		GJ	MJ	11h	10 ⁴	
01 0010		10 GJ	MJ	12h	10 ⁻³	
01 0011		100 GJ	MJ	13h	10 ⁻²	
01 0100	Power	W	kW	14h	10 ⁻³	
01 0101		10 W	kW	15h	10 ⁻²	
01 0110		100 W	kW	16h	10 ⁻¹	
01 0111		kW	kW	17h	1	
01 1000		10 kW	kW	18h	10	
01 1001		100 kW	kW	19h	10 ²	
01 1010		MW	kW	1Ah	10 ³	
01 1011		10 MW	kW	1Bh	10 ⁴	
01 1100		100 MW	kW	1Ch	10 ⁵	
01 1101		Power	kJ/h	MJ/h	1Dh	10 ⁻³
01 1110			10 kJ/h	MJ/h	1Eh	10 ⁻²
01 1111			100 kJ/h	MJ/h	1Fh	10 ⁻¹

0000		ml/h			
11 0001		100 ml/h	l/h	31h	10 ⁻¹
11 0010		l/h	l/h	32h	1
11 0011		10 l/h	l/h	33h	10
11 0100		100 l/h	l/h	34h	10 ²
11 0101		m ³ /h	l/h	35h	10 ³
11 0110		10 m ³ /h	l/h	36h	10 ⁴
11 0111		100 m ³ /h	l/h	37h	10 ⁵
11 1000	Temper ature	m°C	°C	38h	10 ⁻³
11 1001	Units for HCA			39h	1
11 1010	reserved			3Ah	
11 1011				3Bh	
11 1100				3Ch	
11 1101				3Dh	
11 1110		same but historic			3Eh
11 1111	without unit			3Fh	1

- note 1f: only for record 2 (counter 2). It means that record 2 has the same measure unit of record 1, but it has a historic value. Its value is normalized using record 1 specification.

Value normalization for Variable Format Structure response message

Following tables show how values and measure units are normalized.

The four tables refer respectively to record with *Primary VIF*, *FBh Linear VIF Extension*, *FDh Linear VIF Extension* and non metric units. For each possible case, it is indicated measure unit, measure unit at which value is normalized and the multiplier factor used in normalization.

Primary VIF

VIF code	Type of quantity	Measure unit	Normalized measure unit	unit	Multiplier
e0000000	Energy	mWh	Wh	0000h	10 ⁻³
e0000001		10 mWh	Wh	0001h	10 ⁻²
e0000010		100 mWh	Wh	0002h	10 ⁻¹
e0000011		Wh	Wh	0003h	1
e0000100		10 Wh	Wh	0004h	10
e0000101		100 Wh	Wh	0005h	10 ²
e0000110		kWh	Wh	0006h	10 ³
e0000111		10 kWh	Wh	0007h	10 ⁴
e0001000		Energy	J	kJ	0008h
e0001001	10 J		kJ	0009h	10 ⁻²
e0001010	100 J		kJ	000Ah	10 ⁻¹
e0001011	kJ		kJ	000Bh	1
e0001010	10 kJ		kJ	000Ch	10
e0001011	100 kJ		kJ	000Dh	10 ²
e0001110	MJ		kJ	000Eh	10 ³
e0001111	10 MJ		kJ	000Fh	10 ⁴
e0010000	Volume		ml	l	0010h
e0010001		10 ml	l	0011h	10 ⁻²
e0010010		100 ml	l	0012h	10 ⁻¹

VIF code	Type of quantity	Measure unit	Normalized measure unit	unit	Multiplier
e1000000	Volume flow	0.1 ml/h	l/min	0040h	10 ⁻⁴
e1000001		ml/min	l/min	0041h	10 ⁻³
e1000010		10 ml/m	l/min	0042h	10 ⁻²
e1000011		100 ml/	l/min	0043h	10 ⁻¹
e1000100		l/min	l/min	0044h	1
e1000101		10 l/mi	l/min	0045h	10
e1000110		100 l/min	l/min	0046h	10 ²
e1000111		m ³ /min	l/min	0047h	10 ³
e1001000		Volume flow	µl/s	l/s	0048h
e1001001	10 µl/s		l/s	0049h	10 ⁻⁵
e1001010	100 µl/s		l/s	004Ah	10 ⁻⁴
e1001011	ml/s		l/s	004Bh	10 ⁻³
e1001100	10 ml/s		l/s	004Ch	10 ⁻²
e1001101	100 ml/s		l/s	004Dh	10 ⁻¹
e1001110	l/s		l/s	004Eh	1
e1001111	10 l/s		l/s	004Fh	10
e1010000	Mass flow		g/h	kg/h	0050h
e1010001		10 g/h	kg/h	0051h	10 ⁻²
e1010010		100 g/h	kg/h	0052h	10 ⁻¹

e001 0011		l	l	001 3h	1
e001 0100		10 l	l	001 4h	10
e001 0101		100 l	l	001 5h	10 ²
e001 0110		m ³	l	001 6h	10 ³
e001 0111		10 m ³	l	001 7h	10 ⁴
e001 1000	Mass	g	kg	001 8h	10 ⁻³
e001 1001		10 g	kg	001 9h	10 ⁻²
e001 1010		100 g	kg	001 Ah	10 ⁻¹
e001 1011		kg	kg	001 Bh	1
e001 1100		10 kg	kg	001 Ch	10
e001 1101		100 kg	kg	001 Dh	10 ²
e001 1110		t	kg	001 Eh	10 ³
e001 1111		10 t	kg	001 Fh	10 ⁴
e010 0000	On time	s	s	002 0h	1
e010 0001		min	s	002 1h	60
e010 0010		h	s	002 2h	3600
e010 0011		d	s	002 3h	86400
e010 0100	Operating time	s	s	002 4h	1
e010 0101		min	s	002 5h	60
e010 0110		h	s	002 6h	3600
e010 0111		d	s	002 7h	86400
e010 1000	Power	mW	W	002 8h	10 ⁻³

e101 0011		kg/h	kg/h	005 3h	1
e101 0100		10 kg/h	kg/h	005 4h	10
e101 0101		100 kg/h	kg/h	005 5h	10 ²
e101 0110		t/h	kg/h	005 6h	10 ³
e101 0111		10 t/h	kg/h	005 7h	10 ⁴
e101 1000	Flow temperature	m°C	°C	005 8h	10 ⁻³
e101 1001		10 m°C	°C	005 9h	10 ⁻²
e101 1010		100 m°C	°C	005 Ah	10 ⁻¹
e101 1011		°C	°C	005 Bh	1
e101 1100	Return temperature	m°C	°C	005 Ch	10 ⁻³
e101 1101		10 m°C	°C	005 Dh	10 ⁻²
e101 1110		100 m°C	°C	005 Eh	10 ⁻¹
e101 1111		°C	°C	005 Fh	1
e110 0000	Temperature difference	m°C	°C	006 0h	10 ⁻³
e110 0001		10 m°C	°C	006 1h	10 ⁻²
e110 0010		100 m°C	°C	006 2h	10 ⁻¹
e110 0011		°C	°C	006 3h	1
e110 0100	External temperature	m°C	°C	006 4h	10 ⁻³
e110 0101		10 m°C	°C	006 5h	10 ⁻²
e110 0110		100 m°C	°C	006 6h	10 ⁻¹
e110 0111		°C	°C	006 7h	1
e110 1000	Pressure	mbar	bar	006 8h	10 ⁻³

e010 1001		10 mW	W	002 9h	10 ⁻²	
e010 1010		100 mW	W	002 Ah	10 ⁻¹	
e010 1011		W	W	002 Bh	1	
e010 1100		10 W	W	002 Ch	10	
e010 1101		100 W	W	002 Dh	10 ²	
e010 1110		kW	W	002 Eh	10 ³	
e010 1111		10 kW	W	002 Fh	10 ⁴	
e011 0000		Power	J/h	kJ/h	003 0h	10 ⁻³
e011 0001	10 J/h		kJ/h	003 1h	10 ⁻²	
e011 0010	100 J/h		kJ/h	003 2h	10 ⁻¹	
e011 0011	kJ/h		kJ/h	003 3h	1	
e011 0100	10 kJ/h		kJ/h	003 4h	10	
e011 0101	100 kJ/h		kJ/h	003 5h	10 ²	
e011 0110	MJ/h		kJ/h	003 6h	10 ³	
e011 0111	10 MJ/h		kJ/h	003 7h	10 ⁴	
e011 1000	Volume flow		ml/h	l/h	003 8h	10 ⁻³
e011 1001			10 ml/h	l/h	003 9h	10 ⁻²
e011 1010			100 ml/h	l/h	003 Ah	10 ⁻¹
e011 1011		l/h	l/h	003 Bh	1	
e011 1100		10 l/h	l/h	003 Ch	10	
e011 1101		100 l/h	l/h	003 Dh	10 ²	
e011 1110		m ³ /h	l/h	003 Eh	10 ³	

e110 1001		10 mbar	bar	006 9h	10 ⁻²
e110 1010		100 mbar	bar	006 Ah	10 ⁻¹
e110 1011		bar	bar	006 Bh	1
e110 1100	Time point			006 Ch	(note 1v)
e110 1101				006 Dh	(note 1v)
e110 1110	Units for HCA			006 Eh	1
e110 1111	<i>reserved</i>			006 Fh	
e111 0000	Averagin g duration	s	s	007 0h	1
e111 0001		min	s	007 1h	60
e111 0010		h	s	007 2h	3600
e111 0011		d	s	007 3h	8640 0
e111 1000	Actuality duration	s	s	007 4h	1
e111 0101		min	s	007 5h	60
e111 0110		h	s	007 6h	3600
e111 0111		d	s	007 7h	8640 0
e111 1000	Fabricati on No.			007 8h	1
e111 1001	<i>reserved</i>			007 9h	
e111 1010				007 Ah	
e111 1011				007 Bh	
e111 1100				007 Ch	
e111 1101				007 Dh	
e111 1110				007 Eh	

e011 1111		10 m ³ /h	l/h	003 Fh	10 ⁴
--------------	--	----------------------	-----	-----------	-----------------

e111 1111				007 Fh	
--------------	--	--	--	-----------	--

Linear VIF Extension (VIF = FBh)

VIFE code	Type of quantity	Measure unit	Normalized measure unit	unit	Multiplier
e000 0000	Energy	100 kWh	MWh	010 0h	10 ⁻¹
e000 0001		MWh	MWh	010 1h	1
e000 0010	Reactive energy	kvarh	kvarh	010 2h	1
e000 0011		10 kvarh	kvarh	010 3h	10
e000 0100	reserved			010 4h	
e000 0101				010 5h	
e000 0110				010 6h	
e000 0111				010 7h	
e000 1000		Energy	100 MJ	GJ	010 8h
e000 1001	GJ		GJ	010 9h	1
e000 1010	reserved			010 Ah	
e000 1011				010 Bh	
e000 1100	Energy	100 kcal	Mcal	010 Ch	10 ⁻¹
e000 1101		Mcal	Mcal	010 Dh	1
e000 1110		10 Mcal	Mcal	010 Eh	10
e000 1111		100 Mcal	Mcal	010 Fh	10 ²
e001 0000	Volume	100 m ³	m ³	011 0h	10 ²

VIFE code	Type of quantity	Measure unit	Normalized measure unit	unit	Multiplier
e100 0000	reserved			014 0h	
e100 0001				014 1h	
e100 0010				014 2h	
e100 0011				014 3h	
e100 0100				014 4h	
e100 0101				014 5h	
e100 0110				014 6h	
e100 0111				014 7h	
e100 1000				014 8h	
e100 1001				014 9h	
e100 1010				014 Ah	
e100 1011	reserved			014 Bh	
e100 1100				014 Ch	
e100 1101				014 Dh	
e100 1110				014 Eh	
e100 1111			014 Fh		
e101 0000	reserved			015 0h	

e001 0001		1000 m ³	m ³	011 1h	10 ³
e001 0010	<i>reserved</i>			011 2h	
e001 0011				011 3h	
e001 0100				011 4h	
e001 0101				011 5h	
e001 0110				011 6h	
e001 0111				011 7h	
e001 1000		Mass	100 t	t	011 8h
e001 1001	1000 t		t	011 9h	10 ³
e001 1010	<i>reserved</i>			011 Ah	
e001 1011				011 Bh	
e001 1100				011 Ch	
e001 1101				011 Dh	
e001 1110				011 Eh	
e001 1111				011 Fh	
e010 0000		<i>reserved</i>			012 0h
e010 0001	Volume	0.1 ft ³	ft ³	012 1h	10 ⁻¹
e010 0010		0.1 gal	gal	012 2h	10 ⁻¹
e010 0011		gal	gal	012 3h	1
e010 0100	Volume flow	mgal/ min	gal/m in	012 4h	10 ⁻³
e010 0101		gal/m in	gal/m in	012 5h	1
e010 0110		gal/h	gal/h	012 6h	1

e101 0001				015 1h	
e101 0010				015 2h	
e101 0011				015 3h	
e101 0100				015 4h	
e101 0101				015 5h	
e101 0110				015 6h	
e101 0111				015 7h	
e101 1000	Flow temperat ure	m°F	°F	015 8h	10 ⁻³
e101 1001		10 m°F	°F	015 9h	10 ⁻²
e101 1010		100 m°F	°F	015 Ah	10 ⁻¹
e101 1011		°F	°F	015 Bh	1
e101 1100	Return temperat ure	m°F	°F	015 Ch	10 ⁻³
e101 1101		10 m°F	°F	015 Dh	10 ⁻²
e101 1110		100 °F	°F	015 Eh	10 ⁻¹
e101 1111		°F	°F	015 Fh	1
e110 0000	Temper ature differenc e	m°F	°F	016 0h	10 ⁻³
e110 0001		10 m°F	°F	016 1h	10 ⁻²
e110 0010		100 m°F	°F	016 2h	10 ⁻¹
e110 0011		°F	°F	016 3h	1
e110 0100	External temperat ure	m°F	°F	016 4h	10 ⁻³
e110 0101		10 m°F	°F	016 5h	10 ⁻²
e110 0110		100 m°F	°F	016 6h	10 ⁻¹

e010 0111	<i>reserved</i>			012 7h		
e010 1000	Power	100 kW	MW	012 8h	10 ⁻¹	
e010 1001		MW	MW	012 9h	1	
e010 1010	<i>reserved</i>			012 Ah		
e010 1011				012 Bh		
e010 1100				012 Ch		
e010 1101				012 Dh		
e010 1110				012 Eh		
e010 1111				012 Fh		
e011 0000		Power	100 MJ/h	GJ/h	013 0h	10 ⁻¹
e011 0001			GJ/h	GJ/h	013 1h	1
e011 0010	<i>reserved</i>			013 2h		
e011 0011				013 3h		
e011 0100				013 4h		
e011 0101				013 5h		
e011 0110				013 6h		
e011 0111				013 7h		
e011 1000		<i>reserved</i>			013 8h	
e011 1001					013 9h	
e011 1010				013 Ah		
e011 1011				013 Bh		
e011 1100				013 Ch		

e110 0111		°F	°F	016 7h	1	
e110 1000	<i>reserved</i>			016 8h		
e110 1001				016 9h		
e110 1010				016 Ah		
e110 1011				016 Bh		
e110 1100				016 Ch		
e110 1101				016 Dh		
e110 1110				016 Eh		
e110 1111				016 Fh		
e111 0000	Cold/war m temperat ure limit	m°F	°F	017 0h	10 ⁻³	
e111 0001		10 m°F	°F	017 1h	10 ⁻²	
e111 0010		100 m°F	°F	017 2h	10 ⁻¹	
e111 0011		°F	°F	017 3h	1	
e111 0100	Cold/war m temperat ure limit	m°C	°C	017 4h	10 ⁻³	
e111 0101		10 m°C	°C	017 5h	10 ⁻²	
e111 0110		100 m°C	°C	017 6h	10 ⁻¹	
e111 0111			°C	°C	017 7h	1
e111 1000		Cumulati ve count max power	mW	W	017 8h	10 ⁻³
e111 1001	10 mW		W	017 9h	10 ⁻²	
e111 1010	100 mW		W	017 Ah	10 ⁻¹	
e111 1011	W		W	017 Bh	1	
e111 1100	10 W		W	017 Ch	10	

e011 1101				013 Dh	
e011 1110				013 Eh	
e011 1111				013 Fh	

e111 1101		100 W	W	017 Dh	10 ²
e111 1110		kW	W	017 Eh	10 ³
e111 1111		10 kW	W	017 Fh	10 ⁴

Linear VIF Extension (VIF = FDh)

VIFE code	Type of quantity	Measure unit	Normalized measure unit	unit	Multiplier
e000 0000	Credit	0.001 curr.	currency	020 0h	10 ⁻³
e000 0001		0.01 curr.	currency	020 1h	10 ⁻²
e000 0010		0.1 curr.	currency	020 2h	10 ⁻¹
e000 0011		currency	currency	020 3h	1
e000 0100	Debit	0.001 curr.	currency	020 4h	10 ⁻³
e000 0101		0.01 curr.	currency	020 5h	10 ⁻²
e000 0110		0.1 curr.	currency	020 6h	10 ⁻¹
e000 0111		currency	currency	020 7h	1
e000 1000	Access number			020 8h	1
e000 1001	Medium			020 9h	1
e000 1010	Manufacturer			020 Ah	1
e000 1011	Parameter set id			020 Bh	1
e000 1100	Device type			020 Ch	1
e000 1101	Hardware version			020 Dh	1
e000	Firmware			020	1

VIFE code	Type of quantity	Measure unit	Normalized measure unit	unit	Multiplier
e100 0000	Voltage	nV	V	024 0h	10 ⁻⁹
e100 0001		10 nV	V	024 1h	10 ⁻⁸
e100 0010		100 nV	V	024 2h	10 ⁻⁷
e100 0011		μV	V	024 3h	10 ⁻⁶
e100 0100		10 μV	V	024 4h	10 ⁻⁵
e100 0101		100 μV	V	024 5h	10 ⁻⁴
e100 0110		mV	V	024 6h	10 ⁻³
e100 0111		10 mV	V	024 7h	10 ⁻²
e100 1000		100 mV	V	024 8h	10 ⁻¹
e100 1001		V	V	024 9h	1
e100 1010		10 V	V	024 Ah	10
e100 1011	100 V	V	024 Bh	10 ²	
e100 1100	kV	V	024 Ch	10 ³	
e100 1101	10 kV	V	024 Dh	10 ⁴	
e100	100	V	024	10 ⁵	

1110	e version			Eh	
e000 1111	Software version			020 Fh	1
e001 0000	Customer loc.			021 0h	1
e001 0001	Customer			021 1h	1
e001 0010	Access code User			021 2h	1
e001 0011	Access code Op.			021 3h	1
e001 0100	Access code System Op.			021 4h	1
e001 0101	Access code Dev.			021 5h	1
e001 0110	Password			021 6h	1
e001 0111	Error flags			021 7h	1
e001 1000	Error mask			021 8h	1
e001 1001	<i>reserved</i>			021 9h	
e001 1010	Digital Output			021 Ah	1
e001 1011	Digital Input			021 Bh	1
e001 1100	Baudrate			021 Ch	1
e001 1101	Response delay			021 Dh	1
e001 1110	Retry			021 Eh	1
e001 1111	Remote control			021 Fh	1
e010 0000	First storage # for cyclic storage			022 0h	1

1110		kV		Eh	
e100 1111		MV	V	024 Fh	10 ⁶
e101 0000	Current	pA	A	025 0h	10 ⁻¹²
e101 0001		10 pA	A	025 1h	10 ⁻¹¹
e101 0010		100 pA	A	025 2h	10 ⁻¹⁰
e101 0011		nA	A	025 3h	10 ⁻⁹
e101 0100		10 nA	A	025 4h	10 ⁻⁸
e101 0101		100 nA	A	025 5h	10 ⁻⁷
e101 0110		μA	A	025 6h	10 ⁻⁶
e101 0111		10 μA	A	025 7h	10 ⁻⁵
e101 1000		100 μA	A	025 8h	10 ⁻⁴
e101 1001		mA	A	025 9h	10 ⁻³
e101 1010		10 mA	A	025 Ah	10 ⁻²
e101 1011	100 mA	A	025 Bh	10 ⁻¹	
e101 1100	A	A	025 Ch	1	
e101 1101	10 A	A	025 Dh	10	
e101 1110	100 A	A	025 Eh	10 ²	
e101 1111	kA	A	025 Fh	10 ³	
e110 0000	Reset counter			026 0h	1

e010 0001	Last storage # for cyclic storage			022 1h	1
e010 0010	Size of storage block			022 2h	1
e010 0011	<i>reserved</i>			022 3h	
e010 0100	Storage interval	s	s	022 4h	1
e010 0101		min	s	022 5h	60
e010 0110		h	s	022 6h	3600
e010 0111		d	s	022 7h	8640 0
e010 1000	Storage interval	mont hs	s	022 8h	2629 743.8 3
e010 1001		years	s	022 9h	3155 6926
e010 1010	<i>reserved</i>			022 Ah	
e010 1011	Time point	s	s	022 Bh	1
e010 1100	Duration since last readout	s	s	022 Ch	1
e010 1101		min	s	022 Dh	60
e010 1110		h	s	022 Eh	3600
e010 1111		d	s	022 Fh	8640 0
e011 0000	Start of tariff			023 0h	(note 1v)

e110 0001	Cumulation counter			026 1h	1
e110 0010	Control signal			026 2h	1
e110 0011	Day of week			026 3h	1
e110 0100	Week number			026 4h	1
e110 0101	Time point of day change			026 5h	(note 1v)
e110 0110	State of param. activation			026 6h	1
e110 0111	Special supplier information			026 7h	1
e110 1000	Duration since last cumulation	h	s	026 8h	3600
e110 1001		d	s	026 9h	8640 0
e110 1010		mont hs	s	026 Ah	2629 743.8 3
e110 1011		years	s	026 Bh	3155 6926
e110 1100	Operating time battery	h	s	026 Ch	3600
e110 1101		d	s	026 Dh	8640 0
e110 1110		mont hs	s	026 Eh	2629 743.8 3
e110 1111		years	s	026 Fh	3155 6926
e111 0000	Date and time of battery			027 0h	(note 1v)

e011 0001	Duration of tariff	min	s	023 1h	60
e011 0010		h	s	023 2h	3600
e011 0011		d	s	023 3h	8640 0
e011 0100	Period of tariff	s	s	023 4h	1
e011 0101		min	s	023 5h	60
e011 0110		h	s	023 6h	3600
e011 0111		d	s	023 7h	8640 0
e011 1000	Period of tariff	mont hs	s	023 8h	2629 743.8 3
e011 1001		years	s	023 9h	3155 6926
e011 1010	dimensi onless			023 Ah	1
e011 1011	<i>reserved</i>			023 Bh	
e011 1100				023 Ch	
e011 1101				023 Dh	
e011 1110				023 Eh	
e011 1111				023 Fh	

	change				
e111 0001	<i>reserved</i>			027 1h	
e111 0010	Day light saving			027 2h	(note 2v)
e111 0011	Listenin g window manage ment			027 3h	(note 3v)
e111 0100	Remaini ng battery life time	d	s	027 4h	8640 0
e111 0101	# of meter stops			027 5h	1
e111 0110	<i>reserved</i>			027 6h	
e111 0111				027 7h	
e111 1000	<i>reserved</i>			027 8h	
e111 1001				027 9h	
e111 1010				027 Ah	
e111 1011				027 Bh	
e111 1100				027 Ch	
e111 1101				027 Dh	
e111 1110				027 Eh	
e111 1111				027 Fh	

Non metric units (VIFE = 3Dh)

VIF code	Type of quantity	Meas ure unit	Nor ma- lized mea	<i>uni</i> <i>t</i>	Multi plier
-------------	---------------------	---------------------	----------------------------	------------------------	----------------

VIF code	Type of quantity	Meas ure unit	Nor ma- lized mea	<i>uni</i> <i>t</i>	Multi plier
-------------	---------------------	---------------------	----------------------------	------------------------	----------------

			sure unit		
e000 0000	Energy	BTU	kBT U	030 0h	10 ⁻³
e000 0001		10 BTU	kBT U	030 1h	10 ⁻²
e000 0010		100 BTU	kBT U	030 2h	10 ⁻¹
e000 0011		kBTU	kBT U	030 3h	1
e000 0100		10 kBTU	kBT U	030 4h	10
e000 0101		100 kBTU	kBT U	030 5h	10 ²
e000 0110		MWh	kBT U	030 6h	10 ³
e000 0111		10 MWh	kBT U	030 7h	10 ⁴
e000 1000		Energy	J	kJ	030 8h
e000 1001	10 J		kJ	030 9h	10 ⁻²
e000 1010	100 J		kJ	030 Ah	10 ⁻¹
e000 1011	kJ		kJ	030 Bh	1
e000 1010	10 kJ		kJ	030 Ch	10
e000 1011	100 kJ		kJ	030 Dh	10 ²
e000 1110	MJ		kJ	030 Eh	10 ³
e000 1111	10 MJ		kJ	030 Fh	10 ⁴
e001 0000	Volume		mgal	gal	031 0h
e001 0001		10 mgal	gal	031 1h	10 ⁻²
e001 0010		100 mgal	gal	031 2h	10 ⁻¹

			sure unit		
e100 0000	Volume flow	mgal/ min	gal/ min	034 0h	10 ⁻³
e100 0001		10 mgal/ min	gal/ min	034 1h	10 ⁻²
e100 0010		100 mgal/ m	gal/ min	034 2h	10 ⁻¹
e100 0011		gal/mi n	gal/ min	034 3h	1
e100 0100		10 gal/mi n	gal/ min	034 4h	10
e100 0101		100 gal/mi n	gal/ min	034 5h	10 ²
e100 0110		kgal/ min	gal/ min	034 6h	10 ³
e100 0111		10 kgal/ min	gal/ min	034 7h	10 ⁴
e100 1000		Volume flow	ul/s	l/s	034 8h
e100 1001	10 ul/s		l/s	034 9h	10 ⁻⁵
e100 1010	100 ul/s		l/s	034 Ah	10 ⁻⁴
e100 1011	ml/s		l/s	034 Bh	10 ⁻³
e100 1100	10 ml/s		l/s	034 Ch	10 ⁻²
e100 1101	100 ml/s		l/s	034 Dh	10 ⁻¹
e100 1110	l/s		l/s	034 Eh	1
e100 1111	10 l/s		l/s	034 Fh	10
e101 0000	Mass flow		g/h	kg/h	035 0h
e101 0001		10 g/h	kg/h	035 1h	10 ⁻²
e101 0010		100 g/h	kg/h	035 2h	10 ⁻¹

e001 0011		gal	gal	031 3h	1
e001 0100		10 gal	gal	031 4h	10
e001 0101		100 gal	gal	031 5h	10 ²
e001 0110		kgal	gal	031 6h	10 ³
e001 0111		10 kgal	gal	031 7h	10 ⁴
e001 1000	Mass	g	kg	031 8h	10 ⁻³
e001 1001		10 g	kg	031 9h	10 ⁻²
e001 1010		100 g	kg	031 Ah	10 ⁻¹
e001 1011		kg	kg	031 Bh	1
e001 1100		10 kg	kg	031 Ch	10
e001 1101		100 kg	kg	031 Dh	10 ²
e001 1110		t	kg	031 Eh	10 ³
e001 1111		10 t	kg	031 Fh	10 ⁴
e010 0000	On time	s	s	032 0h	1
e010 0001		min	s	032 1h	60
e010 0010		h	s	032 2h	3600
e010 0011		d	s	032 3h	86400
e010 0100	Operating time	s	s	032 4h	1
e010 0101		min	s	032 5h	60
e010 0110		h	s	032 6h	3600
e010 0111		d	s	032 7h	86400
e010 1000	Power	μBTU/s	mBTU/s	032 8h	10 ⁻³

e101 0011		kg/h	kg/h	035 3h	1
e101 0100		10 kg/h	kg/h	035 4h	10
e101 0101		100 kg/h	kg/h	035 5h	10 ²
e101 0110		t/h	kg/h	035 6h	10 ³
e101 0111		10 t/h	kg/h	035 7h	10 ⁴
e101 1000	Flow temperature	m°F	°F	035 8h	10 ⁻³
e101 1001		10 m°F	°F	035 9h	10 ⁻²
e101 1010		100 m°F	°F	035 Ah	10 ⁻¹
e101 1011		°F	°F	035 Bh	1
e101 1100	Return temperature	m°F	°F	035 Ch	10 ⁻³
e101 1101		10 m°F	°F	035 Dh	10 ⁻²
e101 1110		100 m°F	°F	035 Eh	10 ⁻¹
e101 1111		°F	°F	035 Fh	1
e110 0000	Temperature difference	m°F	°F	036 0h	10 ⁻³
e110 0001		10 m°F	°F	036 1h	10 ⁻²
e110 0010		100 m°F	°F	036 2h	10 ⁻¹
e110 0011		°F	°F	036 3h	1
e110 0100	External temperature	m°F	°F	036 4h	10 ⁻³
e110 0101		10 m°F	°F	036 5h	10 ⁻²
e110 0110		100 m°F	°F	036 6h	10 ⁻¹
e110 0111		°F	°F	036 7h	1
e110 1000	Pressure	mbar	bar	036 8h	10 ⁻³

e010 1001		10 μBTU /s	mBT U/s	032 9h	10 ⁻²	e110 1001		10 mbar	bar	036 9h	10 ⁻²
e010 1010		100 μBTU /s	mBT U/s	032 Ah	10 ⁻¹	e110 1010		100 mbar	bar	036 Ah	10 ⁻¹
e010 1011		mBT U/s	mBT U/s	032 Bh	1	e110 1011		bar	bar	036 Bh	1
e010 1100		10 mBT U/s	mBT U/s	032 Ch	10	e110 1100	Time point			036 Ch	(note 1v)
e010 1101		100 mBT U/s	mBT U/s	032 Dh	10 ²	e110 1101				036 Dh	(note 1v)
e010 1110		BTU/s	mBT U/s	032 Eh	10 ³	e110 1110	Units for HCA			036 Eh	1
e010 1111		10 BTU/s	mBT U/s	032 Fh	10 ⁴	e110 1111	<i>reserved</i>			036 Fh	
e011 0000	Power	J/h	kJ/h	033 0h	10 ⁻³	e111 0000	Averagin g duration	s	s	037 0h	1
e011 0001		10 J/h	kJ/h	033 1h	10 ⁻²	e111 0001		min	s	037 1h	60
e011 0010		100 J/h	kJ/h	033 2h	10 ⁻¹	e111 0010		h	s	037 2h	3600
e011 0011		kJ/h	kJ/h	033 3h	1	e111 0011		d	s	037 3h	8640 0
e011 0100		10 kJ/h	kJ/h	033 4h	10	e111 1000	Cold/war m temperat ure limit	m°F	°F	037 4h	10 ⁻³
e011 0101		100 kJ/h	kJ/h	033 5h	10 ²	e111 0101		10 m°F	°F	037 5h	10 ⁻²
e011 0110		MJ/h	kJ/h	033 6h	10 ³	e111 0110		100 m°F	°F	037 6h	10 ⁻¹
e011 0111		10 MJ/h	kJ/h	033 7h	10 ⁴	e111 0111		°F	°F	037 7h	1
e011 1000		Volume flow	ml/h	l/h	033 8h	10 ⁻³	e111 1000	Fabricati on No.			037 8h
e011 1001	10 ml/h		l/h	033 9h	10 ⁻²	e111 1001	<i>reserved</i>			037 9h	
e011 1010	100 ml/h		l/h	033 Ah	10 ⁻¹	e111 1010				037 Ah	
e011 1011	l/h		l/h	033 Bh	1	e111 1011				037 Bh	
e011 1100	10 l/h		l/h	033 Ch	10	e111 1100				037 Ch	

e011 1101		100 l/h	l/h	033 Dh	10 ²	e111 1101				037 Dh	
e011 1110		m ³ /h	l/h	033 Eh	10 ³	e111 1110				037 Eh	
e011 1111		10 m ³ /h	l/h	033 Fh	10 ⁴	e111 1111				037 Fh	

- note 1v: value type depends from record *data field*. Records of *Type F, G* and *I*, when contain a date and time value, are converted in a format compatible with date/time function of scripting language (number of seconds from midnight of 01/01/1901). Records of *Type I* and *J*, when contain only time information, are converted in the number of seconds from midnight.

- note 2v: *Type K* records are not normalized.

- note 3v: *Type L* records are unsupported and generate error.

Following table shows how record value is normalized in presence of particular *Combinable (Orthogonal) VIF Extension*.

VIFE code	Description	Normalization
e011 1001	Start date/time of	
e100 xx1x	Date/time of begin/end of the first/last lower/upper limit exceeded	Value is parsed as <i>Type F, G, I</i> or <i>J</i> basing of <i>data field</i> , see note 1v
e110 1x1x	Date/time of begin/end of the first/last limit exceeded	
e101 xxnn	Duration of first/last lower/upper limit exceeded	
e110 0xnn	Duration of first/last limit exceeded	Value is expressed in seconds ($nn = 00$), minutes ($nn = 01$), hours ($nn = 10$) or days ($nn = 11$), it is converted in seconds
e111 0 nnn	Multiplicative correction factor: 10^{nnn-6}	Value is multiplied by 10^{nnn-6}
e111 10nn	Additive correction constant: 10^{nn-3} unit of VIF	Before any other rescaling, quantity 10^{nn-3} is summed to value
e111 1101	Multiplicative correction factor: 10^3	Value is multiplied by 10^3

Examples

Referring to communication sample reported in the M-Bus Usergroup specification and in EN13757-3 (Annex E):

78 56 34 12	Identification no = 12345678
24 40 01 07	Manufacturer Id = 4024h (PAD in EN 62056-21), version/generation = 1, medium = 7 (water)
55 00 00 00	Access no = 55h = 85, Status = 00h, Signature = 0000h
03 13 15 31 00	Data block (record) 1: Subunit = 0, Storage no = 0, No tariff, Instantaneous volume, 12565 l (24 bit integer)
DA 02 3B 13 01	Data block (record) 2: Subunit = 0, Storage no = 5, No tariff, Maximum

volume flow, 113 l/h (4 digit BCD)

8B 60 04 37 18 02 Data block (record) 3: Subunit = 1, Storage no = 0, Tariff = 2, Instantaneous energy, 218.37 kWh (6 digit BCD)

You can get following values:

Address field of the gate	Description	Resulting gate value
idNo	Field <i>Identification number</i> of the device	12345678
medium	Field <i>Medium</i> of the device	7 (water)
3	Normalized value of the 3rd record	218370 (value is converted to Wh)
0x02.normValue	Normalized value of the 2nd record	113 (value is already in the normalized unit l/h)
2.unit	<i>Unit</i> of the 2nd record	003Bh
3h.value	Value (not normalized) of the 3rd record	21837 (value is returned with VIF = 04 which means 10 Wh)
2.storageNo	<i>Storage no</i> of the 2nd record	5

Blocks of numeric gates

Blocks of numeric gates are supported and suggested. No restrictions limit length of blocks, but gates must be related to the same device.

When a read error occurs, all the gates belonging to the same block is set in error. So during initial debug of the project, can be useful to read gates independently and to group them in blocks only after communication works correctly.

15.3 Digital gates

Digital gates are not allowed in this protocol.

15.4 String gates

To read data from M-Bus devices, supervisor inquires them using *REQ_UD2* message; devices reply with a *SND_UD* telegram. *SND_UD* message is composed by an header (which contains some device info) and a set of *data blocks* (records), one for each variable. Each record contains the value of the variable and other related info (measure unit, scale, ...)

Header
Device variables
Record 1
Value
Other info
Record 2
Value
Other info
: :
Record n
Value
Other info

Using string gates, it is possible to retrieve data from header (*device variables*) and data (value and other info) from each record (*record variables*). To identify the wanted record you need to know the order in which records are arranged in the response message (ask response format to device manufacturer). For example Record 3 is the 3rd record of the response.

String gates address to retrieve *device variables*

To read device variables, *Address* field of string gates must comply to the following format:

var_id

where *var_id* can be one of the following

var_id	Description	Notes
manufacturer	Manufacturer	ASCII decoding of 16 bit value retrieved from the device. It is not available for devices that reply with Fixed Format Structure message

Numeric gates address to retrieve *record variables*

To read record variables, *Address* field of string gates must comply to the following format:

rec_no.var_id

where

- *rec_no* is the number of the record as it appears in response message (for example if you want the 3rd record of the response, type 3). To know the order in which records are arranged in the response message ask format of SND_UD message to device manufacturer. You can insert record number in decimal notation or hexadecimal notation (use prefix 0x or postfix the number with the character h); for example to select 23rd record, type 23 or 0x17 or 17h. For devices that reply with Fixed Format Structure message are allowed only records 1 and 2. If you require a unavailable record, gate will be set in communication error.
- *var_id* is the identification of which information has to be retrieved. Can be one of the following:

var_id	Description	Notes
value	Value	Value of the record as retrieved form

var_id	Description	Notes
		response message. Reading of non-text type records (<i>data field</i> = 1101b and $LVAR \leq BFh$) generates error. It is not available for devices that reply with Fixed Format Structure message.
normValue	Normalized value	"Normalized" value of the record. It is the same as <code>value</code> , except for record of <i>Type F, G, I, J</i> . When they contain a date and time value (or only a time value), are converted in a textual representation. It is not available for devices that reply with Fixed Format Structure message.
unit	Measure unit	Textual representation of measure unit.
normUnit	Normalized measure unit	Textual representation of measure unit in which values are converted. Refer to tables provided for numeric gates. For each case, columns <i>Measure unit</i> and <i>Normalized measure unit</i> indicate the textual representation.
extUnit1 ... extUnit10	Measure unit extensions	Textual representation of all <i>VIFE</i> s not used to determine the main measure unit. Textual representation is provided only for <i>VIFE</i> s that modify measure unit; see detailed table in the next paragraph. They are not available for devices that reply with Fixed Format Structure message.

Omitting `var_id` (and the separator character) and specifying only `rec_no`, the string gate gets the normalized value.

Textual representation of measure unit extensions

Rappresentazione testuali delle estensioni dell'unità di misura

Following table shows textual representation of measure unit extensions accessible using `extUnit1` ... `extUnit10`.

VIFE code	extUnit1 ... extUnit10
e010 0000	per s
e010 0001	per m
e010 0010	per h
e010 0011	per d
e010	per week

VIFE code	extUnit1 ... extUnit10
e010 0111	per revolution/measure ment
e010 1010	per l
e010 1011	per m ³
e010 1110	per kg
e010	per K

VIFE code	extUnit1 ... extUnit10
e011 0010	per kW
e011 0011	per (K*l)
e011 0100	per V
e011 0101	per A
e011	multiplied by s

0100		1111		0110	
e010 0101	per month	e011 0000	per kWh	e011 0111	multiplied by s/V
e010 0110	per year	e011 0001	per GJ	e011 1000	multiplied by s/A

Examples

Referring to communication sample reported in the M-Bus Usergroup specification and in EN13757-3 (Annex E):

78 56 34 12 Identification no = 12345678
 24 40 01 07 Manufacturer Id = 4024h (PAD in EN 62056-21), version/generation = 1, medium = 7 (water)
 55 00 00 00 Access no = 55h = 85, Status = 00h, Signature = 0000h
 03 13 15 31 00 Data block (record) 1: Subunit = 0, Storage no = 0, No tariff, Instantaneous volume, 12565 l (24 bit integer)
 DA 02 3B 13 01 Data block (record) 2: Subunit = 0, Storage no = 5, No tariff, Maximum volume flow, 113 l/h (4 digit BCD)
 8B 60 04 37 18 02 Data block (record) 3: Subunit = 1, Storage no = 0, Tariff = 2, Instantaneous energy, 218.37 kWh (6 digit BCD)

You can get following values:

Address field of the gate	Description	Resulting gate value
manufacturer	Field <i>Manufacturer</i> of the device	PAD
3h.normUnit	Normalized measure unit of the 3rd record	Wh
0x02.normUnit	Normalized measure unit of the 2nd record	l/h
3.unit	Measure unit of the 2nd record	10 Wh

An example with date/time records

17 06 70 19 Identification no = 19700617
 81 26 04 03 Manufacturer Id = 2681h (ITA in EN 62056-21), version/generation = 4, medium = 3 (gas)
 3B 00 00 00 Access no = 3Bh = 43, Status = 00h, Signature = 0000h
 02 FD C7 4E 03 B5 Data block (record) 1: Subunit = 0, Storage no = 0, No tariff, Begin of last upper limit exceeded, 03/05/1988 (Type G)
 06 6D 29 09 09 6C 1A 00 Data block (record) 2: Subunit = 0, Storage no = 0, No tariff, Time point, 12/10/2011 09:09:41 (Type I)

and the following string gates:

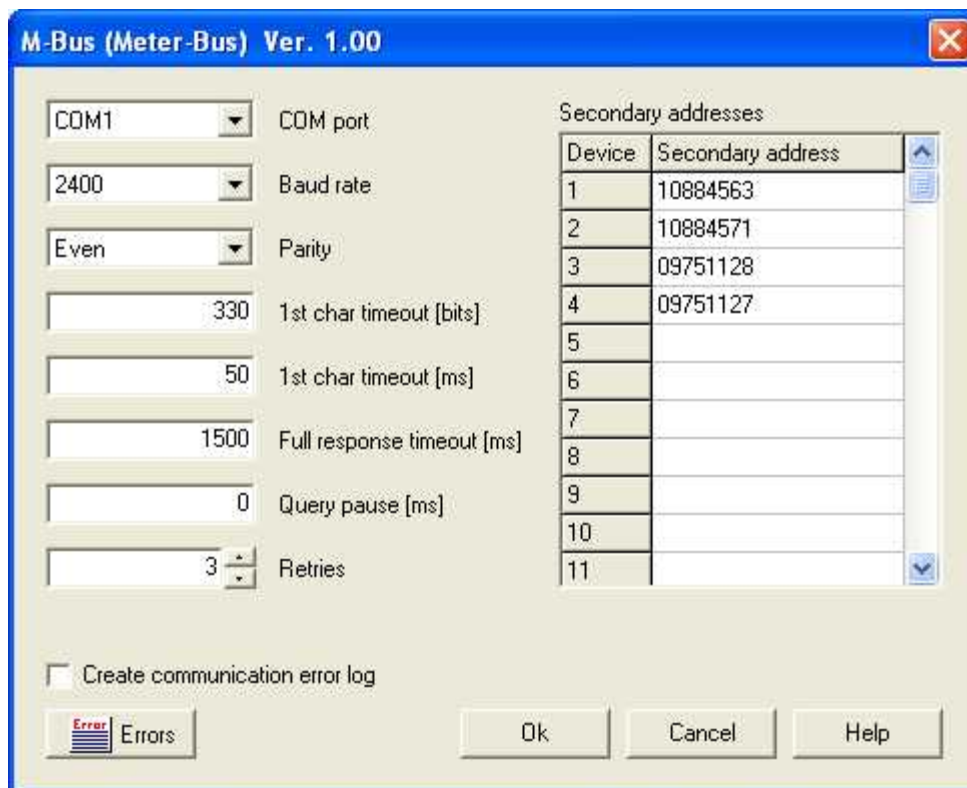
Address field of the gate	Description	Resulting gate value
manufacturer	Field <i>Manufacturer</i> of the device	ITA
1	Normalized value of the 1st record	03/05/1988
0x02.normalized value	Normalized value of the 2nd record	12/10/2011 09:09:41

Blocks of string gates

Blocks of string gates are supported and suggested. No restrictions limit length of blocks, but gates must be related to the same device.

When a read error occurs, all the gates belonging to the same block is set in error. So during initial debug of the project, can be useful to read gates independently and to group them in blocks only after communication works correctly.

15.5 Configuration



Protocol configuration dialog

COM port: COM port (or virtual COM port) connected to the M-Bus adaptor (level converter).

Baud rate: baud rate of communication link. All connected M-Bus devices must operate at same baud rate.

Parity: parity of the communication link. M-Bus specification requires *even* parity.

1st char timeout: maximum time allowed to receive first character of the response message.
Component in bits time (specification requires 330 bit) is summed to component in ms (50).
Using default values, at 2400 baud, first character timeout is about 190 ms.

Full response timeout: maximum time, after the first character, to receive all the characters of the response message. Please take care of baud rate and messages length; at 2400 baud, to receive 200 characters, at least 920 ms are required.

Query pause: delay (in ms) between an answer and the following request.

Retries: number of requests without answer before to declare error and abort communication procedure (specification requires 3 attempts)

Secondary addresses: to inquire devices using secondary address, this table should be compiled. For each device (it is the *Device* field of the gates configuration dialogs) specify the *Identification number* using which the device has to be selected. Wildcards are not allowed.

Create communication error log: if set, during runtime, protocol driver will create an error log file. To display error log press *Errors* button (during runtime as well). Error log is useful to debug projects helping you to find causes of communication errors.

16 MITSUBISHI Computer Link FX

16.1 Introduction

The Mitsubishi Computer Link protocol is a communication system for PLC Mitsubishi FX series via the RS232 or RS485 module.

16.2 Numeric gates

The gate address is specified adding the fields **Function** and **Address** of the following table.

Function	Description	Address	Gate read	Gate write	Block read	Notes
D	DATA REGISTER	XXXXX (0...65535)	Yes	Yes	Yes	1
D	DATA REGISTER	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
D_L_	DATA REGISTER LONG (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	2
D_L_	DATA REGISTER LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
D_F_	DATA REGISTER FLOAT (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	3
D_F_	DATA REGISTER FLOAT (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	3
R	EXTENSION REGISTER	XXXXX (0...65535)	Yes	Yes	Yes	1
R	EXTENSION REGISTER	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
TN	TIMER CURRENT VALUE	XXXXX (0...65535)	Yes	Yes	Yes	1
TN	TIMER CURRENT VALUE	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
TN_L_	TIMER CURRENT VALUE LONG (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	2
TN_L_	TIMER CURRENT VALUE LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
TS	TIMER CONTACT	XXXXX	Yes	Yes	Yes	1

		(0...65535)				
TS	TIMER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
CN	COUNTER CURRENT VALUE	XXXXX (0...65535)	Yes	Yes	Yes	1
CN	COUNTER CIURRENT VALUE	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
CN_L_	COUNTER CURRENT VALUE LONG (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	2
CN_L_	COUNTER CURRENT VALUE LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
CS	COUNTER CONTACT	XXXXX (0...65535)	Yes	Yes	Yes	1
CS	COUNTER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
X	INPU	XXXXX (octadecimal)	Yes	Yes	Yes	1
Y	OUTPUT	XXXXX (octadecimal)	Yes	Yes	Yes	1
M	INTERNAL RELAY	XXXXX (0...65535)	Yes	Yes	Yes	1
M	INTERNAL RELAY	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
S	STATE	XXXXX (0...65535)	Yes	Yes	Yes	
S	STATE	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	

Notes:

1 - 16 bit value : "Variable type" field of the numeric gate must be specified as :
 "U WORD" to consider the value as a 16-bit unsigned (0 ... 65535).
 "S WORD" to consider the value as a 16-bit signed (-32768 ... 32767).

2 - 32 bit value Long : "Variable type" field of the numeric gate must be specified as :
 "U_INT32" to consider the value as a 32-bit unsigned (0 ... 4294967295).
 "S_INT32" to consider the value as a 32-bit signed (-2147483648 ... 2147483647).

3 - 32 bit value Float : "Variable type" field of the numeric gate must be specified as :
 "FLOAT" to consider the value as a 32-bit IEEE 754 floating point.

Example: here are some addresses of numeric gates:

D15 DATA REGISTER 15 (16 bit)

DFh DATA REGISTER 15 (16 bit)

D_L_8 DATA REGISTER 8 LONG (32 bit)

D_L_8h DATA REGISTER 8 LONG (32 bit)

D_F_12 DATA REGISTER 12 FLOAT (32 bit)

D_F_Ch DATA REGISTER 12 FLOAT (32 bit)

R15 EXTENSION REGISTER 15 (16 bit)

RFh EXTENSION REGISTER 15 (16 bit)

Gates having the same **Function** and consecutive **Address** in relation to the type of variable, can be grouped into a block so as to be read them all with a single request.

Blocks can be created automatically from the gate definition tool, by selecting the "File | Gates sampling Optimization" menu item.

Numeric gates that CAN be read as block	Numeric gates thatn CAN NOT be read as block
D1	D1
D2	D2
D3	D7
D4	D8
D5	D10

Numeric gates that CAN be read as block	Numeric gates thatn CAN NOT be read as block
D_L_0	D_L_0
D_L_2	D_L_2
D_L_4	D_L_6
D_L_6	D_L_8
D_L_8	D_L_10

Numeric gates that CAN be read as block	Numeric gates thatn CAN NOT be read as block
X0	X0
X20	X16
X40	X40
X60	X50
X80	X60

16.3 Digital gates

The gate address is specified adding the fields **Function** and **Address** of the following table.

Function	Description	Address	Gate read	Gate write	Block read
TS	TIMER CONTACT	XXXXX (0...65535)	Yes	Yes	Yes
TS	TIMER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
CS	COUNTER CONTACT	XXXXX (0...65535)	Yes	Yes	Yes
CS	COUNTER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
X	INPUT	XXXXX (octadecimal)	Yes	Yes	Yes
Y	OUTPUT	XXXXX (octadecimal)	Yes	Yes	Yes
M	INTERNAL RELAY	XXXXX (0...65535)	Yes	Yes	Yes
M	INTERNAL RELAY	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
S	STATE	XXXXX (0...65535)	Yes	Yes	Yes
S	STATE	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes

Example: here are some addresses of digital gates:

TS15 TIMER CONTACT 15
TSFh TIMER CONTACT 15

CS8 COUNTER CONTACT 8
CS8h COUNTER CONTACT 8

M12 INTERNAL RELAY 12
MCh INTERNAL RELAY 12

S15 STATE 15
SFh STATE 15

Gates having the same **Function** and consecutive **Address** can be grouped into a block so as to be read them all with a single request.

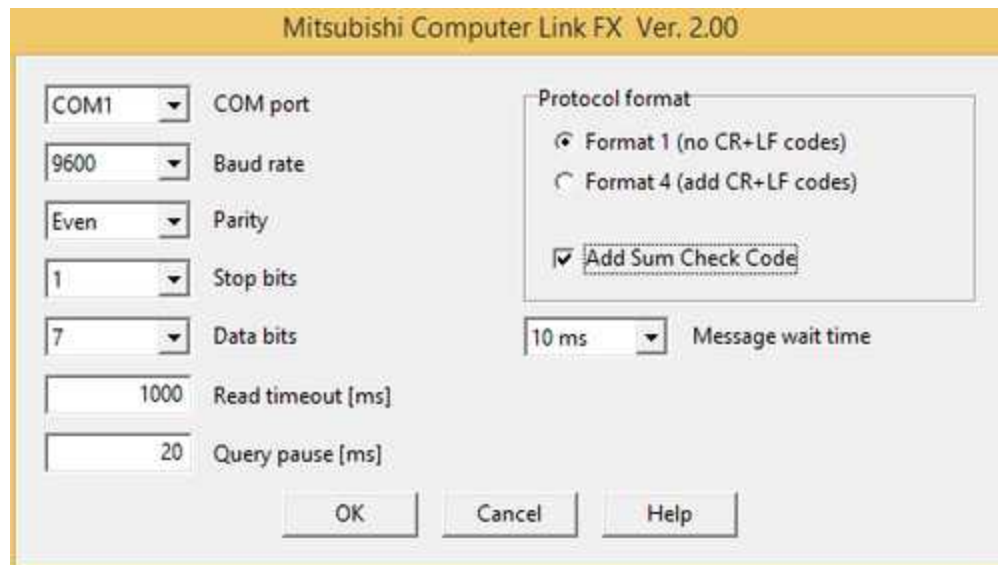
Blocks can be created automatically from the gate definition tool, by selecting the "File | Gates sampling Optimization" menu item.

Digital gates that CAN be read as block	Digital gates that CAN NOT be read as block
TS1	TS1
TS2	TS2
TS3	CS4
TS4	CS6
TS5	M10

16.4 String gates

String gates are not allowed in this protocol.

16.5 Configuration



Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.

- **Data bits:** number of bits.
- **Read timeout [ms]:** timeout (milliseconds) for a complete answer .
- **Query pause [ms]:** timeout (milliseconds) between two request messages .
- **Protocol format:** can be:
 - Format 1 (no CR + LF codes)**
 - Format 4 (con CR + LF codes)**
 - Add Sum Check Code :** if activated the checksums will be added to the frame.
- **Message wait time:** this is a delay time required to switch between send and receive states. The message wait time determines the minimum delay before the programmable controller sends data after receiving a message from the computer. The message wait time may be selected between 0 and 150 ms. When using the 485PC-IF in a 1:n system always set the message wait time to 70 ms or longer.

17 MITSUBISHI FR-CU03

17.1 Introduction

This protocol is used for communication with Mitsubishi inverters equipped with computer link unit FR-CU03.

17.2 Numeric gates

The gate address must be specified with the following format: **LXDDD**

Data length(2,4,6 characters)	Link Parameter Expansion (0 for parameters ranging from 0 to 99, 1 for parameters ranging from 100 to 915)	Function	Read Gate	Write gate	Read Block
L	X	DDD	Yes	Yes	No

Example: we want to read the parameter Max Frequency limit which corresponds to function 1: the gate address will be **40001**:

“4” (Data length = 4 characters) + “0” (Parameter from 0 to 99) + “001” (function code for Read Max frequency limit).

Function provided

Function	Description	Gate type	Read gate	Write gate	Read Block
0...127	Read parameters	Numeric	Yes	No	No

When using this protocol, the read function must be specified in the gate address; in case of gate write, the software will automatically substitute the read function with the write function.

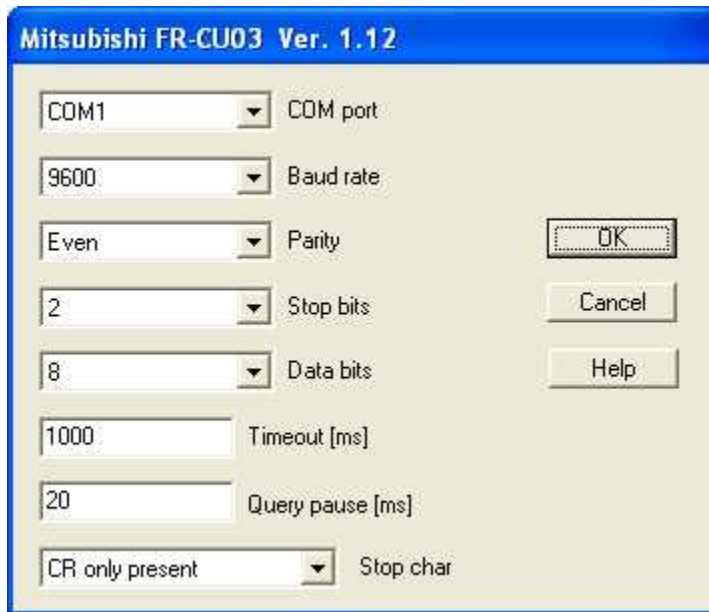
17.3 Digital gates

Digital gates are not allowed in this protocol.

17.4 String gates

String gates are not allowed in this protocol.

17.5 Configuration



Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages.
- **Stop char :** Stop characters at the end of the message. Three cases:
 - No stop character
 - Carriage Return
 - Carriage Return + Line Feed

18 MITSUBISHI MC PROTOCOL (1E frame)

18.1 Introduction

The MELSEC protocol (also known as MC protocol) is a communication system for Mitsubishi FX, Q and L PLC via the Ethernet interface.

Supported CPU:

FX3 (Ethernet built-in)

LCPU (Ethernet built-in)

QCPU (Ethernet built-in)

Additional supported Ethernet modules:

FX3U-ENET-ADP

FX3U-ENET

LJ71EN71

QJ71EN71

18.2 Numeric gates

The gate address is specified adding the fields **Function** and **Address** of the following table.

For **FX3** (with built-in Ethernet) or additional Ethernet modules **FX3U-ENET-ADP** and **FX3U-ENET**

Function	Description	Address	Gate read	Gate write	Block read	Notes
D	DATA REGISTER	XXXXX (0...65535)	Yes	Yes	Yes	1
D	DATA REGISTER	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
D_L_	DATA REGISTER LONG (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	2
D_L_	DATA REGISTER LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
D_F_	DATA REGISTER FLOAT (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	3
D_F_	DATA REGISTER FLOAT (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	3
R	EXTENSION REGISTER	XXXXX (0...65535)	Yes	Yes	Yes	1
R	EXTENSION REGISTER	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
R_L_	EXTENSION REGISTER LONG (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	2
R_L_	EXTENSION REGISTER LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
R_F_	EXTENSION REGISTER FLOAT (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	3
R_F_	EXTENSION REGISTER FLOAT (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	3
TN	TIMER CURRENT VALUE	XXXXX (0...65535)	Yes	Yes	Yes	1
TN	TIMER CURRENT VALUE	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
TN_L_	TIMER CURRENT VALUE LONG (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	2

TN_L_	TIMER CURRENT VALUE LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
TS	TIMER CONTACT	XXXXX (0..65535)	Yes	Yes	Yes	1
TS	TIMER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
CN	COUNTER CURRENT VALUE	XXXXX (0..65535)	Yes	Yes	Yes	1
CN	COUNTER CURRENT VALUE	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
CN_L_	COUNTER CURRENT VALUE LONG (32 bit)	XXXXX (0..65535)	Yes	Yes	Yes	2
CN_L_	COUNTER CURRENT VALUE LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
CS	COUNTER CONTACT	XXXXX (0..65535)	Yes	Yes	Yes	1
CS	COUNTER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
X	INPUT	XXXXX (octadecimal)	Yes	Yes	Yes	1,4
Y	OUTPUT	XXXXX (octadecimal)	Yes	Yes	Yes	1,4
M	INTERNAL RELAY	XXXXX (0..65535)	Yes	Yes	Yes	1
M	INTERNAL RELAY	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
S	STATE	XXXXX (0..65535)	Yes	Yes	Yes	
S	STATE	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	

For **LCPU** (with built-in Ethernet), **QCPU** (with built-in Ethernet) or additional Ethernet modules **LJ71EN71** and **QJ71EN71**

Function	Description	Address	Gate read	Gate write	Block read	Notes
D	DATA REGISTER	XXXXX (0..65535)	Yes	Yes	Yes	1
D	DATA REGISTER	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
D_L_	DATA REGISTER LONG (32 bit)	XXXXX (0..65535)	Yes	Yes	Yes	2
D_L_	DATA REGISTER LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
D_F_	DATA REGISTER FLOAT (32 bit)	XXXXX (0..65535)	Yes	Yes	Yes	3
D_F_	DATA REGISTER FLOAT (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	3
D_D_	DATA REGISTER DOUBLE FLOAT (64 bit)	XXXXX (0..65535)	Yes	Yes	Yes	5
D_D_	DATA REGISTER DOUBLE FLOAT (64 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	5
W	LINK REGISTER	XXXXX (0..65535)	Yes	Yes	Yes	1
W	LINK REGISTER	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
W_L_	LINK REGISTER LONG (32 bit)	XXXXX (0..65535)	Yes	Yes	Yes	2
W_L_	LINK REGISTER LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
W_F_	LINK REGISTER FLOAT (32 bit)	XXXXX (0..65535)	Yes	Yes	Yes	3

W_F_	LINK REGISTER FLOAT (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	3
W_D_	LINK REGISTER DOUBLE FLOAT (64 bit)	XXXXX (0...65535)	Yes	Yes	Yes	5
W_D_	LINK REGISTER DOUBLE FLOAT (64 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	5
R	FILE REGISTER	XXXXX (0...65535)	Yes	Yes	Yes	1
R	FILE REGISTER	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
R_L_	FILE REGISTER LONG (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	2
R_L_	FILE REGISTER LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
R_F_	FILE REGISTER FLOAT (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	3
R_F_	FILE REGISTER FLOAT (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	3
R_D_	FILE REGISTER DOUBLE FLOAT (64 bit)	XXXXX (0...65535)	Yes	Yes	Yes	5
R_D_	FILE REGISTER DOUBLE FLOAT (64 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	5
TN	TIMER CURRENT VALUE	XXXXX (0...65535)	Yes	Yes	Yes	1
TN	TIMER CURRENT VALUE	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
TN_L_	TIMER CURRENT VALUE LONG (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	2
TN_L_	TIMER CURRENT VALUE LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
TS	TIMER CONTACT	XXXXX (0...65535)	Yes	Yes	Yes	1
TS	TIMER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
TC	TIMER COIL	XXXXX (0...65535)	Yes	Yes	Yes	1
TC	TIMER COIL	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
CN	COUNTER CURRENT VALUE	XXXXX (0...65535)	Yes	Yes	Yes	1
CN	COUNTER CURRENT VALUE	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
CN_L_	COUNTER CURRENT VALUE LONG (32 bit)	XXXXX (0...65535)	Yes	Yes	Yes	2
CN_L_	COUNTER CURRENT VALUE LONG (32 bit)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	2
CS	COUNTER CONTACT	XXXXX (0...65535)	Yes	Yes	Yes	1
CS	COUNTER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
CC	COUNTER COIL	XXXXX (0...65535)	Yes	Yes	Yes	1
CC	COUNTER COIL	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
X	INPUT	XXXXX (octadecimale)	Yes	Yes	Yes	1,4
Y	OUTPUT	XXXXX (octadecimale)	Yes	Yes	Yes	1,4
M	INTERNAL RELAY	XXXXX (0...65535)	Yes	Yes	Yes	1

M	INTERNAL RELAY	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
B	LINK RELAY	XXXXX (0...65535)	Yes	Yes	Yes	1
B	LINK RELAY	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1
F	ANNUNCIATOR	XXXXX (0...65535)	Yes	Yes	Yes	1
F	ANNUNCIATOR	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes	1

Notes:

1 - 16 bit value : "Variable type" field of the numeric gate must be specified as :

"U WORD" to consider the value as a 16-bit unsigned (0 ... 65535).

"S WORD" to consider the value as a 16-bit signed (-32768 ... 32767).

2 - 32 bit value Long : "Variable type" field of the numeric gate must be specified as :

"U_INT32" to consider the value as a 32-bit unsigned (0 ... 4294967295).

"S_INT32" to consider the value as a 32-bit signed (-2147483648 ... 2147483647).

3 - 32 bit value Float : "Variable type" field of the numeric gate must be specified as :

"FLOAT" to consider the value as a 32-bit IEEE 754 floating point.

4 - X and Y are expressed in octadecimal so the address that can be designated for numeric gates must comply with the following rule : X0,X20,X40Y0,Y20,Y40...

5 - 64 bit value Double Float : "Variable type" field of the numeric gate must be specified as :

"DOUBLE" to consider the value as a 64-bit double floating point.

Example: here are some addresses of numeric gates:

D15 DATA REGISTER 15 (16 bit)
DFh DATA REGISTER 15 (16 bit)

D_L_8 DATA REGISTER 8 LONG (32 bit)
D_L_8h DATA REGISTER 8 LONG (32 bit)

D_F_12 DATA REGISTER 12 FLOAT (32 bit)
D_F_Ch DATA REGISTER 12 FLOAT (32 bit)

R15 EXTENSION REGISTER 15 (16 bit)
RFh EXTENSION REGISTER 15 (16 bit)

Gates having the same **Function** and consecutive **Address** in relation to the type of variable, can be grouped into a block so as to be read them all with a single request.

Blocks can be created automatically from the gate definition tool, by selecting the "File | Gates sampling Optimization" menu item.

Numeric gates that CAN be read as block	Numeric gates thatn CAN NOT be read as block
D1	D1
D2	D2
D3	D7
D4	D8
D5	D10

Numeric gates that CAN be read as block	Numeric gates thatn CAN NOT be read as block
---	--

D_L_0	D_L_0
D_L_2	D_L_2
D_L_4	D_L_6
D_L_6	D_L_8
D_L_8	D_L_10

Numeric gates that CAN be read as block	Numeric gates thatn CAN NOT be read as block
X0	X0
X20	X16
X40	X40
X60	X50
X80	X60

18.3 Digital gates

The gate address is specified adding the fields **Function** and **Address** of the following table.

For **FX3** (with built-in Ethernet) or additional Ethernet modules **FX3U-ENET-ADP** and **FX3U-ENET**

Function	Description	Address	Gate read	Gate write	Block read
TS	TIMER CONTACT	XXXXX (0...65535)	Yes	Yes	Yes
TS	TIMER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
CS	COUNTER CONTACT	XXXXX (0...65535)	Yes	Yes	Yes
CS	COUNTER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
X	INPUT	XXXXX (octadecimal)	Yes	Yes	Yes
Y	OUTPUT	XXXXX (octadecimal)	Yes	Yes	Yes
M	INTERNAL RELAY	XXXXX (0...65535)	Yes	Yes	Yes
M	INTERNAL RELAY	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
S	STATE	XXXXX (0...65535)	Yes	Yes	Yes
S	STATE	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes

For **LCP** (with built-in Ethernet), **QCP** (with built-int Ethernet) or additional Ethernet modules **LJ71EN71** and **QJ71EN71**

Function	Description	Address	Gate read	Gate write	Block read
TS	TIMER CONTACT	XXXXX (0...65535)	Yes	Yes	Yes
TS	TIMER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
TC	TIMER COIL	XXXXX (0...65535)	Yes	Yes	Yes
TC	TIMER COIL	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
CS	COUNTER CONTACT	XXXXX (0...65535)	Yes	Yes	Yes
CS	COUNTER CONTACT	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
CC	COUNTER COIL	XXXXX (0...65535)	Yes	Yes	Yes

CC	COUNTER COIL	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
X	INPUT	XXXXX (octadecimal)	Yes	Yes	Yes
Y	OUTPUT	XXXXX (octadecimal)	Yes	Yes	Yes
M	INTERNAL RELAY	XXXXX (0...65535)	Yes	Yes	Yes
M	INTERNAL RELAY	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
B	LINK RELAY	XXXXX (0...65535)	Yes	Yes	Yes
B	LINK RELAY	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
F	ANNUNCIATOR	XXXXX (0...65535)	Yes	Yes	Yes
F	ANNUNCIATOR	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes

Example: here are some addresses of digital gates:

TS15 TIMER CONTACT 15
TSFh TIMER CONTACT 15

CS8 COUNTER CONTACT 8
CS8h COUNTER CONTACT 8

M12 INTERNAL RELAY 12
MCh INTERNAL RELAY 12

S15 STATE 15
SFh STATE 15

Gates having the same **Function** and consecutive **Address** can be grouped into a block so as to be read them all with a single request.

Blocks can be created automatically from the gate definition tool, by selecting the "File | Gates sampling Optimization" menu item.

Digital gates that CAN be read as block	Digital gates that CAN be read as block
TS1	TS1
TS2	TS2
TS3	CS4
TS4	CS6
TS5	M10

18.4 String gates

The gate address is specified adding the fields **Function** and **Address** of the following table.

For **LCPU** (with built-in Ethernet), **QCPU** (with built-int Ethernet) or additional Ethernet modules **LJ71EN71** and **QJ71EN71**

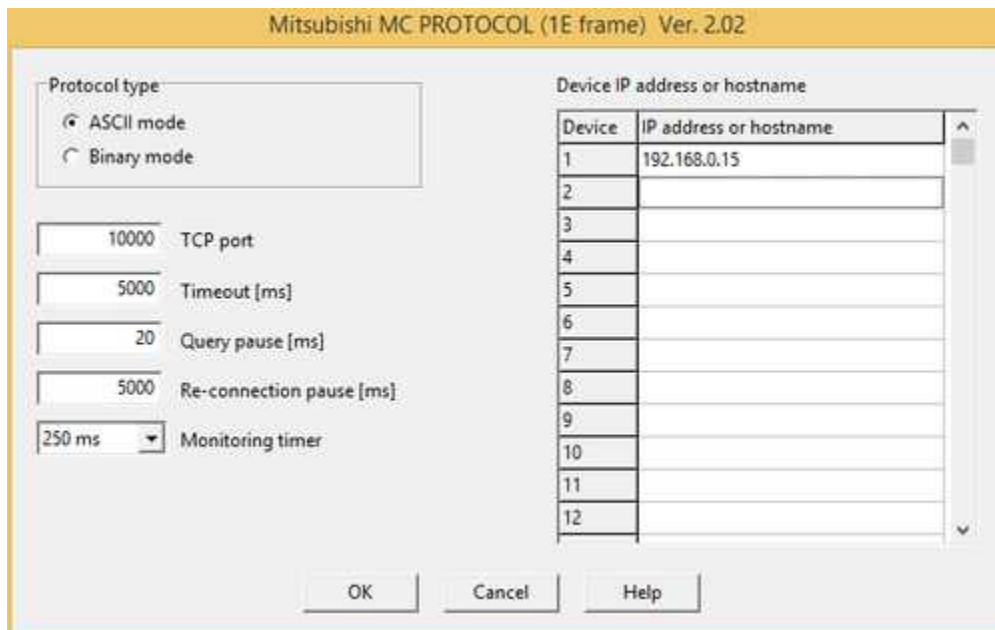
Function	Description	Address	Gate read	Gate write	Block read
D	DATA REGISTER	XXXXX (0...65535)	Yes	Yes	Yes
D	DATA REGISTER	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
W	LINK REGISTER	XXXXX (0...65535)	Yes	Yes	Yes
W	LINK REGISTER	XXXXh	Yes	Yes	Yes

		(0...FFFF Hexadecimal)			
R	FILE REGISTER	XXXXX (0...65535)	Yes	Yes	Yes
R	FILE REGISTER	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes

Example: here are some addresses of string gates:

D15 DATA REGISTER 15 (16 bit)
DFh DATA REGISTER 15 (16 bit)

18.5 Configuration



Protocol configuration window.

- **Protocol type:**
 - ASCII mode:** communication in ASCII format (1 byte = 2 chars sent)
 - Binary mode:** communication in Binary format (1 byte = 1 char sent)
- **TCP port:** protocol communication TCP port
- **Timeout [ms]:** timeout (milliseconds) for a complete answer.
- **Query pause [ms]:** timeout between two request messages.
- **Re-connection pause [ms]:** pause between close socket and open socket in case of communication errors.
- **Monitoring timer:** period of time that the Ethernet module should wait after outputting a read/write request to the PLC until the result is returned.
- **Device IP address or hostname:** PLC IP address or PLC hostname. The following syntax are supported:

192.168.1.100	Access at 192.168.1.100 to default TCP port
mydomain.com	Access at mydomain.com to default TCP port
192.168.1.100:88	Access at 192.168.1.100 to TCP port 88 (custom)
mydomain.com:88	Access at mydomain.com to TCP port 88 (custom)

Note: on the PLC side the **Ethernet Port open setting** must be configured as :

Protocol = TCP

Open System = MC Protocol

Host Station Port no. must be the same of **TCP port** parameter in the above window.

19 MODBUS TCP - MODBUS RTU - MODBUS ASCII

19.1 Introduction

Modbus TCP, RTU and ASCII protocols are used by a large number of devices.

With these two protocols it is necessary to indicate the read function in the address of all gates; in case of gate write, the supervisor software will automatically substitute the read function with the write function.

Gate address can be one of the sequent format:

30001 (Function 3 and address 0001) : obsolete format.

3:1 (Function 3 and address 1) : new format.

3h:1h (Function 3 and address 1) : new format.

In hexadecimal format the "h" char must be lower case.

19.2 Numeric gates

The gate address is specified adding the fields Function and Address of the following table.

Decription	Function	Address	Gate read	Gate write	Block read
3 (obsolete)	HOLDING REGISTER 16 bit	XXXX (0...9999 decimal)	Yes	Yes	Yes
3:	HOLDING REGISTER 16 bit	XXXXX (0...65535 decimal)	Yes	Yes	Yes
3h:	HOLDING REGISTER 16 bit	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
3:16:	HOLDING REGISTER 16 bit	XXXXX (0...65535 decimal)	Yes	Yes	Yes
3h:10h:	HOLDING REGISTER 16 bit	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
4 (obsolete)	INPUT REGISTER 16 bit	XXXX (0...9999 decimal)	Yes	No	Yes
4:	INPUT REGISTER 16 bit	XXXXX (0...65535 decimal)	Yes	No	Yes
4h:	INPUT REGISTER 16 bit	XXXXh (0...FFFF Hexadecimal)	Yes	No	Yes
7	EXCEPTION STATUS		Yes	No	No
33 (obsolete)	HOLDING REGISTER 32 bit (FLOAT)	XXXX (0...9999 decimal)	Yes	Yes	Yes

33:	HOLDING REGISTER 32 bit (FLOAT)	XXXXX (0...65535 decimal)	Yes	Yes	Yes
21h:	HOLDING REGISTER 32 bit (FLOAT)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
36 (obsolete)	HOLDING REGISTER 32 bit (FLOAT) reverse mode	XXXX (0...9999 decimal)	Yes	Yes	Yes
36:	HOLDING REGISTER 32 bit (FLOAT) reverse mode	XXXXX (0...65535 decimal)	Yes	Yes	Yes
24h:	HOLDING REGISTER 32 bit (FLOAT) reverse mode	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
44 (obsolete)	INPUT REGISTER 32 bit (FLOAT)	XXXX (0...9999 decimal)	Yes	No	Yes
44:	INPUT REGISTER 32 bit (FLOAT)	XXXXX (0...65535 decimal)	Yes	No	Yes
2Ch:	INPUT REGISTER 32 bit (FLOAT)	XXXXh (0...FFFF Hexadecimal)	Yes	No	Yes
46 (obsolete)	INPUT REGISTER 32 bit (FLOAT) reverse mode	XXXX (0...9999 decimal)	Yes	No	Yes
46:	INPUT REGISTER 32 bit (FLOAT) reverse mode	XXXXX (0...65535 decimal)	Yes	No	Yes
2Eh:	INPUT REGISTER 32 bit (FLOAT) reverse mode	XXXXh (0...FFFF Hexadecimal)	Yes	No	Yes
35 (obsolete)	HOLDING REGISTER 32 bit (LONG)	XXXX (0...9999 decimal)	Yes	Yes	Yes
35:	HOLDING REGISTER 32 bit (LONG)	XXXXX (0...65535 decimal)	Yes	Yes	Yes
23h:	HOLDING REGISTER 32 bit (LONG)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
37 (obsolete)	HOLDING REGISTER 32 bit (LONG) reverse mode	XXXX (0...9999 decimal)	Yes	Yes	Yes
37:	HOLDING REGISTER 32 bit (LONG) reverse mode	XXXXX (0...65535 decimal)	Yes	Yes	Yes
25h:	HOLDING REGISTER 32 bit (LONG) reverse mode	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
45 (obsolete)	INPUT REGISTER 32 bit (LONG)	XXXX (0...9999 decimal)	Yes	No	Yes
45:	INPUT REGISTER 32 bit (LONG)	XXXXX (0...65535 decimal)	Yes	No	Yes
2Dh:	INPUT REGISTER 32 bit (LONG)	XXXXh (0...FFFF Hexadecimal)	Yes	No	Yes
47 (obsolete)	INPUT REGISTER 32 bit (LONG) reverse mode	XXXX (0...9999 decimal)	Yes	No	Yes

47:	INPUT REGISTER 32 bit (LONG) reverse mode	XXXXX (0...65535 decimal)	Yes	No	Yes
2Fh:	INPUT REGISTER 32 bit (LONG) reverse mode	XXXXh (0...FFFF Hexadecimal)	Yes	No	Yes
38 (obsolete)	HOLDING REGISTER 64 bit (DOUBLE FLOAT)	XXXX (0...9999 decimal)	Yes	Yes	Yes
38:	HOLDING REGISTER 64 bit (DOUBLE FLOAT)	XXXXX (0...65535 decimal)	Yes	Yes	Yes
26h:	HOLDING REGISTER 64 bit (DOUBLE FLOAT)	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
48 (obsolete)	INPUT REGISTER 64 bit (DOUBLE FLOAT)	XXXX (0...9999 decimal)	Yes	No	Yes
48:	INPUT REGISTER 64 bit (DOUBLE FLOAT)	XXXXX (0...65535 decimal)	Yes	No	Yes
30h:	INPUT REGISTER 64 bit (DOUBLE FLOAT)	XXXXh (0...FFFF Hexadecimal)	Yes	No	Yes
39 (obsolete)	HOLDING REGISTER 64 bit (DOUBLE FLOAT) reverse mode	XXXX (0...9999 decimal)	Yes	Yes	Yes
39:	HOLDING REGISTER 64 bit (DOUBLE FLOAT) reverse mode	XXXXX (0...65535 decimal)	Yes	Yes	Yes
27h:	HOLDING REGISTER 64 bit (DOUBLE FLOAT) reverse mode	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
49 (obsolete)	INPUT REGISTER 64 bit (DOUBLE FLOAT) reverse mode	XXXX (0...9999 decimal)	Yes	No	Yes
49:	INPUT REGISTER 64 bit (DOUBLE FLOAT) reverse mode	XXXXX (0...65535 decimal)	Yes	No	Yes
31h:	INPUT REGISTER 64 bit (DOUBLE FLOAT) reverse mode	XXXXh (0...FFFF Hexadecimal)	Yes	No	Yes

Example: the following are some examples of numeric gates:

30011 : Holding register 0011 (16 bit).Obsolete

3:11 : Holding register 11 (16 bit).

3h:Bh : Holding register 11 (16 bit).

3h:10h:Bh : Holding register 11 (16 bit) with write function 10h instead of 6h.

40004 : Input register 0004 (16 bit).Obsolete

4:4 : Input register 4 (16 bit).

4h:4h : Input register 4 (16 bit).

7: Exception Status.

330011 : Holding register 0011 (32 bit float).Obsolete

33:11 : Holding register 11 (32 bit float).

21h:Bh : Holding register 11 (32 bit float).

440004 : Input register 0004 (32 bit float). Obsolete

44:4 : Input register 4 (32 bit float).

2Ch:4h : Input register 4 (32 bit float).

Note:

A block of numeric gates must be made only by gates with the same function and sequential addresses.

Numeric gates that CAN be read as block	Numeric gates thatn CAN NOT be read as block
3:3	4:3
3:4	4:5
3:5	3:12
3:6	4:13
3:7	4:14

19.3 Digital gates

The gate address is specified adding the fields Function and Address of the following table.

Function	Description	Address	Gate read	Gate write	Read Block
1 (obsolete)	COIL REGISTER	XXXX (0...9999 decimal)	Yes	Yes	Yes
1:	COIL REGISTER	XXXXX (0...65535 decimal)	Yes	Yes	Yes
1h:	COIL REGISTER	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	Yes
2 (obsolete)	INPUT STATUS	XXXX (0...9999 decimal)	Yes	No	Yes
2:	INPUT STATUS	XXXXX (0...65535 decimal)	Yes	No	Yes
2h:	INPUT STATUS	XXXXh (0...FFFF Hexadecimal)	Yes	No	Yes

Example: the following are some examples of digital gates:

10003 : Coil register 0003.

1:3 : Coil register 3.

1h:3h : Coil register 3.

20120 : Input Status 0120.

2:120 : Input Status 120.

2h:78h : Input Status 120.

Note:

A block of digital gates must be made only by gates with the same function and sequential addresses.

Digital gates that CAN be read as block	Digital gates thatn CAN NOT be read as block
1:123	2:3
1:124	1:5
1:125	1:12
1:126	1:13
1:127	2:14

19.4 String gates

String gates function are supported only in Modbus RTU and Modbus RTU TCP protocol

For Holding register:

Function **33 (21h)** extract a string from a block of holding register in the following mode:

LowByte (HoldingRegister 1) +
LowByte (HoldingRegister 2) +
LowByte (HoldingRegister ...) +
LowByte (HoldingRegister N)

Function	Address	Gate read	Gate write	Block read
33 (obsolete)	XXXX (0...9999 decimal)	Yes	Yes	No
33:	XXXXX (0...65535 decimal)	Yes	Yes	No
21h:	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	No

Example:

These are 3 way to read data string from Holding (read/write) register Modbus address 0001
330001
33:0001
21h:0001h

Function **36 (24h)** extract a string from a block of holding register in the following mode:

HighByte (HoldingRegister 1) +
HighByte (HoldingRegister 2) +
HighByte (HoldingRegister ...) +
HighByte (HoldingRegister N)

Function	Address	Gate read	Gate write	Block read
36 (obsolete)	XXXX (0...9999 decimal)	Yes	Yes	No
36:	XXXXX (0...65535 decimal)	Yes	Yes	No
24h:	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	No

Example:

These are 3 way to read data string from Holding (read/write) register Modbus address 0001
360001
36:0001
24h:0001h

Function **35 (23h)** extract a string from a block of holding register in the following mode:

HighByte (HoldingRegister 1) + **LowByte** (HoldingRegister 1) +
HighByte (HoldingRegister 2) + **LowByte** (HoldingRegister 2) +
HighByte (HoldingRegister ...) + **LowByte** (HoldingRegister ...) +
HighByte (HoldingRegister N) + **LowByte** (HoldingRegister N)

Function	Address	Gate read	Gate write	Block read
35 (obsolete)	XXXX (0...9999 decimal)	Yes	Yes	No
35:	XXXXX (0...65535 decimal)	Yes	Yes	No
23h:	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	No

Example:

These are 3 way to read data string from Holding (read/write) register Modbus address 0001
350001
35:0001
23h:0001h

Function **37 (25h)** extract a string from a block of holding register in the following mode:

LowByte (HoldingRegister 1) + **HighByte** (HoldingRegister 1) +
LowByte (HoldingRegister 2) + **HighByte** (HoldingRegister 2) +
LowByte (HoldingRegister ...) + **HighByte** (HoldingRegister ...) +
LowByte (HoldingRegister N) + **HighByte** (HoldingRegister N)

Function	Address	Gate read	Gate write	Block read
37 (obsolete)	XXXX (0...9999 decimal)	Yes	Yes	No
37:	XXXXX (0...65535 decimal)	Yes	Yes	No
25h:	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	No

Example:

These are 3 way to read data string from Holding (read/write) register Modbus address 0001

370001
 37:0001
 25h:0001h

Function **37:0 (25h:0h:)** add a string terminator word (that is 0) during the writing to device procedure.

Function	Address	Gate read	Gate write	Block read
37:0:	XXXXX (0...65535 decimal)	Yes	Yes	No
25h:0h:	XXXXh (0...FFFF Hexadecimal)	Yes	Yes	No

Example:

37:0:1000
 25h:0h:3E8h

For Input register:

Function **43 (2Bh)** extract a string from a block of input register in the following mode:

LowByte (InputRegister 1) +
LowByte (InputRegister 2) +
LowByte (InputRegister ...) +
LowByte (InputRegister N)

Function	Address	Gate read	Gate write	Block read
43 (obsolete)	XXXX (0...9999 decimal)	Yes	No	No
43:	XXXXX (0...65535 decimal)	Yes	No	No
2Bh:	XXXXh (0...FFFF Hexadecimal)	Yes	No	No

Example:

These are 3 way to read data string from Input register Modbus address 0001

430001
 43:0001
 2Bh:0001h

Function **46 (2Eh)** extract a string from a block of input register in the following mode:

HighByte (InputRegister 1) +
HighByte (InputRegister 2) +
HighByte (InputRegister ...) +
HighByte (InputRegister N)

Function	Address	Gate read	Gate write	Block read
46 (obsolete)	XXXX (0...9999 decimal)	Yes	No	No
46:	XXXXX (0...65535 decimal)	Yes	No	No
2Eh:	XXXXh (0...FFFF Hexadecimal)	Yes	No	No

Example:

These are 3 way to read data string from Input register Modbus address 0001

460001

46:0001

2Eh:0001h

Function **45 (2Dh)** extract a string from a block of input register in the following mode:

HighByte (InputRegister 1) + **LowByte** (InputRegister 1) +
HighByte (InputRegister 2) + **LowByte** (InputRegister 2) +
HighByte (InputRegister ...) + **LowByte** (InputRegister ...) +
HighByte (InputRegister N) + **LowByte** (InputRegister N)

Function	Address	Gate read	Gate write	Block read
45 (obsolete)	XXXX (0...9999 decimal)	Yes	No	No
45:	XXXXX (0...65535 decimal)	Yes	No	No
2Dh:	XXXXh (0...FFFF Hexadecimal)	Yes	No	No

Example:

These are 3 way to read data string from Input register Modbus address 0001

450001

45:0001

2Dh:0001h

Function **47 (2Fh)** extract a string from a block of input register in the following mode:

LowByte (InputRegister 1) + **HighByte** (InputRegister 1) +
LowByte (InputRegister 2) + **HighByte** (InputRegister 2) +
LowByte (InputRegister ...) + **HighByte** (InputRegister ...) +
LowByte (InputRegister N) + **HighByte** (InputRegister N)

Function	Address	Gate read	Gate write	Block read
47 (obsolete)	XXXX (0...9999 decimal)	Yes	No	No
47:	XXXXX (0...65535 decimal)	Yes	No	No
2Fh:	XXXXh (0...FFFF Hexadecimal)	Yes	No	No

Example:

These are 3 way to read data string from Input register Modbus address 0001

470001

47:0001

2Fh:0001h

19.5 Configuration

MODBUS TCP

Modbus RTU TCP Ver. 2.01

Protocol type

Modbus TCP
 Modbus RTU over TCP

502 TCP port

5000 Timeout [ms]

20 Query pause [ms]

5000 Re-connection pause [ms]

Register format

Standard mode (big-endian)
 Reverse mode (little-endian)

Device network addresses

Use the same network address for all devices
 Use a different network address for each device

Device IP address or hostname

OK Cancel Help

Modbus RTU TCP Ver. 2.01

Protocol type

Modbus TCP
 Modbus RTU over TCP

502 TCP port

5000 Timeout [ms]

20 Query pause [ms]

5000 Re-connection pause [ms]

Register format

Standard mode (big-endian)
 Reverse mode (little-endian)

Device network addresses

Use the same network address for all devices
 Use a different network address for each device

Device	IP address or hostname	UnitId
1		1
2		2
3		3
4		4
5		5
6		6
7		7
8		8
9		9

OK Cancel Help

Protocol configuration windows.

- **Protocol type:**
 - Modbus TCP:** Modbus TCP protocol
 - Modbus RTU over TCP:** Serial Modbus RTU protocol encapsulated in TCP frame
- **TCP port:** protocol communication TCP port
- **Timeout [ms]:** timeout (milliseconds) for a complete answer.

- **Query pause [ms]:** timeout between two request messages.
- **Re-connection pause [ms]:** pause between close socket and open socket in case of communication errors.
- **Register format:** order of the bytes inside the register. Can be:
 - Standard mode (big-endian)** : high byte + low byte
 - Reverse mode (little-endian)** : low byte + high byte
- **Device network address:**
 - Use the same network address for all device:** all devices associated with the channel are accessed via a single address (IP or hostname)
 - Use a different network address for each device:** each Modbus address can be associated with a different address (IP or hostname). In this mode you can also specify the UnitID with which to interrogate the device.
- **Device IP address or hostname:** IP address or hostname of Modbus devices or the relevant Modbus gateway. The following syntax are supported:

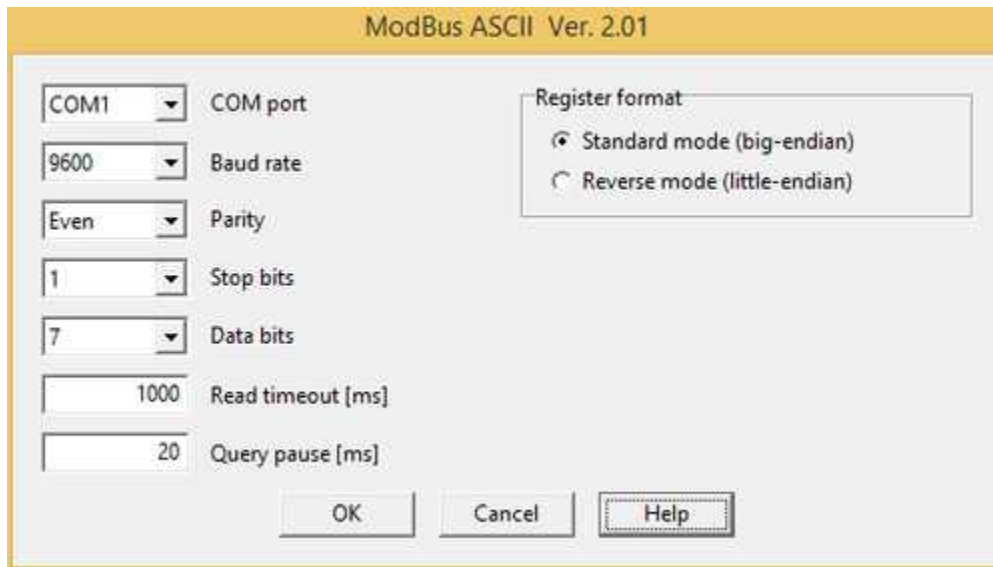
192.168.1.100	Access at 192.168.1.100 to default TCP port
mydomain.com	Access at mydomain.com to default TCP port
192.168.1.100:88	Access at 192.168.1.100 to TCP port 88 (custom)
mydomain.com:88	Access at mydomain.com to TCP port 88 (custom)

MODBUS RTU - MODBUS ASCII

ModBus RTU Ver. 2.01

COM1	COM port	Register format
9600	Baud rate	<input checked="" type="radio"/> Standard mode (big-endian)
None	Parity	<input type="radio"/> Reverse mode (little-endian)
1	Stop bits	20
8	Data bits	Silent interval characters
1000	Read timeout [ms]	
1000	Write timeout [ms]	
20	Query pause [ms]	

OK Cancel Help



Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.
- **Read timeout [ms]:** timeout (milliseconds) for a complete answer (for Read functions).
- **Write timeout [ms]:** timeout (milliseconds) for a complete answer (for Write functions).
- **Query pause [ms]:** timeout (milliseconds) between two request messages .
- **Register format:** order of the bytes inside the register. Can be:
 - Standard mode (big-endian)** : high byte + low byte
 - Reverse mode (little-endian)** : low byte + high byte
- **Silent interval characters:** number of allowed silent characters between two data characters in the same frame. Using GSM modem or Radio Modem can happen that the answer frame may be split in two parts, and the "silent interval characters" between them is over the 1,5 character as required in Modbus RTU specifications. This situation can produce a lot of communication errors or in the worst case communication can result impossible. To avoid this problem, it is possible to operate on the "Silent interval characters" by increasing it till to have a good communications without errors.

20 ODBC Client

20.1 Introduction

This driver provides an access point to ODBC data sources via **ODBC**.

ODBC (**Open database connectivity**) is a standard method to access DBMS (**Data Base Management System**) .

See "*Protocol configuration*" section for more details.

20.2 Numeric gates

The gate address must be specified in one of the following mode:

QUERY_x

where $x= 1...50$

In this first case QUERY_x represents one of the queries defined in the ODBC Client protocol configuration - "Read - SQL query" Section: it is used to read a value from ODBC data source.

QUERY_x,QUERY_y

where $x= 1...50$

where $y= 1...50$

In this second case QUERY_x always represents one of the queries defined in the ODBC Client protocol configuration - "Read - SQL Query" section: it is used to read a value from ODBC data source.

Instead QUERY_y represents one of the queries defined in the ODBC Client protocol configuration - "Write - SQL Query" section: it is used to write a value in the ODBC data source.

QUERY_x[R],QUERY_y

where $x= 1...50$

where $y= 1...50$

In this third case QUERY_x[R] represents one of the queries defined in the ODBC Client protocol configuration - "Read - SQL Query" section: it is used to read a **block of values** from ODBC data source. The values are taken as row (**[R]**): that means that each value is a different field of the same record inside the ODBC data source.

Instead QUERY_y represents one of the queries defined in the ODBC Client protocol configuration - "Write - SQL Query" section: it is used to write a value in the ODBC data source.

Note: if you need to read a block of values, the same QUERY_x must be inserted in the field address of each gate that compound the block.

See "*Protocol configuration*" section for more details.

20.3 Digital gates

The gate address must be specified in one of the following mode:

QUERY_x

where $x= 1...50$

In this first case QUERY_x represents one of the queries defined in the ODBC Client protocol configuration - "Read - SQL query" Section: it is used to read a value from ODBC data source.

QUERY_x,QUERY_y

where $x= 1...50$

where $y= 1...50$

In this second case QUERY_x always represents one of the queries defined in the ODBC Client protocol configuration - "Read - SQL Query" section: it is used to read a value from ODBC data source.

Instead QUERY_y represents one of the queries defined in the ODBC Client protocol configuration - "Write - SQL Query" section: it is used to write a value in the ODBC data source.

QUERY_x[R],QUERY_y

where $x= 1...50$

where $y= 1...50$

In this third case QUERY_x[R] represents one of the queries defined in the ODBC Client protocol configuration - "Read - SQL Query" section: it is used to read a **block of values** from ODBC data source. The values are taken as row (**[R]**): that means that each value is a different field of the same record inside the ODBC data source.

Instead QUERY_y represents one of the queries defined in the ODBC Client protocol configuration - "Write - SQL Query" section: it is used to write a value in the ODBC data source.

Note: if you need to read a block of values, the same QUERY_x must be inserted in the field address of each gate that compound the block.

See "*Protocol configuration*" section for more details.

20.4 String gates

The gate address must be specified in one of the following mode:

QUERY_x

where $x= 1...50$

In this first case QUERY_x represents one of the queries defined in the ODBC Client protocol configuration - "Read - SQL query" Section: it is used to read a value from ODBC data source.

QUERY_x,QUERY_y

where $x= 1...50$

where $y= 1...50$

In this second case QUERY_x always represents one of the queries defined in the ODBC Client protocol configuration - "Read - SQL Query" section: it is used to read a value from ODBC data source.

Instead QUERY_y represents one of the queries defined in the ODBC Client protocol configuration - "Write - SQL Query" section: it is used to write a value in the ODBC data source.

QUERY_x[R],QUERY_y

where $x= 1...50$

where $y= 1...50$

In this third case QUERY_x[R] represents one of the queries defined in the ODBC Client protocol configuration - "Read - SQL Query" section: it is used to read a **block of values** from ODBC data source. The values are taken as row (**[R]**): that means that each value is a different field of the same record inside the ODBC data source.

Instead QUERY_y represents one of the queries defined in the ODBC Client protocol configuration - "Write - SQL Query" section: it is used to write a value in the ODBC data source.

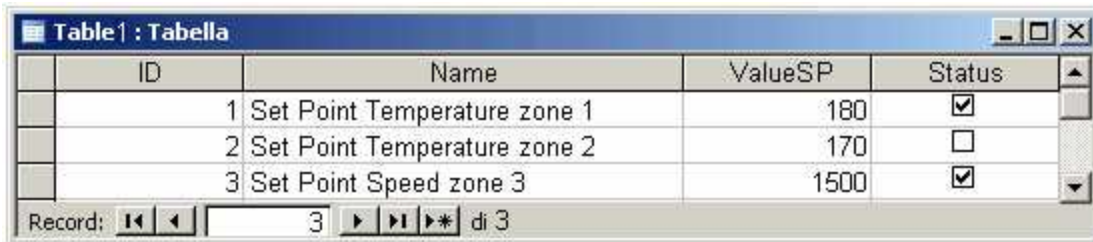
Note: if you need to read a block of values, the same QUERY_x must be inserted in the field address of each gate that compound the block.

See "*Protocol configuration*" section for more details.

20.5 Configuration

Let's see with an example how to configure the ODBC Client driver.

Suppose that you need to access the following Microsoft Access table for read and write values:



ID	Name	ValueSP	Status
1	Set Point Temperature zone 1	180	<input checked="" type="checkbox"/>
2	Set Point Temperature zone 2	170	<input type="checkbox"/>
3	Set Point Speed zone 3	1500	<input checked="" type="checkbox"/>

Figure 1

First of all you need to create a DSN (Data Source Name) to gain access to the file (Figure 2)

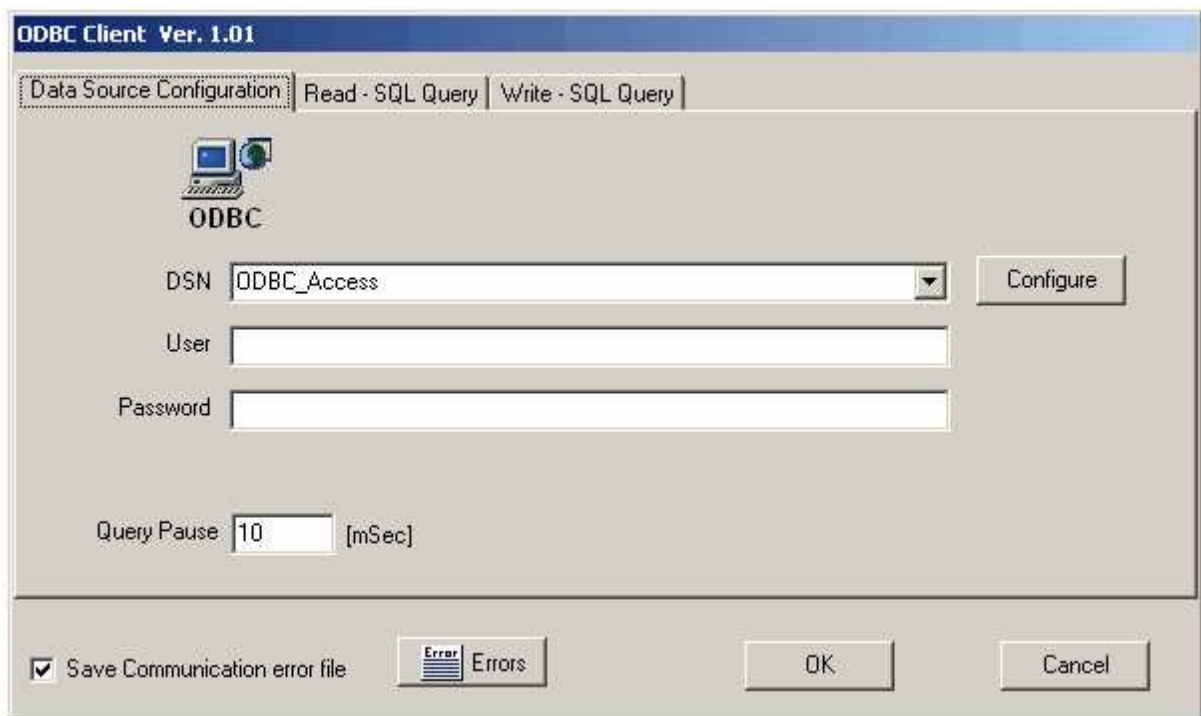


Figure 2

- **DSN:** DSN must be configured before access ODBC database. Data Source Name (DSN) must be selected from the available DSN or created as new. To create or modify a DSN push "Configure" button. System and User DSN are supported, instead File DSN are not supported.
- **Query Pause[ms]:** pause time between two request .
- **Save Communication error file :** if it is checked, a communication error message will be saved in a file on the disk every time that a communication error happens. A list of the last 100 communication error messages can be viewed (also in Runtime mode), by clicking on "Errors" button.
- **User:** user id to access database.

•**Password**: password to access database.

The second thing to do is to specify all the queries that you need to get values from the database table showed in Figure 1. Each query must be expressed as SQL language format.

For example:

1) How to read a single gate value

If you need to read as numeric gate the "ValueSP" field of record 2 of Table1 (figure 1) you must specify the following query (in the Read- SQL Query TabSheet ,Figure 3)

QUERY_1: SELECT ValueSP FROM Table1 WHERE Id=2

Using *Gate Builder* Tool, in the "Address" field of the numeric gate you must specify the position (inside the Read- SQL Query table) where the query is specified: in this case "QUERY_1".

2) How to read a block of gate values for Column

Suppose that you need to read as block of numeric gates the "ValueSP" field of record 1,2 and 3 of Table1 (figure 1): you must specify the following query (in the Read- SQL Query TabSheet ,Figure 3)

QUERY_2: SELECT ValueSP,Id FROM Table1 ORDER BY Id ASC

Using *Gate Builder* Tool, in the "Address" field of the three gates that compound the block you must specify the position (inside the Read- SQL Query table) where the query is defined: in this case "QUERY_2".

3) How to read a block of gate values for Row

Suppose that you need to read as block of string gates all the field of record 2 of Table1 (figure 1): you must specify the following query (in the Read- SQL Query TabSheet , Figure 3)

QUERY_3: SELECT Id,Name,ValueSP,Status FROM Table1 WHERE Id=2

Using "*Gate Builder*" Tool you must define 4 string gates grouped in a block.

In the "Address" field of the gates you must specify the position (inside the Read- SQL Query table) where the query is specified followed by **[R]** (that means "Read for Row"): in this case "QUERY_3[R]".

Note that each time that a read function will be made, in the first string gate will be saved the value of "Id" field, in the second string gate the value of "Name" field and so on accordingly with the order of the fields specified in the QUERY_3.

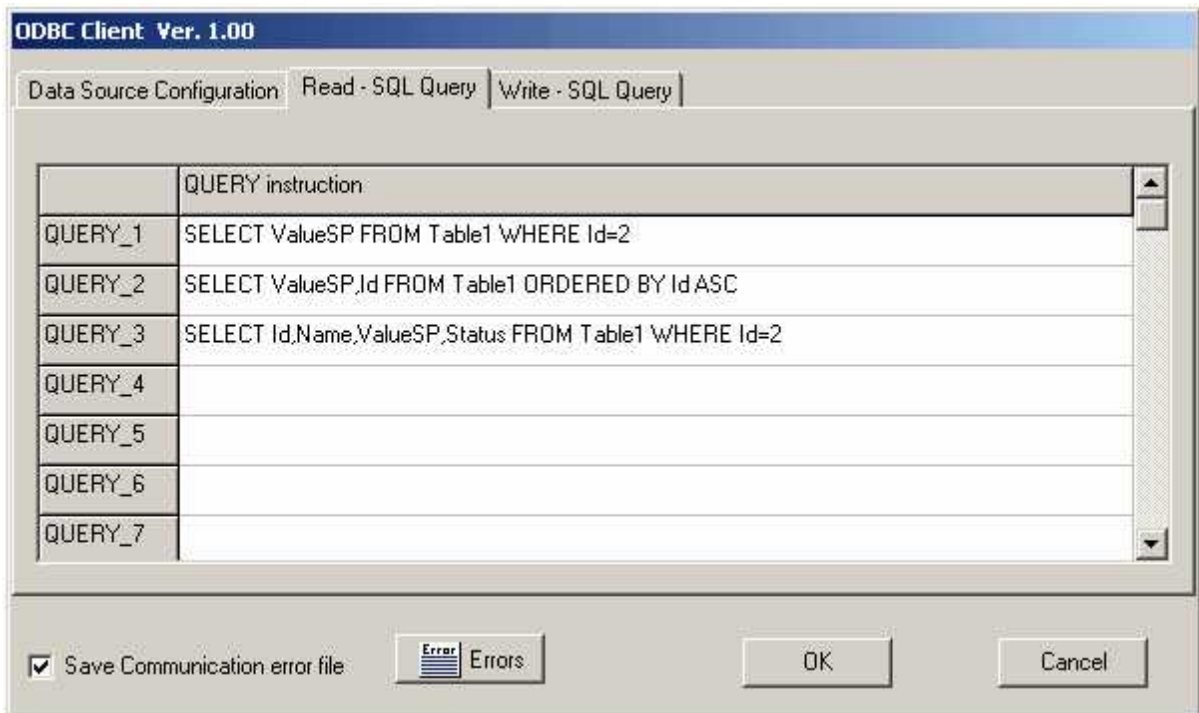


Figura 3

Query_1...Query_n: specify the query functions for read values from ODBC data source (expressed as SQL language format).

The last thing to do is to specify all the queries that you need to set values in the database table showed in Figure 1. Each query must be expressed as SQL language format.

For example:

1) How to write a numeric gate value

If you need to write a numeric gate in the "ValueSP" field of record 1 of Table1 (figure 1) you must specify the following query (in the Write- SQL Query TabSheet ,Figure 4)

QUERY_1: UPDATE Table1 SET ValueSP=%5.1f WHERE Id=1

Note that **%5.1f** inside QUERY_1 means "Insert a numeric value of 5 characters and 1 decimal": you can use also different combination like (**%6.2f**, **%4.0f** and so on...)

Using "Gate Builder" Tool, in the "Address" field of the gate, you must specify the position (inside the Write - SQL Query table) where the query is defined: in this case "QUERY_1". Note that it must be specified **after** the read query (separated by a comma) as showed below:

QUERY_3,QUERY_1 (That meas : use QUERY_3 for read and QUERY_1 for write)

2) How to write a digital gate value

If you need to write a digital gate in the "Status" field of record 1 of Table1 (figure 1)

you must specify the following query (in the Write- SQL Query TabSheet ,Figure 4)
QUERY_2: UPDATE Table1 SET Status=%d WHERE Id=1

Note that **%d** inside QUERY_2 means "Insert an integer value".

In the "Address" field of the gate (in Gate Builder) are valid the same consideration explained in "How to write a numeric gate value" example.

3) How to write a string gate value

If you need to write a string gate in the "Name" field of record 1 of Table1 (figure 1) you must specify the following query (in the Write- SQL Query TabSheet ,Figure 4)

QUERY_3: UPDATE Table1 SET Name='%s' WHERE Id=1

Note that **%s** inside QUERY_3 means "Insert a string value".

In the "Address" field of the gate (in Gate Builder) are valid the same consideration explained in "How to write a numeric gate value" example.

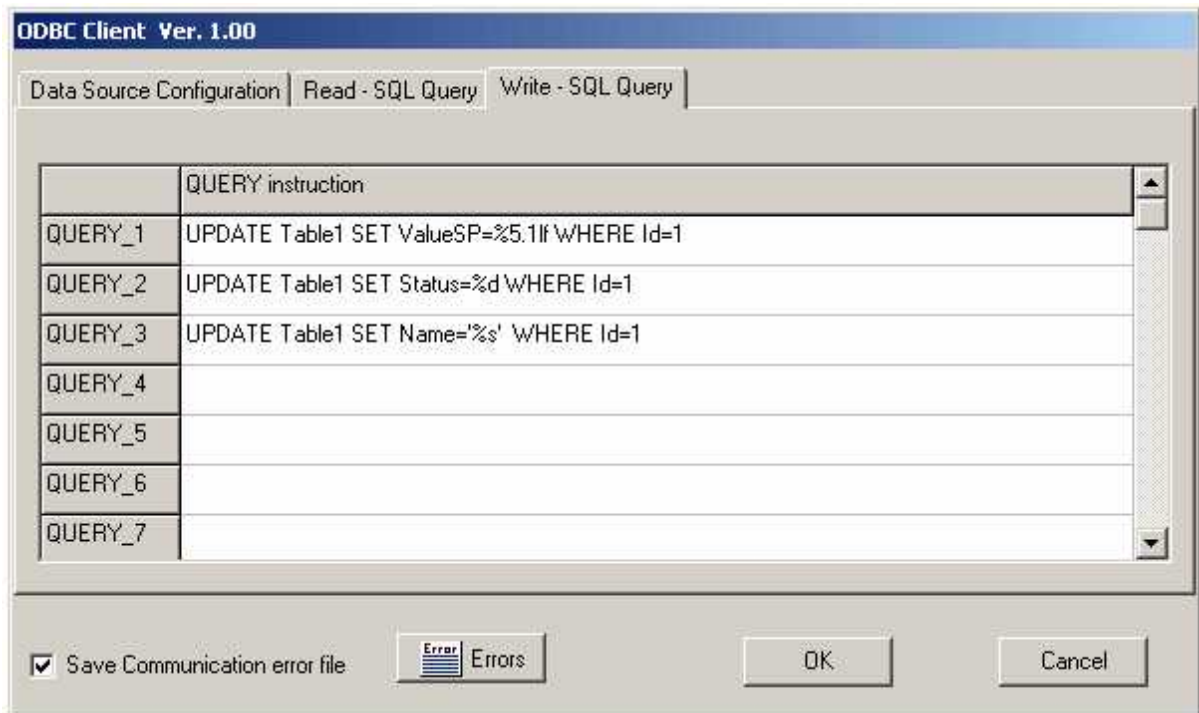


Figure 4

Query_1...Query_n: specify the query functions to write values in ODBC data source (expressed as SQL language format).

21 OMRON FINS

21.1 Introduction

OMRON FINS / UDP Ethernet communication protocol for **CJ,CS,CV** and **CP** series OMRON programmable controllers.

Data is sent and received as UDP/IP packets on a Ethernet network.

21.2 Numeric gates

The gate address is specified adding the fields **Command** and **Address** of the following tables.

OMRON CJ/CS/CP Family

Command	Description	Address	Gate read	Gate write	Block read
CIO	CIO area	XXXX 0000...6143	Yes	Yes	Yes
WR	Work area	XXX 000...511	Yes	Yes	Yes
WR_UL_	Work area unsigned long 32 bit	XXX 000...511	Yes	Yes	Yes
WR_SL_	Work area signed long 32 bit	XXX 000...511	Yes	Yes	Yes
WR_FL_	Work area Float 32 bit	XXX 000...511	Yes	Yes	Yes
HR	Holding bit area	XXX 000...511	Yes	Yes	Yes
HR_UL_	Holding bit area unsigned long 32 bit	XXX 000...511	Yes	Yes	Yes
HR_SL_	Holding bit area signed long 32 bit	XXX 000...511	Yes	Yes	Yes
HR_FL_	Holding bit area Float 32 bit	XXX 000...511	Yes	Yes	Yes
AR	Auxiliary bit area	XXX 000...447	Yes	Yes	Yes
TIM	Timer area PV	XXXX 0000...4095	Yes	Yes	Yes
CNT	Counter area PV	XXXX 0000...4095	Yes	Yes	Yes
EMn_ for n=0...7	EM bank n	XXXXX 00000...32767	Yes	Yes	Yes
EMn_UL_ for n=0...7	EM bank n unsigned long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EMn_SL_ for n=0...7	EM bank n signed long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EMn_FL_ for n=0...7	EM bank n float 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EM	EM current bank	XXXXX 00000...32767	Yes	Yes	Yes
EM_UL_	EM current bank unsigned long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EM_SL_	EM current bank signed long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EM_FL_	EM current bank float 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EM	EM current bank number		Yes	Yes	Yes
IR	Index register	XX 00...15	Yes	Yes	No
DR	Data register	XX 00...15	Yes	Yes	No
DM	DM area	XXXXX 00000...32767	Yes	Yes	Yes
DM_UL_	DM area	XXXXX	Yes	Yes	Yes

	unsigned long 32 bit	0000...32767			
DM_SL_	DM area signed long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
DM_FL_	DM area Float 32 bit	XXXXX 00000...32767	Yes	Yes	Yes

OMRON CV Family

Command	Description	Address	Gate read	Gate write	Block read
CIO	CIO area	XXXX 0000...2555	Yes	Yes	Yes
AR	Auxiliary bit area	XXX 000...447	Yes	Yes	Yes
TIM	Timer area PV	XXXX 0000...2047	Yes	Yes	Yes
CNT	Counter area PV	XXXX 0000...2047	Yes	Yes	Yes
EMn_ for n=0...7	EM bank n	XXXXX 00000...32767	Yes	Yes	Yes
EMn_UL_ for n=0...7	EM bank n unsigned long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EMn_SL_ for n=0...7	EM bank n signed long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EMn_FL_ for n=0...7	EM bank n float 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EM	EM current bank	XXXXX 00000...32767	Yes	Yes	Yes
EM_UL_	EM current bank unsigned long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EM_SL_	EM current bank signed long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EM_FL_	EM current bank float 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
EM	EM current bank number		Yes	Yes	Yes
DM	DM area	XXXXX 00000...32767	Yes	Yes	Yes
DM_UL_	DM area unsigned long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
DM_SL_	DM area signed long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
DM_FL_	DM area Float 32 bit	XXXXX 00000...32767	Yes	Yes	Yes

Example: the following are some examples of numeric gates:

DM00011 : Data Memory 00011.

CNT0004 : Counter 0004 (PV).

EM4_0005 : EM bank 4 0005.

Numeric gates block

A block of numeric gates must be made only by gates with the same function and sequential

addresses.

Example of valid block	Example of NOT valid block
DM00001	DM00001
DM00002	DM00003
DM00003	DM00005
DM00004	DM00007
DM00005	DM00008
DM00006	DM00010

21.3 Digital gates

The gate address is specified adding the fields **Command**, **Word address** and **Bit address** of the following table.

OMRON CJ/CS/CP Family

Command	Description	Address	Bit address	Gate read	Gate write	Block read
CIO	CIO area	XXXX 0000...6143	XX 00..15	Yes	Yes	Yes
WR	Work area	XXX 000...511	XX 00..15	Yes	Yes	Yes
HR	Holding bit area	XXX 000...511	XX 00..15	Yes	Yes	Yes
AR	Auxiliary bit area	XXX 000...447	XX 00..15	Yes	Yes	Yes
TIM	Timer area PV	XXXX 0000...4095	XX 00..15	Yes	Yes	Yes
CNT	Counter area PV	XXXX 0000...4095	XX 00..15	Yes	Yes	Yes
EM0_	EM bank 0	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM1_	EM bank 1	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM2_	EM bank 2	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM3_	EM bank 3	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM4_	EM bank 4	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM5_	EM bank 5	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM6_	EM bank 6	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM7_	EM bank 7	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM8_	EM bank 8	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM9_	EM bank 9	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EMA_	EM bank 10	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EMB_	EM bank 11	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EMC_	EM bank 12	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
DM	DM area	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes

OMRON CV Family

Command	Description	Address	Bit address	Gate read	Gate write	Block read
CIO	CIO area	XXXX 0000...2555	XX 00..15	Yes	Yes	Yes

AR	Auxiliary bit area	XXX 000...447	XX 00..15	Yes	Yes	Yes
TIM	Timer area PV	XXXX 0000...2047	XX 00..15	Yes	Yes	Yes
CNT	Counter area PV	XXXX 0000...2047	XX 00..15	Yes	Yes	Yes
EM0_	EM bank 0	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM1_	EM bank 1	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM2_	EM bank 2	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM3_	EM bank 3	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM4_	EM bank 4	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM5_	EM bank 5	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM6_	EM bank 6	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM7_	EM bank 7	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
DM	DM area	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes

Example: the following are some examples of digital gates:

HR00107 Holding bit area word 001 – bit 07.

TIM000412 : Timer status 0004 bit 12.

DM3014102 : Data Memory 30141 bit 02.

Digital gates block

A block of digital gates must be made only by gates with the same Command and sequential addresses and sequential bit .

Example of valid block	Example of NOT valid block
DM0000101	DM0000102
DM0000102	DM0000201
DM0000103	DM0000202
DM0000104	DM0000203
DM0000105	DM0000301
DM0000106	DM0000401

21.4 String gates

The gate address is specified adding the fields **Command** and **Address** of the following tables.

Command	Description	Address	Gate read	Gate write	Block read
DM	DM area	XXXXX 00000...32767	Yes	Yes	NO
DMR	DM area (reverse mode)	XXXXX 00000...32767	Yes	Yes	NO
EM	EM current bank	XXXXX 00000...32767	Yes	Yes	NO
EMR	EM current bank (reverse mode)	XXXXX 00000...32767	Yes	Yes	NO

Number of chars to be read corresponding to the "Max dimension" parameter specified in the GateBuilder string gate definition.

Example: the following are some examples of numeric gates:

DM00011 : Data Memory 00011.

DMR05632 : Data Memory 05632.

21.5 Configuration

Connection	CPU Model	IP address	Request size (Bytes)	Destination network address	Destination node
1	CJ-Family	192, 168, 250, 1	512	0	1
2		255, 255, 255, 255	512	0	0
3		255, 255, 255, 255	512	0	0
4		255, 255, 255, 255	512	0	0
5		255, 255, 255, 255	512	0	0
6		255, 255, 255, 255	512	0	0
7		255, 255, 255, 255	512	0	0
8		255, 255, 255, 255	512	0	0
9		255, 255, 255, 255	512	0	0
10		255, 255, 255, 255	512	0	0
11		255, 255, 255, 255	512	0	0
12		255, 255, 255, 255	512	0	0
13		255, 255, 255, 255	512	0	0
14		255, 255, 255, 255	512	0	0
15		255, 255, 255, 255	512	0	0
16		255, 255, 255, 255	512	0	0

Protocol configuration window.

It is possible to have at maximum 16 connections to OMRON PLC on the same channel. The connection number is also the number that must be specified in the "Device" item of each OMRON PLC gate defined in the Gate Builder.

- **Client IP address:** computer IP address.
- **Use local network:** when this option is selected, "Source Network Address" will be set to 0 (that means "Local Network") and "Source Node" will be set equal to the last byte of the Client IP address (in the example is 12 that is the last byte of 192.168.250.12). If *Use Local Network* is not selected then *Source Network Address* and *Source Node* can be set from the user.
- **Port number:** ethernet port utilized for communication.
- **Communication timeout:** maximum time to wait for a complete answer.
- **Query pause:** time to wait between two request.
- **CPU model:** PLC family type (CJ,CS,CV or CP).
- **IP address:** PLC IP address.
- **Request size:** maximum size of communication buffer between Personal Computer and PLC.
- **Destination network address:** PLC Network Address (0=Local Network).
- **Destination node :** PLC Node Address.
- **Save communication error file :** if it is checked, a communication error message will be saved in a file on the disk every time that a communication error happens. A list of the last 100 communication

error messages can be viewed (also in Runtime mode), by clicking on "Errors" button.

22 OMRON FINS in Host Link Protocol

22.1 Introduction

OMRON FINS frame in **Host Link** communication protocol for **CJ,CS** and **CV** series OMRON programmable controllers.

Passes Host Link standard limitations: for example it allow to reach DM higher than 9999.

Communication between PC and PLC requires a RS232/RS485 converter.

22.2 Numeric gates

The gate address is specified adding the fields **Command** and **Address** of the following tables.

OMRON CJ/CS Family

Command	Description	Address	Gate read	Gate write	Block read
CIO	CIO area	XXXX 0000...6143	Yes	Yes	Yes
WR	Work area	XXX 000...511	Yes	Yes	Yes
HR	Holding bit area	XXX 000...511	Yes	Yes	Yes
AR	Auxiliary bit area	XXX 000...447	Yes	Yes	Yes
TIM	Timer area PV	XXXX 0000...4095	Yes	Yes	Yes
CNT	Counter area PV	XXXX 0000...4095	Yes	Yes	Yes
EM0_	EM bank 0	XXXXX 00000...32767	Yes	Yes	Yes
EM1_	EM bank 1	XXXXX 00000...32767	Yes	Yes	Yes
EM2_	EM bank 2	XXXXX 00000...32767	Yes	Yes	Yes
EM3_	EM bank 3	XXXXX 00000...32767	Yes	Yes	Yes
EM4_	EM bank 4	XXXXX 00000...32767	Yes	Yes	Yes
EM5_	EM bank 5	XXXXX 00000...32767	Yes	Yes	Yes
EM6_	EM bank 6	XXXXX 00000...32767	Yes	Yes	Yes
EM7_	EM bank 7	XXXXX 00000...32767	Yes	Yes	Yes
EM8_	EM bank 8	XXXXX 00000...32767	Yes	Yes	Yes
EM9_	EM bank 9	XXXXX 00000...32767	Yes	Yes	Yes
EMA_	EM bank 10	XXXXX 00000...32767	Yes	Yes	Yes
EMB_	EM bank 11	XXXXX 00000...32767	Yes	Yes	Yes
EMC_	EM bank 12	XXXXX 00000...32767	Yes	Yes	Yes
EM	EM current bank	XXXXX 00000...32767	Yes	Yes	Yes
EM	EM current bank number		Yes	Yes	Yes

IR	Index register	XX 00...15	Yes	Yes	No
DR	Data register	XX 00...15	Yes	Yes	No
DM	DM area	XXXXX 00000...32767	Yes	Yes	Yes
DM_UL_	DM area unsigned long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
DM_SL_	DM area signed long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
DM_FL_	DM area Float 32 bit	XXXXX 00000...32767	Yes	Yes	Yes

OMRON CV Family

Command	Description	Address	Gate read	Gate write	Block read
CIO	CIO area	XXXX 0000...2555	Yes	Yes	Yes
AR	Auxiliary bit area	XXX 000...447	Yes	Yes	Yes
TIM	Timer area PV	XXXX 0000...2047	Yes	Yes	Yes
CNT	Counter area PV	XXXX 0000...2047	Yes	Yes	Yes
EM0_	EM bank 0	XXXXX 00000...32767	Yes	Yes	Yes
EM1_	EM bank 1	XXXXX 00000...32767	Yes	Yes	Yes
EM2_	EM bank 2	XXXXX 00000...32767	Yes	Yes	Yes
EM3_	EM bank 3	XXXXX 00000...32767	Yes	Yes	Yes
EM4_	EM bank 4	XXXXX 00000...32767	Yes	Yes	Yes
EM5_	EM bank 5	XXXXX 00000...32767	Yes	Yes	Yes
EM6_	EM bank 6	XXXXX 00000...32767	Yes	Yes	Yes
EM7_	EM bank 7	XXXXX 00000...32767	Yes	Yes	Yes
EM	EM current bank	XXXXX 00000...32767	Yes	Yes	Yes
EM	EM current bank number		Yes	Yes	Yes
DM	DM area	XXXXX 00000...32767	Yes	Yes	Yes
DM_UL_	DM area unsigned long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
DM_SL_	DM area signed long 32 bit	XXXXX 00000...32767	Yes	Yes	Yes
DM_FL_	DM area Float 32 bit	XXXXX 00000...32767	Yes	Yes	Yes

Example: the following are some examples of numeric gates:
DM00011 : Data Memory 00011.
CNT0004 : Counter 0004 (PV).

Numeric gates block

A block of numeric gates must be made only by gates with the same function and sequential addresses.

Example of valid block	Example of NOT valid block
DM00001	DM00001
DM00002	DM00003
DM00003	DM00005
DM00004	DM00007
DM00005	DM00008
DM00006	DM00010

22.3 Digital gates

The gate address is specified adding the fields **Command**, **Word address** and **Bit address** of the following table.

OMRON CJ/CS Family

Command	Description	Address	Bit address	Gate read	Gate write	Block read
CIO	CIO area	XXXX 0000...6143	XX 00..15	Yes	Yes	Yes
WR	Work area	XXX 000...511	XX 00..15	Yes	Yes	Yes
HR	Holding bit area	XXX 000...511	XX 00..15	Yes	Yes	Yes
AR	Auxiliary bit area	XXX 000...447	XX 00..15	Yes	Yes	Yes
TIM	Timer area PV	XXXX 0000...4095	XX 00..15	Yes	Yes	Yes
CNT	Counter area PV	XXXX 0000...4095	XX 00..15	Yes	Yes	Yes
EM0_	EM bank 0	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM1_	EM bank 1	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM2_	EM bank 2	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM3_	EM bank 3	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM4_	EM bank 4	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM5_	EM bank 5	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM6_	EM bank 6	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM7_	EM bank 7	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM8_	EM bank 8	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM9_	EM bank 9	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EMA_	EM bank 10	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EMB_	EM bank 11	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EMC_	EM bank 12	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
DM	DM area	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes

OMRON CV Family

Command	Description	Address	Bit address	Gate read	Gate write	Block read
CIO	CIO area	XXXX 0000...2555	XX 00..15	Yes	Yes	Yes
AR	Auxiliary bit area	XXX 000...447	XX 00..15	Yes	Yes	Yes
TIM	Timer area PV	XXXX 0000...2047	XX 00..15	Yes	Yes	Yes
CNT	Counter area PV	XXXX 0000...2047	XX 00..15	Yes	Yes	Yes
EM0_	EM bank 0	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM1_	EM bank 1	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM2_	EM bank 2	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM3_	EM bank 3	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM4_	EM bank 4	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM5_	EM bank 5	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM6_	EM bank 6	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
EM7_	EM bank 7	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes
DM	DM area	XXXXX 00000...32767	XX 00..15	Yes	Yes	Yes

Example: the following are some examples of digital gates:

HR00107 Holding bit area word 001 – bit 07.

TIM000412 : Timer status 0004 bit 12.

DM3014102 : Data Memory 30141 bit 02.

Digital gates block

A block of digital gates must be made only by gates with the same Command and sequential addresses and sequential bit .

Example of valid block	Example of NOT valid block
DM0000101	DM0000102
DM0000102	DM0000201
DM0000103	DM0000202
DM0000104	DM0000203
DM0000105	DM0000301
DM0000106	DM0000401

22.4 String gates

The gate address is specified adding the fields **Command** and **Address** of the following tables.

Command	Description	Address	Gate read	Gate write	Block read
DM	DM area	XXXXX 00000...32767	SI	SI	NO
DMR	DM area (reverse mode)	XXXXX 00000...32767	SI	SI	NO

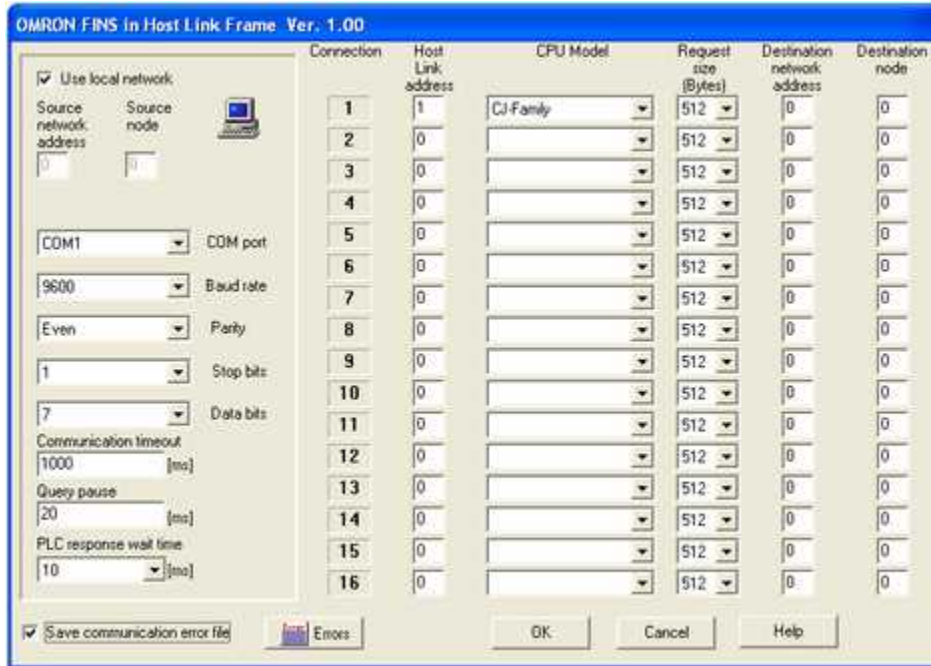
Number of chars to be read corresponding to the "Max dimension" parameter specified in the GateBuilder string gate definition.

Example: the following are some examples of numeric gates:

DM00011 : Data Memory 00011.

DMR05632 : Data Memory 05632.

22.5 Configuration



Protocol configuration window.

It is possible to have at maximum 16 connections to OMRON PLC on the same channel. The connection number is also the number that must be specified in the "Device" item of each OMRON PLC gate defined in the Gate Builder.

- **Use Local Network:** when this option is selected, "Source Network Address" will be set to 0 (that means "Local Network") . If *Use Local Network* is not selected then *Source Network Address* and *Source Node* can be set from the user.
- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.
- **Communication timeout [ms]:** total timeout (milliseconds) waiting for answer.
- **Query pause [ms]:** timeout between answer and next request .
- **PLC response wait time [ms]:** delay of PLC answer .
- **Host link address :** PLC Host Link address.
- **CPU model:** PLC family type (CJ,CS or CV).
- **IP address:** PLC IP address.
- **Request size:** maximum size of communication buffer between Personal Computer and PLC.
- **Destination network address:** PLC Network Address (0=Local Network).
- **Destination node :** PLC Node Address.
- **Save communication error file :** if it is checked, a communication error message will be saved in a file on the disk every time that a communication error happens. A list of the last 100 communication error messages can be viewed (also in Runtime mode), by clicking on "Errors" button.

23 OMRON SYSMAC

23.1 Introduction

Communication protocol for the following OMRON PLC's:

CQM1,C200H,C200HS,C200HE,C200HG,C200HX,C1000H,C2000H,CVM1, CPM1,C-H,C-K CJ1 and CS1.

Communication between PC and PLC requires a RS232/RS485 converter; up to 32 devices can be connected to the same serial link.

23.2 Numeric gates

The gate address is specified adding the fields Function and Address of the following table.

Function	Description	Word address	Gate read	Gate write	Block read
DM	Data Memory	XXXX decimal value	Yes	Yes	Yes
DM_UL_	Data memory Unsigned Long	XXXX decimal value	Yes	Yes	Yes
DM_SL_	Data memory Signed Long	XXXX decimal value	Yes	Yes	Yes
PV	Timer Preset Value	XXXX decimal value	Yes	Yes	Yes
MS	Read PLC operation conditions		Yes	No	No
SC	Modify PLC operation conditions		No	Yes	No
MF	Read PLC errors		Yes	No	No
IR	Input Output Relay	XXXX Decimal value of the status word of 16 I/O relay	Yes	Yes	Yes
LR	Link Relay	XXXX Decimal value of the status word of 16 Link Relay	Yes	Yes	Yes
HR	Holding Relay	XXXX Decimal value of the status word of 16 Holding Relay	Yes	Yes	Yes

Example: the following are some examples of numeric gates:

DM0011 : Data Memory 0011.

PV0004 : Timer 0004.

MF : PLC errors read.

Note:

The **MF** function associated to a numeric gate gives back the 32 bit which correspond to error information of first channel (16 most significant bits) and second channel (16 less significant bits); this function gives back a text string that can be read using a string gate.

The **PV** function return a numeric value already converted in BDC format (Binary Cided Decimal).

A block of numeric gates must be made only by gates with the same function and sequential addresses.

Example of valid block	Example of NOT valid block
DM0003	DM0003
DM0004	DM0005
DM0005	PV0012
DM0006	DM0013
DM0007	DM0014

23.3 Digital gates

The gate address is specified adding the fields Function, Word address and Bit address of the following table.

Function	Description	Word address	Bit address	Gate read	Gate write	Block read
IR	INPUT OUTPUT RELAY	XXXX 0000...9999	XX 00...15	Yes	Yes	Yes
LR	LINK RELAY	XXXX 0000...9999	XX 00...15	Yes	Yes	Yes
HR	HOLDING RELAY	XXXX 0000...9999	XX 00...15	Yes	Yes	Yes
TC	TIMER COUNTER STATUS	XXXX 0000...9999		Yes	Yes	Yes

Example: the following are some examples of digital gates:

IR000312 I/O relay word 0003 – bit 12.

TC0004 : Timer/counter status 4.

Note:

Digital gates can be grouped to form two different block types.

The first block type is made of **IR**, **LR**, **HR** type gates. A block must be made only by gates with the same function and the same or the sequential address.

Example of valid block	Example of valid block	Example of NOT valid block	Example of NOT valid block
IR000401	HR000400	IR000401	HR000400
IR000402	HR000401	IR000402	HR000401
IR000403	HR000402	IR000403	HR000402
IR000501	HR000403	IR000801	IR000403
IR000504	HR000404	IR000804	IR000404
IR000505	HR000405	IR000905	IR000405
IR000601	HR000409	IR001001	HR000409
IR000701	HR000410	IR001101	HR000410
IR000804	HR000411	IR001304	HR000411
IR000805	HR000412	IR001305	HR000412
IR000812	HR000413	IR001412	HR000413

The second block type is made of **TC** type gates. A block must be made only by gates with the same function and sequential.

Example of valid block	Example of NOT valid block
TC0001	TC0001
TC0002	TC0002
TC0003	TC0005
TC0004	TC0006
TC0005	TC0007

23.4 String gates

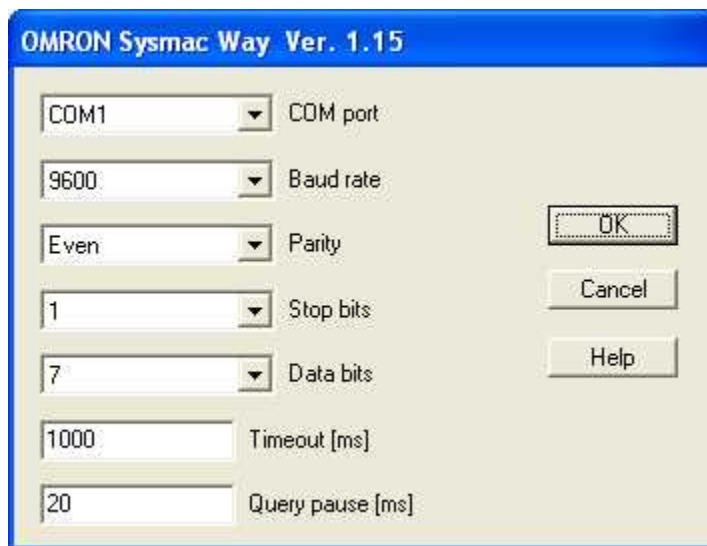
The gate address is specified by the field Function of the following table.

Function	Description	Gate read	Gate write	Block read
MS	READ PLC OPERATION	Yes	No	No

Note:

If the function **MF** associated to a string gate gives back the 16 characters following status data; status data given back by this function can be read using a numeric.

23.5 Configuration



Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages.

24 OPC Client

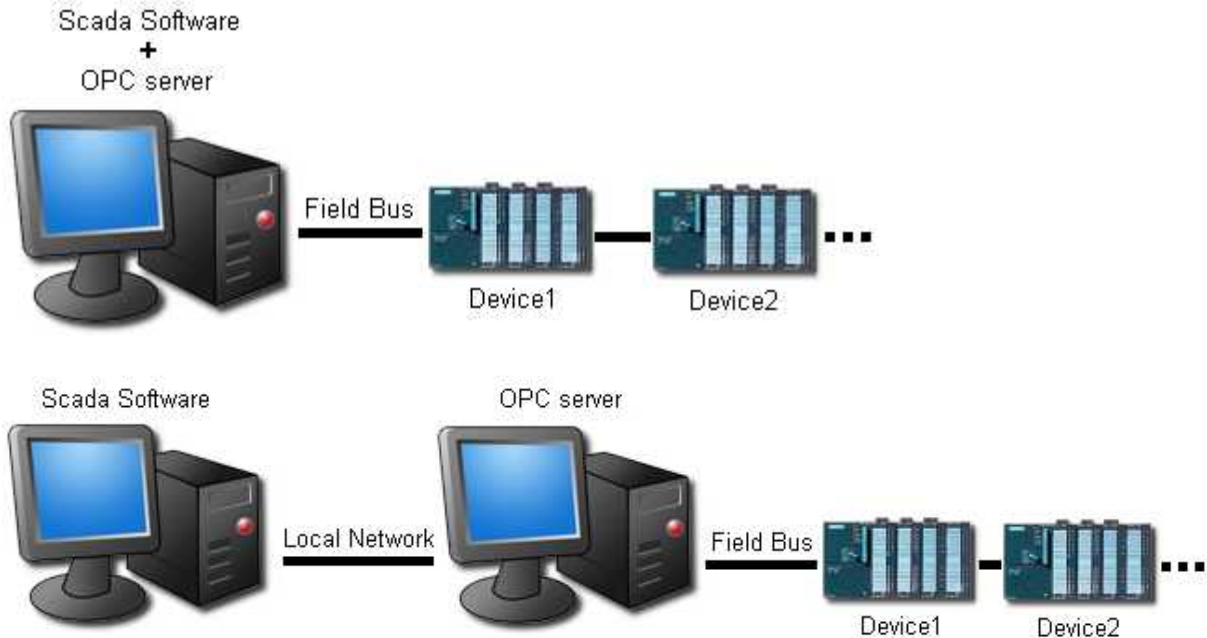
24.1 Introduction

OPC (OLE for Process Control) is an industry standard created in collaboration with a number of worldwide leading automation hardware and software manufacturers. OPC allows software

components such as software connectors to be combined and enables these components to intercommunicate with no need of special adaptations.

This **OPC Client** driver support data access (**DA**) to OPC servers **DA 1.0** , **2.0** and **3.0**. It can connect to local OPC servers through COM object or to remote OPC servers (in the local network) through DCOM object.

In case of remote server, you must ensure that DCOM is properly configured. **DCOMCNFG** is a Windows tool that allows users to configure the DCOM settings. Before you can access a COM component via DCOM, you must provide the authentication credentials of a user who has been granted permission to access/launch the component.



DA Server is organized in a structure of groups and items that are directly connected to device or PLC internal variables, so the problem of communication protocol with them is solved by the specific OPC server. **OPC Client** communicates with OPC servers always in the same mode without need to know the specific device or PLC communication protocol.

The first thing to do is to install the OPC server on the computer, and configure it defining communication parameters and items (devices read or write variables). The Item full name (usually compound by DeviceName+GroupName+ItemId) does not be more than 80 chars.

After that open ProjectManager and create a new project, then select *ProjectManager->Configuration->Channel* and choose **OPC Client** protocol and configure it by selecting the computer and OPC server name to connect to.

At this point open GateBuilder and define all the gates that you want to read and write from OPC server.

The address of each gate is the Item ID defined in the OPC server. You can browse all the available items in the server by clicking on the button that appear on the right side of the address field of the gate.

24.2 Numeric gates

The address of each gate is the Item ID defined in the OPC server. You can browse all the available items in the server by clicking on the button that appear on the right side of the address field of the gate.

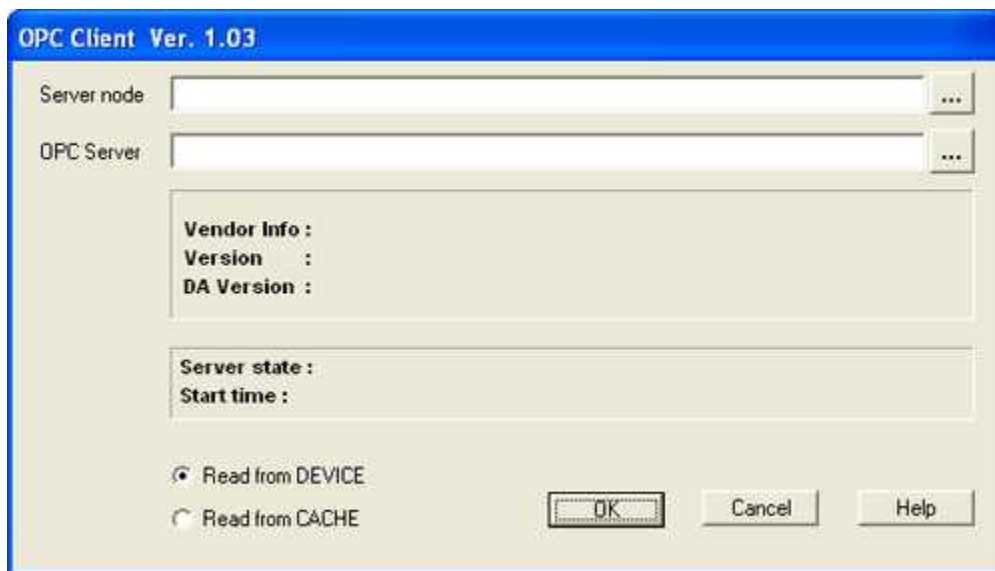
24.3 Digital gates

The address of each gate is the Item ID defined in the OPC server. You can browse all the available items in the server by clicking on the button that appear on the right side of the address field of the gate.

24.4 String gate

The address of each gate is the Item ID defined in the OPC server. You can browse all the available items in the server by clicking on the button that appear on the right side of the address field of the gate.

24.5 Configuration



Protocol configuration window.

- **Server node:** computer name where is installed the OPC server to connect to.
- **OPC Server:** name of the OPC server to connect to.
- **Read from DEVICE:** data will be read directly from the device.
- **Read from CACHE:** data will be read from the OPC server cache memory.

25 RED LION PAXI-1/8 DIN COUNTER/RATE METER

25.1 Introduction

This communication protocol is used by RED LION PAXI devices.

25.2 Numeric gates

The address of a numeric gate is specified by the ID register which can be found on the device manual (see "Register Identification Chart").

Register ID	Gate read	Gate write	Block read
X	Yes	Yes	No

Examples: here are some examples of numeric gates:

A : Count A.

M : SetPoint 1.

S : SetPoint 4.

U: Auto/Manual Register

25.3 Digital gates

Digital gates are not allowed in this protocol.

25.4 String gates

String gates are not allowed in this protocol.

25.5 Configuration

Protocol configuration window.

- **COM port:** serial port name.

- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.
- **String termination char:** message terminator character (“*” o “\$”).
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages.

26 SAIA P800

26.1 Introduction

This protocol is used for communication with SAIA PLCs series:
PCD1,PCD2,PCD2.M220,PCD4,PCD6.

Communication between PC and PLC is performed via the standard PC serial link; only one device can be.

26.2 Numeric gates

The gate address is specified adding the fields Function and Address of the following table.

Function	Description	Address	Gate read	Gate write	Block read
R	REGISTER	XXXX	Yes	Yes	Yes
C	COUNTER	XXXX	Yes	Yes	Yes
T	TIMER	XXXX	Yes	Yes	Yes
S	CPU STATUS	X 0...7	Yes	No	No
D	DISPLAY REGISTER		Yes	No	No
K_WOY	REAL TIME CLOCK WEEK OF YEAR		No	No	Yes
K_DOW	REAL TIME CLOCK DAY OF WEEK		No	No	Yes
K_YEA	REAL TIME CLOCK YEAR		No	No	Yes
K_MON	REAL TIME CLOCK MONTH		No	No	Yes
K_DAY	REAL TIME CLOCK DAY		No	No	Yes
K_HOU	REAL TIME CLOCK HOURS		No	No	Yes
K_MIN	REAL TIME CLOCK MINUTES		No	No	Yes
K_SEC	REAL TIME CLOCK SECONDS		No	No	Yes

Example: here are some examples of numeric gates:

R0123 : Register 0123.

S5 : Cpu 5 status .

D : Display register.

Note:

The gates of the Real Time Clock must always defined as a block and must have the following sequence: **K_WOY, K_DOW, K_YEA, K_MON, K_DAY, K_HOU, K_MIN, K_SEC.**

A block can have a maximum of 32 numeric gates

A block of numeric gates (with the exception of the Real Time Clock) must have only gates with the same function and sequential address.

Example of valid block	Example of NOT valid block
R0003	R0003
R0004	R0005
R0005	S0012
R0006	R0013
R0007	R0014

26.3 Digital gates

The gate address is specified adding the fields Function and Bit address of the following table.

Function	Description	Bit address	Gate read	Gate write	Block read
F	FLAG	XXXX	Yes	Yes	Yes
O	OUTPUT	XXXX	Yes	No	Yes
I	INPUT	XXXX	Yes	No	No

Example: here are some examples of digital gate address:

F0031 : Flag 0031.

O0004 : Output 0004.

Note:

A block can have a maximum of 128 digital gates

A block of digital must have only gates with the same function and sequential bit address.

Example of valid block	Example of NOT valid block
F0003	F0003
F0004	F0005
F0005	I0012
F0006	F0013
F0007	F0014

26.4 String gates

The string address is specified by the Function field of the following.

Function	Description	Gate read	Gate write	Block read
VER	CPU VERSION READ	Yes	No	No

26.5 Configuration



Protocol configuration window.

- **COM port**: serial port name.
- **Baud rate**: communication speed.
- **Timeout [ms]**: timeout (milliseconds) before answer message.
- **Query pause [ms]**: timeout between two request messages.

27 SAIA S-BUS

27.1 Introduction

This protocol is used for communication with SAIA PLCs series:
PCD1,PCD2,PCD2.M220,PCD4,PCD6.

Communication between PC and PLC is performed via the standard PC serial link but requires a RS232/RS485 converter; up to 255 devices can be connected to the same serial link.

27.2 Numeric gates

The gate address is specified adding the fields Function and Address of the following table.

Function	Description	Address	Gate read	Gate write	Block read
R	REGISTER	XXXX	Yes	Yes	Yes
C	COUNTER	XXXX	Yes	Yes	Yes
T	TIMER	XXXX	Yes	Yes	Yes
S	CPU STATUS	X 0...7	Yes	No	No
D	DISPLAY REGISTER		Yes	No	No
K_WOY	REAL TIME CLOCK WEEK OF YEAR		No	No	Yes
K_DOW	REAL TIME CLOCK DAY OF WEEK		No	No	Yes
K_YEA	REAL TIME CLOCK YEAR		No	No	Yes
K_MON	REAL TIME CLOCK MONTH		No	No	Yes
K_DAY	REAL TIME CLOCK DAY		No	No	Yes
K_HOU	REAL TIME CLOCK HOURS		No	No	Yes
K_MIN	REAL TIME CLOCK MINUTES		No	No	Yes
K_SEC	REAL TIME CLOCK SECONDS		No	No	Yes

Example: here are some examples of numeric gates address:

R0123 : Register 0123.

S5 : Cpu 5 status .

D : Display register.

Note:

The gates of the Real Time Clock must always defined as a block and must have the following sequence: **K_WOY, K_DOW, K_YEA, K_MON, K_DAY, K_HOU, K_MIN, K_SEC.**

A block can have a maximum of 32 numeric gates

A block of numeric gates (with the exception of the Real Time Clock) must have only gates with the same function and sequential address.

Example of valid block	Example of NOT valid block
R0003	R0003
R0004	R0005
R0005	S0012
R0006	R0013
R0007	R0014

27.3 Digital gates

The gate address is specified adding the fields Function and Bit address of the following table

Function	Description	Bit address	Gate read	Gate write	Block read
F	FLAG	XXXX	Yes	Yes	Yes
O	OUTPUT	XXXX	Yes	No	Yes
I	INPUT	XXXX	Yes	No	No

Example: here are some examples of digital gate address:

F0031 : Flag 0031.

O0004 : Output 0004.

Note:

A block can have a maximum of 128 digital gates

A block of digital gates must have only gates with the same function and sequential bit address.

Example of valid block	Example of NOT valid block
F0003	F0003
F0004	F0005
F0005	I0012
F0006	F0013
F0007	F0014

27.4 String gates

The string address is specified by the Function field of the following.

Function	Description	Gate read	Gate write	Block read
VER	CPU VERSION READ	Yes	No	No

27.5 Configuration

Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** timeout between two request messages.
- **Training sequence delay [ms] :** values of this parameter depend from Baud Rate according to the following table

Baud rate	Timeout	Training sequence delay in Windows 95	Training sequence delay in Windows NT
110	1000	0	50
150	1000	0	46
300	500	0	42
600	500	0	20
1200	500	0	10
2400	500	0	10
4800	500	0	10
9600	250	0	10
19200	200	0	10
38400	200	0	10

28 SIEMENS MPI

28.1 Introduction

MPI communication protocol for S7-200 S7-300 S7-400 S7-1200 Siemens and VIPA PLC.

Supported communication:

- via **COM** port using **SIEMENS SIMATIC S7 - PC Adapter V5.1** - Code 6ES7 972-0CA23-0XA0
- via **USB** port connected to a virtual **COM** using **Sielco Sistemi IC 3580-MPIVC**
- via **USB** port using **SIEMENS SIMATIC S7 - PC Adapter USB** - Code 6ES7 972-0CB20-0XA0
- via **ethernet** card.

With this driver is possible to access the following PLC data type:

For S7-300 , S7-400 and S7-1200 PLC family:

Data Block
Digital Input
Digital Output
Timer
Counter
Flags

For S7-200 PLC family:

V area
Digital Input
Digital Output
Analog Input
Analog output

28.2 Numeric gates

Note: the "Device" field in the GateBuilder must match the Plc Station Address.

Note 1: during PLC configuration, Data Block (DB) on S7-1200 must be defined with the flag "Symbolic access only" deactivated.

Numeric gates can be:

for S7-300,S7-400 and S7-1200 PLC family:**Input(E), Output(O), Timers(T), Counters (C), DataBase(DB),Merker(M)**

for S7-200 PLC family:**Digital Input(E), Digital Output(O), Analog Input(AE), Analog Output(AO), V-area(V).**

For some data type (e.g. DB) different data format can be specified: Byte, Word,BCD, Long, Float.

Note that the data address refers to Byte alignment. That mean :

PLC Memory	Data type "B"	Data type "W"	Data type "X"	Data type "L"	Data type "F"
DB9.0	B.009.0000	W.009.0000	X.009.0000	L.009.0000	F.009.0000
DB9.1	B.009.0001				
DB9.2	B.009.0002	W.009.0002	X.009.0002		
DB9.3	B.009.0003				
DB9.4	B.009.0004	W.009.0004	X.009.0004	L.009.0004	F.009.0004
DB9.5	B.009.0005				
DB9.6	B.009.0006	W.009.0006	X.009.0006		
DB9.7	B.009.0007				
DB9.8	B.009.0008	W.009.0008	X.009.0008	L.009.0008	F.009.0008
DB9.9	B.009.0009				
DB9.10	B.009.0010	W.009.0010	X.009.0010		
DB1.11	B.009.0011				

Addresses of numeric gates like Input, Output, Timers, Counter,Flags,V area:

* About V area on S7-1200 see Note 1 at the begin of page

Data type	Data format	Data address	Gate read	Gate write	Read block	Example
E (Input german mode) S7-200/300/400/ 1200	B (Byte)	XXXXX Decimal value (0..99999)	Yes	No	Yes	EB2
E (Input german mode) S7-200/300/400/ 1200	W (word)	XXXXX Decimal value (0..99999)	Yes	No	Yes	EW2
I (Input english mode) S7-200/300/400/ 1200	B (Byte)	XXXXX Decimal value (0..99999)	Yes	No	Yes	IB2
I (Input english mode) S7-200/300/400/ 1200	W (Word)	XXXXX Decimal value (0..99999)	Yes	No	Yes	IW2
A (Output german mode) S7-200/300/400/ 1200	B (Byte)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	AB2
A (Output german mode)	W (Word)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	AW2

S7-200/300/400/ 1200						
Q (Output english mode) S7-200/300/400/ 1200	B (Byte)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	QB2
Q (Output english mode) S7-200/300/400/ 1200	W (Word)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	QW2
M (Merker german mode) S7-300/400/ 1200	B (Byte)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	MB2
M (Merker german mode) S7-300/400/ 1200	W (Word)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	MW2
M (Merker german mode) S7-300/400/ 1200	D (Long)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	MD2
M (Merker german mode) S7-300/400/ 1200	F (Float)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	MF2
F (Merker english mode) S7-300/400/ 1200	B (Byte)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	FB2
F (Merker english mode) S7-300/400/ 1200	W (Word)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	FW2
F (Merker english mode) S7-300/400/ 1200	D (Long)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	FD2
F (Merker english mode) S7-300/400/ 1200	F (Float)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	FF2
T (Timer) S7-300/400/ 1200		XX Decimal value (0..99)	Yes	No	Yes	T5
Z (Counter german mode) S7-300/400/ 1200		XX Decimal value (0..99)	Yes	No	Yes	Z5
C (Counter english mode) S7-300/400/ 1200		XX Decimal value (0..99)	Yes	No	Yes	C5
AE (Analog Input german mode) S7-200	W (Word)	XXXXX Decimal value (0..99999)	Yes	No	Yes	AEW9
AI (Analog Input english mode) S7-200	W (Word)	XXXXX Decimal value (0..99999)	Yes	No	Yes	AIW9
AA (Analog Input german mode) S7-200	W (Word)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	AAW7
AQ (Analog Input english mode) S7-200	W (Word)	XXXXX Decimal value (0..99999)	Yes	Yes	Yes	AQW7
V (V area) S7-200/1200	B (Byte)	XXXXX Decimal value (0..65535)	Yes	Yes	Yes	VB10
V	W	XXXXX	Yes	Yes	Yes	VW10

(V area) S7-200/1200	(Word)	Decimal value (0..65535)				
V (V area) S7-200/1200	X (BCD format)	XXXXX Decimal value (0..65535)	Yes	Yes	Yes	VX10
D (V area) S7-200/1200	D Double (Word)	XXXXX Decimal value (0..65535)	Yes	Yes	Yes	VD8
F (V area) S7-200/1200	F (Float Word)	XXXXX Decimal value (0..65535)	Yes	Yes	Yes	VF8

Addresses of numeric gates like DB (for S7-300,S7-400,S7-1200):

* About DB area on S7-1200 see Note 1 at the begin of page

DB	DB number	.	Data type	Data format	Data address	Gate read	Gate write	Block read	Example
DB	XXXXX (1..32767)	.	DB	B (Byte)	XXXXX (0..65535)	Yes	Yes	Yes	DB2.DBB3
DB	XXXXX (1..32767)	.	DB	W (Word)	XXXXX (0..65535)	Yes	Yes	Yes	DB2.DBW4
DB	XXXXX (1..32767)	.	DB	X (BCD)	XXXXX (0..65535)	Yes	Yes	Yes	DB2.DBX6
DB	XXXXX (1..32767)	.	DB	D (Long)	XXXXX (0..65535)	Yes	Yes	Yes	DB2.DBD4
DB	XXXXX (1..32767)	.	DB	F (Float)	XXXXX (0..65535)	Yes	Yes	Yes	DB2.DBF8

Numeric gates block of same data type with consecutive address

To improve speed communications between PC and PLC, use of Blocks sampling **is recommended**. **Input, Output, Timers, Counters** gates block must be compound from gates that have the same Data Type but can have different Data Format and that have a consecutive address in increasing order (in relation to the data format).

DataBase block gates must be compound from gates that have the same DB number, but can have different data format and that have a consecutive address in increasing order (in relation to the data format).

The block length is related to the gates data format grouped in the block.

Data format	Max block length
BYTE	200
WORD	100
BCD	100
LONG	50
FLOAT	50
TIMER	100
COUNTER	100

Numeric gates that can be grouped as block	Numeric gates that can be grouped as block
E10	DB1.DBB10
E11	DB1.DBB11
E12	DB1.DBW12
E13	DB1.DBW14
E14	DB1.DBX16
E15	DB1.DBD18
E16	DB1.DBD22
E17	DB1.DBB26
E18	DB1.DBB27

Numeric gates block of different data type or NON-consecutive address

There is the possibility to read till to 20 different items in a single request to the PLC . Items can have different Data Type and address.

The block length can be at most 20 items.

Data format	Max block length
BYTE,WORD,BCD,LONG,FLOAT,TIMER,COUNTER	20

Numeric gates that can be grouped as block	Numeric gates that can be grouped as block
E10	DB1.DBB10
E11	DB1.DBB11
DB1.DBW12	E12
DB1.DBW14	E16
E14	DB1.DBX16
E15	DB1.DBD18
E16	E16
E17	E17
E18	DB1.DBB27

28.3 Digital gates

Note: the "Device" field in the GateBuilder must match the Plc Station Address.

Note 1: during PLC configuration, Data Block (DB) on S7-1200 must be defined with the flag "Symbolic access only" deactivated.

Digital gates can be **Input(E)**, **Output(E)**, **DataBase(DB)** , **V Area(V)** or **Merker(M)**.

The address in byte alignment. The position of the byte and of the bit inside the byte must be specified.

Input, Output and V area digital gates address:

* About DB area on S7-1200 see Note 1 at the end of page

Data type	Byte address	.	Bit address	Read Gate	Write Gate	Read Block	Example
EB (Input german mode)	XXXXX (0..99999)	.	X (0..7)	Yes	No	Yes	EB3.1
IB (Input english mode)	XXXXX (0..99999)	.	X (0..7)	Yes	No	Yes	IB3.1
AB (Output german mode)	XXXXX (0..99999)	.	X (0..7)	Yes	Yes	Yes	AB2.4
QB (Output english mode)	XXXXX (0..99999)	.	X (0..7)	Yes	Yes	Yes	QB2.4
MB (Merker german mode)	XXXXX (0..99999)	.	X (0..7)	Yes	Yes	Yes	MB2.4
FB (Merker english mode)	XXXXX (0..99999)	.	X (0..7)	Yes	Yes	Yes	FB2.4
VB (V area) S7-200 1200	XXXXX (0..65535)	.	X (0..7)	Yes	Yes	Yes	VB1.7

DB digital gates address for S7-300,S7-400 and S7-1200:

* About DB area on S7-1200 see Note 1 at the end of page

DB	DB number	.	Data type	Byte address	.	Bit address	Gate read	Gate write	Block read
DB	XXXX (1..8191)	.	DBB (bit)	XXXXX (0..65535)	.	X (0..7)	Yes	Yes	Yes

Digital gates block

To improve speed communications between PC and PLC, use of Blocks sampling **is recommended** . **Input** and **Output** block gates must be compound from gates that have the same type and data format and that have a consecutive address in increasing order (in relation to the data format). Is not necessary that the bit position inside the byte is consecutive.

DataBase block gates must be compound from gates that have the same DB number, the same gate type and data format and that have a consecutive address in increasing order (in relation to the data format). Is not necessary that the bit position inside the byte is consecutive.

The maximum block length is 1600 gates.

Digital gates that CAN be grouped as block	Digital gates that CAN be grouped as block	Digital gates that CAN NOT be grouped as block	Digital gates that CAN NOT be grouped as block
EB10.1	DB1.DBB10.0	EB10.1	DB1.DBB10.0
EB10.2	DB1.DBB10.1	EB10.2	DB1.DBB10.1
EB10.3	DB1.DBB10.2	EB10.3	DB1.DBB10.2
EB10.7	DB1.DBB10.3	EB10.7	DB1.DBB12.3
EB11.3	DB1.DBB10.4	EB14.3	DB1.DBB17.4
EB11.4	DB1.DBB11.0	EB17.4	DB1.DBB19.0
EB12.5	DB1.DBB11.1	EB18.5	DB1.DBB19.1
EB12.6	DB1.DBB11.4	EB22.6	DB1.DBB19.4
EB13.0	DB1.DBB11.5	EB23.0	DB1.DBB19.5

28.4 String gates

Note: the "Device" field in the GateBuilder must match the Plc Station Address.

Note 1: during PLC configuration, Data Block (DB) on S7-1200 must be defined with the flag "Symbolic access only" deactivated.

String gates can be related only and exclusively to STRING[] type variables in the **DataBase(DB)** area for S7-300,S7-400 and S7-1200 PLC family.

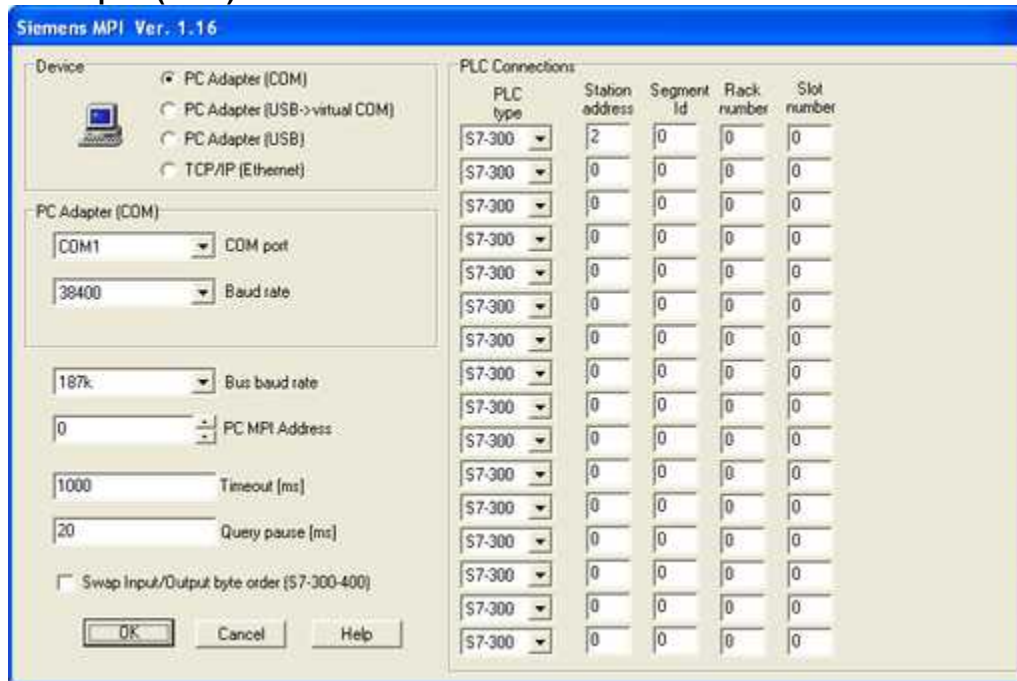
The maximum size specified for the string gate (in Gate Builder) must be equal to the size specified for the relative STRING[] variable inside the PLC.

Address of string gates (for S7-300,S7-400 and S7-1200):

DB	DB number	.	Data type	Data format	Data address	Gate read	Gate write	Block read	Example
DB	XXXXX (1..32767)	.	DB	B (Byte)	XXXXX (0..65535)	Yes	Yes	No	DB2.DBB3

28.5 Configuration

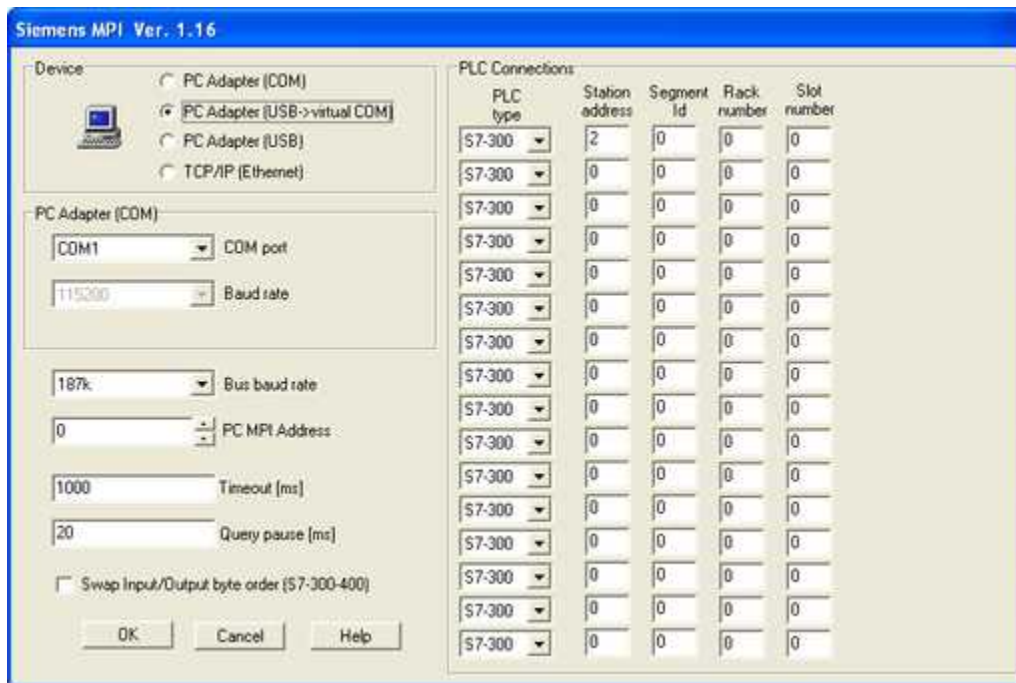
PC adapter (COM) selected



Protocol configuration window.

- **COM port:** PC serial port (COM) to utilize for communication with PLC.
- **Baud rate:** serial port communication speed.
- **Bus baud rate:** MPI bus communication speed.
- **PC MPI Address:** Siemens PC Adapter MPI address.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** wait time between two message requests.
- **Swap Input/Output byte order (S7-300-400):** enable byte low/high swapping in case of Input/output Read/Write Word for S7-300 and S7-400 PLC.
- **Plc type:** can be S7-200, S7-300, S7-400, S7-1200
- **Station address:** station number (must match the "Device" field in the GateBuilder)
- **Segment id:** segment identifier .
- **Rack number:** rack number.
- **Slot number:** slot number.

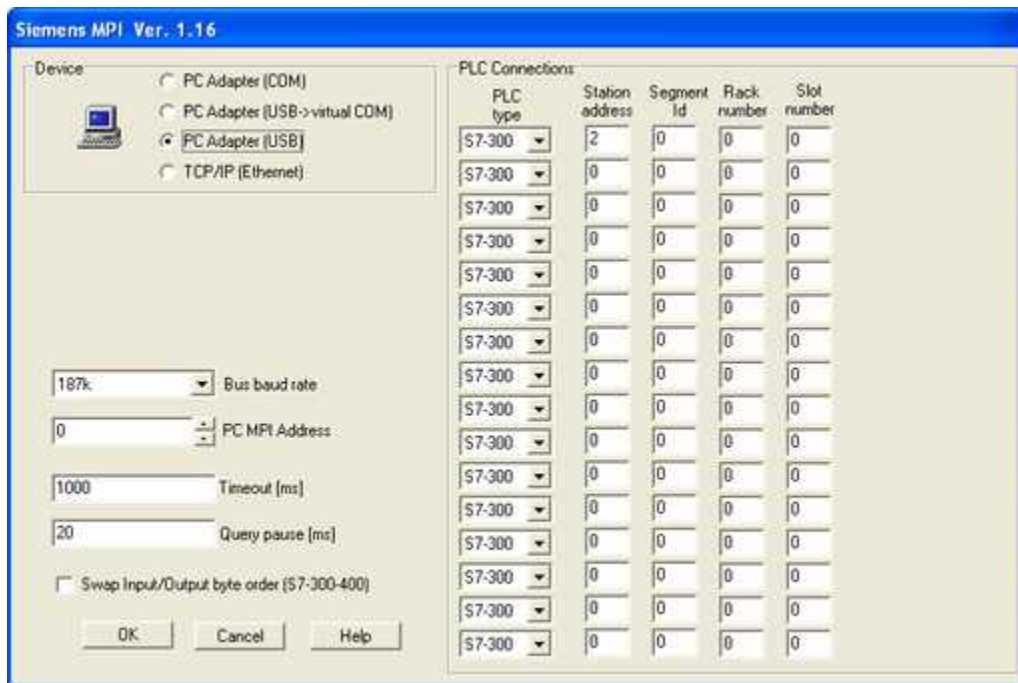
PC adapter (USB -> vrtual COM) selected



Protocol configuration window.

- **COM port:** PC serial port (COM) to utilize for communication with PLC.
- **Bus baud rate:** MPI bus communication speed.
- **PC MPI Address:** Siemens PC Adapter MPI address.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** wait time between two message requests.
- **Swap Input/Output byte order (S7-300-400):** enable byte low/high swapping in case of Input/output Read/Write Word for S7-300 and S7-400 PLC.
- **Plc type:** can be S7-200, S7-300, S7-400, S7-1200
- **Station address:** station number (must match the "Device" field in the GateBuilder)
- **Segment id:** segment identifier .
- **Rack number:** rack number.
- **Slot number:** slot number.

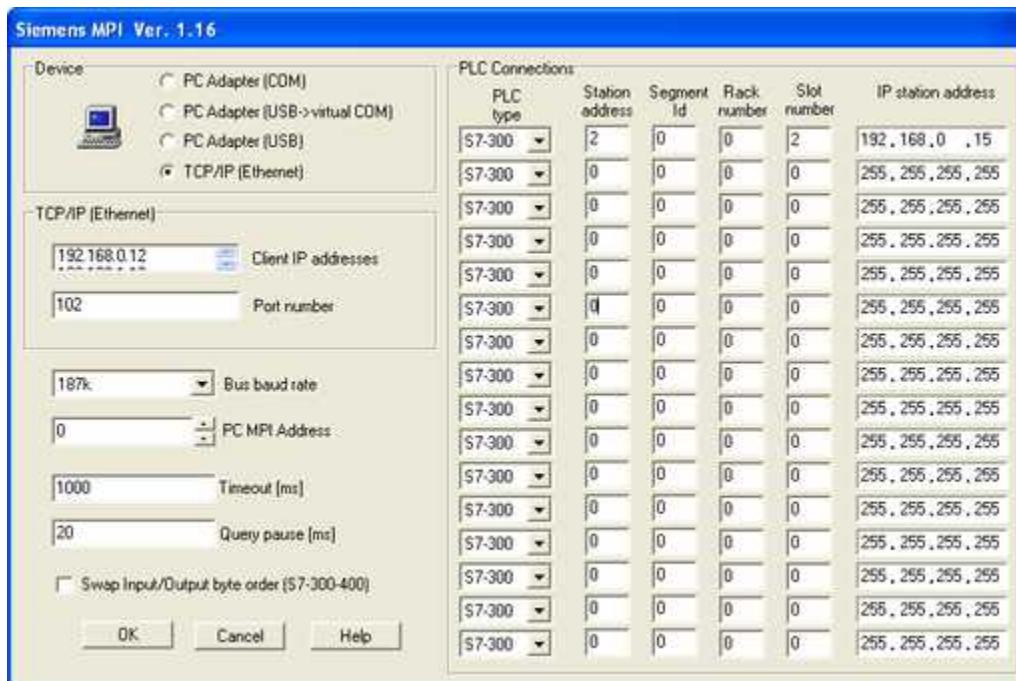
PC adapter (USB) selected



Protocol configuration window.

- **Bus baud rate:** MPI bus communication speed.
- **PC MPI Address:** Siemens PC Adapter MPI address.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query pause [ms]:** wait time between two message requests.
- **Swap Input/Output byte order (S7-300-400):** enable byte low/high swapping in case of Input/output Read/Write Word for S7-300 and S7-400 PLC.
- **Plc type:** can be S7-200, S7-300, S7-400, S7-1200
- **Station address:** station number (must match the "Device" field in the GateBuilder)
- **Segment id:** segment identifier .
- **Rack number:** rack number.
- **Slot number:** slot number.

TCP/IP (Ethernet) selected



Protocol configuration window.

- **Port number:** PC ethernet card port number to utilize for communication with PLC.
- **Bus baud rate:** MPI bus communication speed.
- **PC MPI Address:** PC Adapter MPI address.
- **Timeout [ms]:** timeout (milliseconds) before answer message.
- **Query Pause [ms]:** wait time between two message requests.
- **Swap Input/Output byte order (S7-300-400):** enable byte low/high swapping in case of Input/output Read/Write Word for S7-300 and S7-400 PLC.
- **Plc type:** can be S7-200, S7-300, S7-400, S7-1200
- **Station address:** station number (must match the "Device" field in the GateBuilder)
- **Segment id:** segment identifier .
- **Rack number:** rack number.
- **Slot number:** slot number.
- **IP station address :** IP address associated to the PLC

29 TUTONDO

29.1 Introduction

RS232 communication protocol for Tutondo devices.

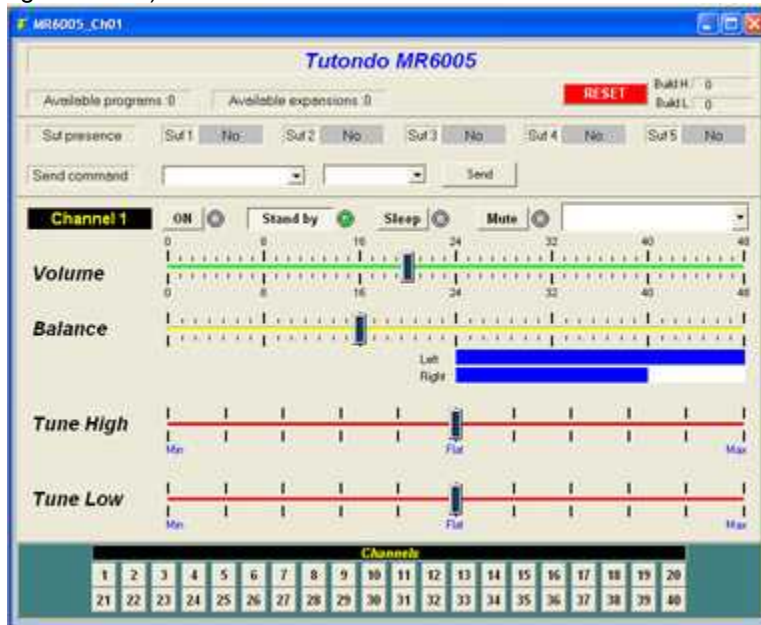
Supported devices:

Tutondo MR6005

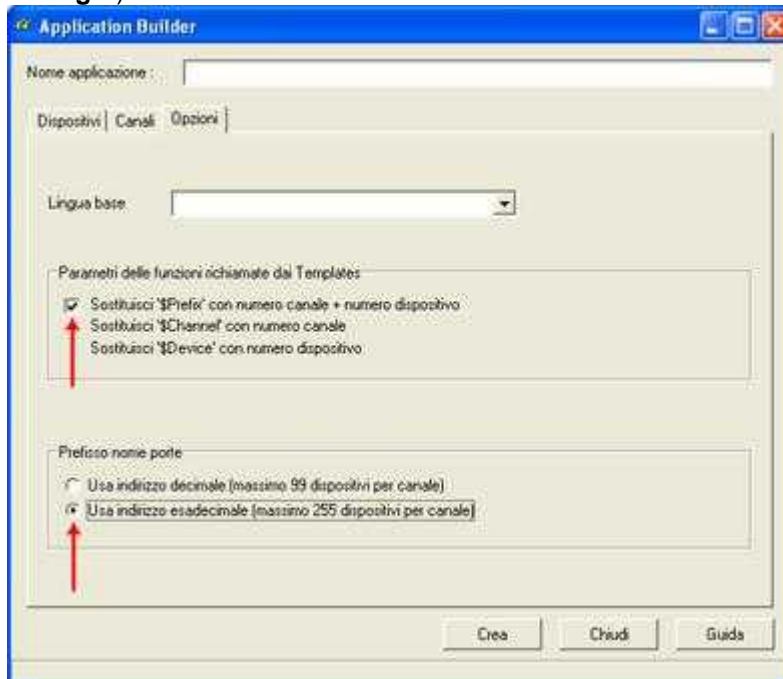
Tutondo MR9005

They are also already available in the Library, in order to allow to build a working application (like in the

figure below) in few seconds.

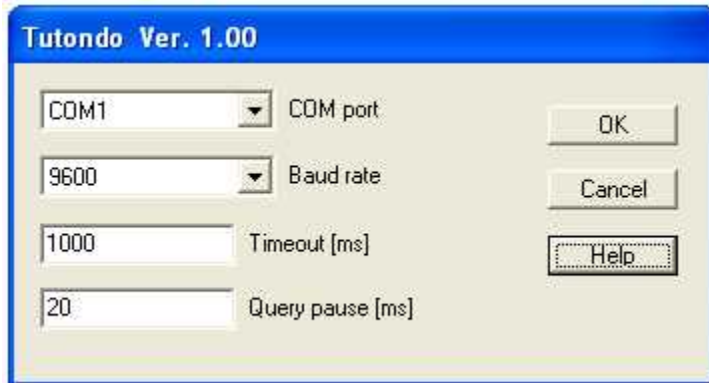


To build applications with Tutondo devices, use **Application Builder** (accessible from **Project Manager**).



Set the "Parameters of functions called from Templates" and the "Gate name prefix" as reported in the picture above.

29.2 Configuration



Protocol configuration window.

- **COM port:** COM port number .
- **Baud rate:** communication speed.
- **Timeout [ms]:** timeout (milliseconds) for a complete answer .
- **Query pause [ms]:** timeout between an answer and the next query.

Note: 'Protocol B' must be selected on the MR6005 and MR9005 physical devices.

30 PANASONIC (MATSUSHITA) MEWTOCOL - COM

30.1 Introduction

This protocol is used for communication with series FP of PANASONIC (Matsushita) PLCs through two different interfaces:

- PC serial interface (COM) : it requires a RS232/RS485 converter.
- PC ethernet interface (LAN) : it uses MEWTOCOL-COM protocol in a TCP/IP frame.

30.2 Numeric gates

The gate address is specified adding the fields Function and Address of the following table.

Function	Description	Address	Gate read	Gate write	Block read
D	DATA REGISTER	XXXXX	Yes	Yes	Yes
D_L	DATA REGISTER (LONG)	XXXXX	Yes	Yes	Yes
D_F	DATA REGISTER (FLOAT)	XXXXX	Yes	Yes	Yes
F	FILE REGISTER	XXXXX	Yes	Yes	Yes
L	LINK DATA REGISTER	XXXXX	Yes	Yes	Yes
IX	IX INDEX REGISTER		Yes	Yes	No
IY	IY INDEX REGISTER		Yes	Yes	No
SDD	DATA REGISTER SET PATTERN	XXXXX	No	Yes	No
SDL	LINK DATA REGISTER SET PATTERN	XXXXX	No	Yes	No
SDF	FILE REGISTER SET	XXXXX	No	Yes	No

	PATTERN				
S	TIMER COUNTER PRESET VALUE	XXXX	Yes	Yes	Yes
S_L	TIMER COUNTER PRESET VALUE (LONG)	XXXX	Yes	Yes	Yes
S_F	TIMER COUNTER PRESET VALUE (FLOAT)	XXXX	Yes	Yes	Yes
K	TIMER COUNTER ELAPSED VALUE	XXXX	Yes	Yes	Yes
R0	SYSTEM REGISTER	XXX	Yes	Yes	Yes
CX	EXTERNAL INPUT RELAY WORD (bit 15...0)	XXXX	Yes	Yes	Yes
CY	EXTERNAL OUTPUT RELAY WORD (bit 15...0)	XXXX	Yes	Yes	Yes
CR	INTERNAL RELAY WORD (bit 15...0)	XXXX	Yes	Yes	Yes
CL	LINK RELAY WORD (bit 15...0)	XXXX	Yes	Yes	Yes
CT	TIMER CONTACT WORD (bit 15...0)	XXXX	Yes	Yes	Yes
CC	COUNTER CONTACT WORD (bit 15...0)	XXXX	Yes	Yes	Yes
SCX	EXTERNAL INPUT RELAY SET DATA PATTERN (bit 15...0)	XXXX	No	Yes	No
SCY	EXTERNAL OUTPUT RELAY SET DATA PATTERN (bit 15...0)	XXXX	No	Yes	No
SCR	INTERNAL RELAY SET DATA PATTERN (bit 15...0)	XXXX	No	Yes	No
SCL	LINK RELAY SET DATA PATTERN (bit 15...0)	XXXX	No	Yes	No

Example: here are some examples of numeric gates:

D00011 : Data Register 00011.

K0004 : Elapsed timer counter 0004.

SDL00015: Set Link Data Register pattern 00015.

CX0012: External input relay word 0012 – bit 15..bit 0.

Note:

A block of numeric gates must be made of gates with the same Function and sequential Address.

Example of valid block	Example of NOT valid block
D00003	D00003
D00004	D00005
D00005	L00012
D00006	D00013
D00007	D00014

Index registers IX,IY can be grouped in a block only if declared sequentially as: IX,IY.

Example of valid block	Example of NOT valid block
IX	IY
IY	IX

30.3 Digital gates

The gate address is specified adding the fields Function, Word address and Bit address of the following table.

Function	Description	Word address	Bit address	Gate read	Gate write	Block read
X	EXTERNAL INPUT RELAY	XXX (decimal value)	X (Hexadecimal value of the bit)	Yes	Yes	Yes
Y	EXTERNAL OUTPUT RELAY	XXX (decimal value)	X (Hexadecimal value of the bit)	Yes	Yes	Yes
R	INTERNAL RELAY	XXX (decimal value)	X (Hexadecimal value of the bit)	Yes	Yes	Yes
L	LINK RELAY	XXX (decimal value)	X (Hexadecimal value of the bit)	Yes	Yes	Yes
T	TIMER CONTACT	XXXX (decimal value)		Yes	Yes	Yes
C	COUNTER CONTACT	XXXX (decimal value)		Yes	Yes	Yes

Example: here are some examples of digital gates address:

X001A : External input relay word 001 – bit 10.

R003C : Internal relay word 003 – bit 12.

T0007 : Timer contact word 0007.

Note:

Digital gates can be grouped in two different types of blocks.

The first type is made of 2...8 gates with any function and address

Digital gates grouped in a block	
X001A	
Y002B	
R003C	
L004D	
L004E	
T0001	
C0002	
C0003	

The second type is made of a number of gates which is equal or multiple of 16; all gates must have the same function and sequential address; in case of function X,Y,R,L, the first gate must have bit address equal.

Example of valid block	Example of NOT valid block
X0010	X0010
X0011	X0011
X0012	X0012
X0013	X0013
X0014	X0014
X0015	X0015
X0016	X0016
X0017	Y0010
X0018	Y0011
X0019	R0010
X001A	X001A
X001B	X001B
X001C	X001C
X001D	X001D
X001E	X001E

X001F	X001F
-------	-------

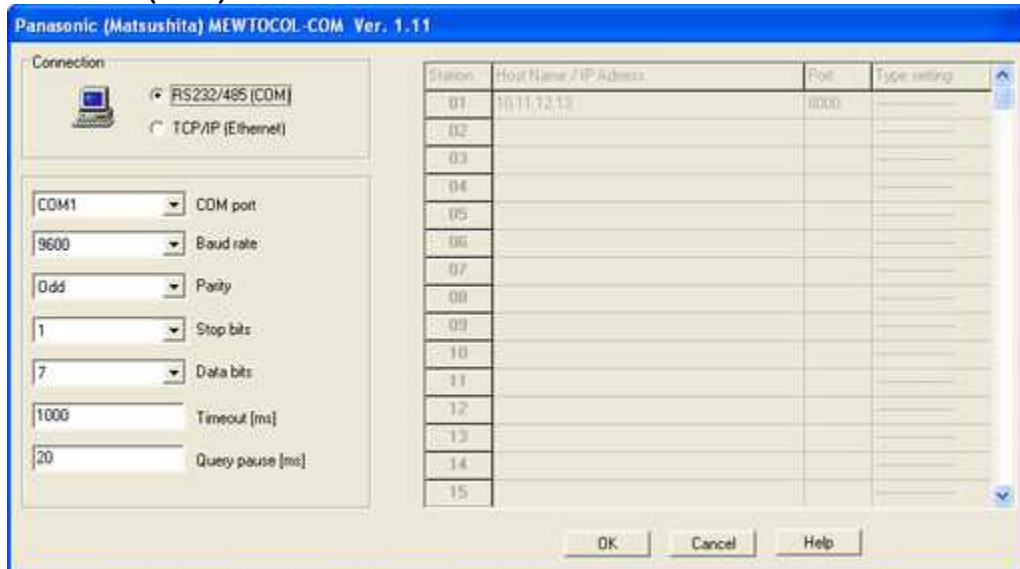
30.4 String gates

The string gate address is specified by the function in the following.

Function	Description	Gate read	Gate write	Block read
RT	READ THE STATUS OF THE PROGRAMMABLE CONTROLLER	Yes	No	No

30.5 Configuration

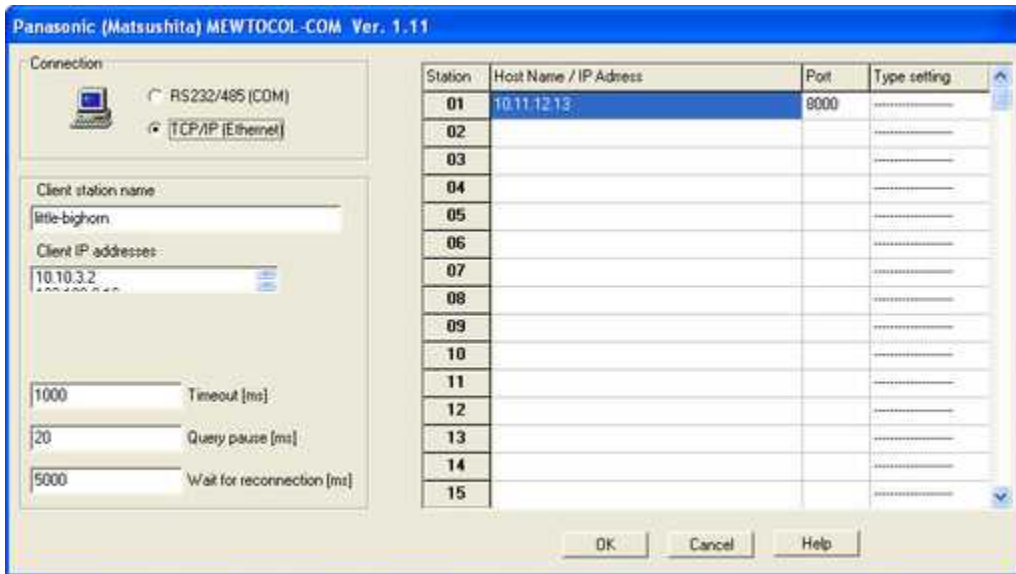
RS232/485 (COM) selected



Protocol configuration window.

- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.
- **Timeout [ms]:** timeout (milliseconds) for a complete answer.
- **Query pause [ms]:** timeout between two request messages.

TCP/IP (Ethernet) selected



Protocol configuration window.

- **Client station name:** PC station name (just only to know)
- **Client IP address:** PC IP address (just only to know)
- **Timeout [ms]:** timeout (milliseconds) for a complete answer.
- **Query pause [ms]:** timeout between two request messages.
- **Wait for reconnection [ms]:** timeout to elapse (milliseconds) after a communication error, before try a new connection.
- **Station:** PLC station number (1..99).
- **Host name / IP Address:** PLC host name or IP address.
- **Port:** PLC destination port.
- **Type setting:** connect or not with FP2-ET-LAN.

31 PPI S7 200 (PPI Adapter)

31.1 Introduction

PPI communication protocol for PLC Siemens **S7-200** series.

One of the following hardware device is needed:

SIEMENS RS232 / PPI Multi - Master Cable - Code 6ES7 901-3CB30-0XA0
Sielco Sistemi IC 3580-PPIVC

In case of using **SIEMENS RS232 / PPI Multi - Master Cable**, it must be configured (with dip-switch) as:

- **PPI / Freeport**
- **Local/DCE**
- **11 Bit**

With this driver is possible to access the following data :

Input
Output
Analog Input
Analog Output
Merker
Special merker
V area

31.2 Numeric gates

Numeric gates can be **Input(I)**, **Output(Q)**, **Merker(M)**, **Special Merker(SM)**, **V area (V)**, **Analog Input (AI)**, **Analog Output(AQ)**.

For all data type, less Analog Input and Analog Output, different data format can be specified: Byte, Word, Double Word, Float.

Note that the data address refers to Byte alignment. That mean :

PLC Memory	Data type "B"	Data type "W"	Data type "D"	Data type "F"
V0	VB0	VW0	VD0	VF0
V1	VB1			
V2	VB2	VW2		
V3	VB3			
V4	VB4	VW4	VD4	VF4
V5	VB5			
V6	VB6	VW6		
V7	VB7			
V8	VB8	VW8	VD8	VF8
V9	VB9			
V10	VB10	VW10		
V11	VB11			

Addresses of numeric gates:

Description	Type	Format	Address	Gate read	Gate write	Block read
V BYTE	V	B	0...99999	Yes	Yes	Yes
V WORD	V	W	0...99999	Yes	Yes	Yes
V DOUBLE WORD	V	D	0...99999	Yes	Yes	Yes
V FLOAT	V	F	0...99999	Yes	Yes	Yes
INPUT BYTE	I	B	0...99999	Yes	No	Yes
INPUT WORD	I	W	0...99999	Yes	No	Yes
INPUT DOUBLE WORD	I	D	0...99999	Yes	No	Yes
INPUT FLOAT	I	F	0...99999	Yes	No	Yes
OUTPUT BYTE	Q	B	0...99999	Yes	Yes	Yes
OUTPUT WORD	Q	W	0...99999	Yes	Yes	Yes
OUTPUT DOUBLE WORD	Q	D	0...99999	Yes	Yes	Yes
OUTPUT FLOAT	Q	F	0...99999	Yes	Yes	Yes
MERKER BYTE	M	B	0...99999	Yes	Yes	Yes
MERKER WORD	M	W	0...99999	Yes	Yes	Yes
MERKER DOWBLE WORD	M	D	0...99999	Yes	Yes	Yes
MERKER FLOAT	M	F	0...99999	Yes	Yes	Yes

SPECIAL MERKER BYTE	SM	B	0...99999	Yes	Yes	Yes
SPECIAL MERKER WORD	SM	W	0...99999	Yes	Yes	Yes
SPECIAL MERKER DOWBLE WORD	SM	D	0...99999	Yes	Yes	Yes
SPECIAL MERKER FLOAT	SM	F	0...99999	Yes	Yes	Yes
ANALOG INPUT WORD	AI	W	0...99999	Yes	Yes	Yes
ANALOG OUTPUT WORD	AQ	W	0...99999	Yes	Yes	Yes

Example:

MB12:Merker Byte 12

MW5 :Merker Word 5

MD11:Merker Double word 11

Numeric gates block

To improve speed communications between PC and PLC, use of Blocks sampling **is recommended**. Gates block must be compound from gates that have the same Data Type but can have different Data Format and that have a consecutive address in increasing order (in relation to the data format).

The block length is related to the gates data format grouped in the block.

Data format	Max block length
BYTE	200
WORD	100
LONG	50
FLOAT	50

Numeric gates that can be grouped as block	Numeric gates that can be grouped as block	Numeric gates that CAN NOT be grouped as block	Numeric gates that CAN NOT be grouped as block
IB0	VW0	IB0	VW0
IB1	VW2	IB3	VW3
IB2	VD4	IB5	VW5
IB3	VD8	IW4	VW8
IW4	VF12	IW8	VW7
IB6	VB16	IB10	VD8
IB7	VB17	IB12	QB10

31.3 Digital gates

Digital gates can be **Input(I)**, **Output(Q)**, **V area(V)**, **Merker (M)** o **Special Merker (SM)**.

The address in byte alignment. The position of the byte and of the bit inside the byte must be specified.

Addresses of digital gates:

Description	Type	Byte address	Bit address	Gate read	Gate write	Block read
V	VB	0...99999	0..7	Yes	Yes	Yes
INPUT	IB	0...99999	0..7	Yes	No	Yes
OUTPUT	QB	0...99999	0..7	Yes	Yes	Yes
MERKER	MB	0...99999	0..7	Yes	Yes	Yes

SPECIAL MERKER	SMB	0...99999	0..7	Yes	Yes	Yes
----------------	-----	-----------	------	-----	-----	-----

Example:

IB13.7 : Input byte 13 – Bit 7

QB20.3 : Output byte 20 – Bit 3

Digital gates block

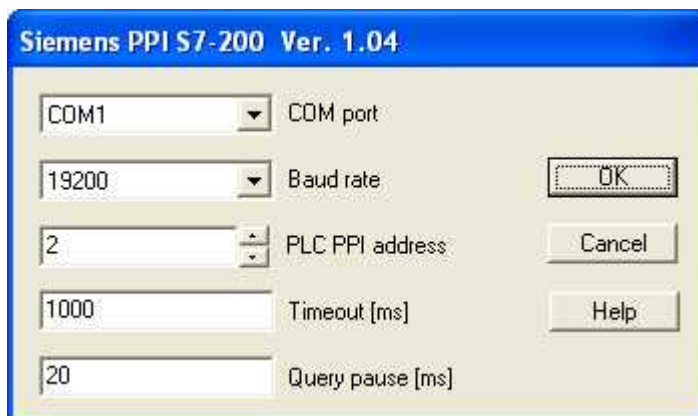
To improve speed communications between PC and PLC, use of Blocks sampling **is recommended**. Block gates must be compound from gates that have the same type and data format and that have a consecutive address in increasing order (in relation to the data format). Is not necessary that the bit position inside the byte is consecutive.

The maximum block length is 1600 gates.

Digital gates that can be grouped as block	Digital gates that can be grouped as block	Digital gates that CAN NOT be grouped as block	Digital gates that CAN NOT be grouped as block
IB0.0	VB0.3	IB0.1	VB0.0
IB0.1	VB0.4	IB3.3	VB3.0
IB0.5	VB0.5	IB5.2	VB5.0
IB1.1	VB1.3	IB5.3	VB8.0
IB1.2	VB1.4	IB5.4	VB7.4
IB1.7	VB1.6	IB5.5	VB8.3
IB2.0	VB1.7	IB6.0	VB8.4

31.4 String gates

String gates are not allowed in this protocol.

31.5 Configuration

Protocol configuration window.

- **COM port:** PC serial port (COM) to utilize for communication with PLC.
- **Baud rate:** serial port communication speed..
- **PLC PPI address:** PLC address.
- **Timeout [ms]:** timeout (milliseconds) for a complete answer.

- **Query pause [ms]:** wait time between two message requests.

32 Raw ASCII Output

32.1 Introduction

ASCII format serial communication protocol.
Whit this protocol is possible to send a string of ASCII characters on serial Channel (COM).

32.2 Numeric gates

Numeric gates address are not allowed in this protocol.

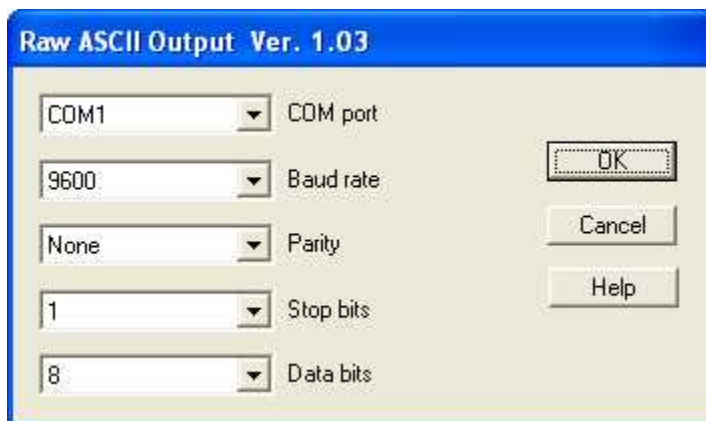
32.3 Digital gates

Digital gates address are not allowed in this protocol.

32.4 String gates

The string gates Text will be sent on the serial channel.
The string gate must be defined as "**Sample Never**" and "**Write enabled**"

32.5 Configuration



Protocol configuration window.

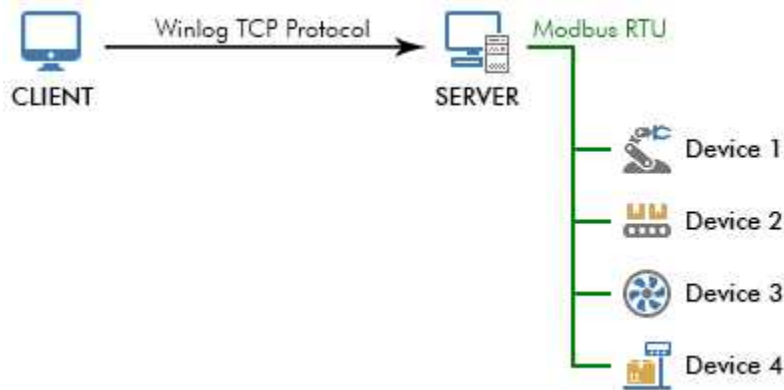
- **COM port:** serial port name.
- **Baud rate:** communication speed.
- **Parity:** parity.
- **Stop bits:** stop bits.
- **Data bits:** number of bits.

33 Winlog TCP Protocol

33.1 Introduction

This protocol allows communication between two or more supervisor software stations. It is possible to sample gates directly from one or more server stations as if they were devices.

Stations can work simultaneously as client and as server.



Architecture example: a client station communicate with a server station to access devices data

Protocol driver provides two working modes:

- **standard** mode (predefined)
- **legacy** mode (not recommended for new projects)

You can choose driver working mode in the configuration dialog.

In standard mode, for each client gate you can define which is the server gate to read and write. Instead, in legacy mode, client can read (or write) only gates that have the same name ($Id + n/d$) in the server. This may be a constrain for some applications; for example from a client you cannot access gates from two servers that have the same gate list.

Furthermore standard mode provides more features. In addition to the value of a remote gate, it allows to access additional data, like state of communication with the real device. Moreover standard mode provides way to read compound gates and events from remote server and to store their value in numeric or digital gates of the client station.

Standard mode transfers the engineering value of numeric gates and compound gates. So, during definition of client gates, do not set conversion between measured value and engineering value. In legacy mode, client gate must have the same conversion as related server gate.

Use legacy mode only for compatibility issues in projects developed with version 2.08 or below of the SCADA software.

When a client is connected to more servers using the same communication channel, *Device* field of gate defines from which server the value has to be sampled. In configuration dialog, you can define association between *Device* (a number from 0 to 255) and the address of the server.

On client station and on server station, it is recommended to use the same version of the software.

You need to enable server station to listen for client connections. To do it, go to project *Configuration | Options | TCP* and check the box *Run TCP Server*.

33.2 Numeric gates

In legacy mode, field *Address* of gates is ignored. Strictly, protocol driver, associates each local numeric gate to the remote numeric gate that has the same name, namely the same *Id* and the same *nId*.

In standard mode, you can associate a local numeric gate to any remote gate, be it numeric, digital, compound or event. This allows to read (or write) value of remote gate or to retrieve information about his state.

If a local numeric gate is associated to a remote numeric or digital gate and you change his value, modification will be transferred to the server and the value of the remote gate will be changed too.

Numeric gates address in standard mode

Address field of numeric gates must comply to the following format:

```
type:id,nId
type:id,nId:var_id
```

where

- *type*, *id*, *nId* are type (numeric, digital, compound or event), field *Id* and field *nId* of remote gate that you want to associate to the local gate. *type* must be one of the following: NUM, DIG, CMP, EVN to indicate type numeric, digital, compound or event, respectively.
- *var_id* is the identification of which information has to be retrieved. Can be one of the following:

var_id	type	Description
value	NUM	Reading: local numeric gate ? value of remote numeric gate Writing: value of remote numeric gate ? local numeric gate
	DIG	Reading: local numeric gate ? value of remote digital gate (0 or 1) Writing: value of remote digital gate ? 1 if local numeric gate is different from 0
	CMP	Reading: local numeric gate ? value of remote compound gate Writing: gate cannot be written
	EVN	Reading: local numeric gate ? 1 if remote event is active Writing: gate cannot be written
commStatus	NUM, DIG, STR, CMP	Reading: local numeric gate ? 1 if remote gate has communication failure (reading error or writing error) Writing: gate cannot be written
commRxStatus	NUM, DIG, STR	Reading: local numeric gate ? 1 if remote gate has communication failure (reading error) Writing: gate cannot be written
commTxStatus	NUM, DIG, STR, CMP	Reading: local numeric gate ? 1 if remote gate has communication failure (writing error) Writing: gate cannot be written

var_id	type	Description
excluded	EVN	Reading: local numeric gate ? 1 if remote event is excluded Writing: gate cannot be written
acked	EVN	Reading: local numeric gate ? 1 if remote event is acknowledged Writing: gate cannot be written
significant	EVN	Reading: local numeric gate ? 1 if remote event is significant (active or not acknowledged) Writing: gate cannot be written
raw	EVN	Reading: local numeric gate ? all flags or remote event gate bit 0: event is active bit 1: event is excluded bit 2: event is acknowledged bit 3: event is significant (active or not acknowledged) Writing: gate cannot be written

var_id is optional. If you omit it (and the separator character) is like you specify *value*.

Furthermore it is possible to leave blank the field *Address* of the gate. In this case protocol driver associates the local numeric gate to remote numeric gate that has the same name.

Examples

NUM:temperature,1	Value of remote numeric gate <i>temperature,1</i>
NUM:temperature,1:value	Value of remote numeric gate <i>temperature,1</i>
NUM:temperature,1:commRxStatus	State of communication (reading error) of remote numeric gate <i>temperature,1</i>
DIG:pumpState,3	Value of remote digital gate <i>pumpState,3</i>
CMP:energyPerHour,0	Value of remote compound gate <i>energyPerHour,0</i>
EVN:doorOpen,1	State (active or not active) of the remote event
<i>doorOpen,1</i>	
EVN:currentI1Overload,0:acked	Acknowledgement of the remote event
<i>currentI1Overload,0</i>	

Blocks of numeric gates

Blocks of numeric gates are supported and suggested. Maximum block length is 512 gates. Gates must not observe any particular order and in the remote station can have any type.

33.3 Digital gates

In legacy mode, field *Address* of gates is ignored. Strictly, protocol driver, associates each local digital gate to the remote digital gate that has the same name, namely the same *Id* and the same *nId*.

In standard mode, you can associate a local digital gate to any remote gate, be it numeric, digital, compound or event. This allows to read (or write) value of remote gate or to retrieve information about his state.

If a local digital gate is associated to a remote digital or numeric gate and you change his value, modification will be transferred to the server and the value of the remote gate will be changed too.

Digital gates address in standard mode

Address field of digital gates must comply to the following format:

```
type:id,nId
type:id,nId:var_id
```

where

- *type, id, nId* are type (digital, numeric, compound or event), field *Id* and field *nId* of remote gate that you want to associate to the local gate. *type* must be one of the following: DIG, NUM, CMP, EVN to indicate type digital, numeric, compound or event, respectively.
- *var_id* is the identification of which information has to be retrieved. Can be one of the following:

var_id	type	Description
value	DIG	Reading: local digital gate ? value of remote digital gate Writing: value of remote digital gate ? local digital gate
	NUM	Reading: local digital gate ? 1 if remote numeric gate is different from 0 Writing: value of remote numeric gate ? local digital gate (0 or 1)
	CMP	Reading: local digital gate ? 1 if remote compound gate is different from 0 Writing: gate cannot be written
	EVN	Reading: local digital gate ? 1 if remote event is active Writing: gate cannot be written
commStatus	DIG, NUM, STR, CMP	Reading: local digital gate ? 1 if remote gate has communication failure (reading error or writing error) Writing: gate cannot be written
commRxStatus	DIG, NUM, STR	Reading: local digital gate ? 1 if remote gate has communication failure (reading error) Writing: gate cannot be written
commTxStatus	DIG, NUM, STR, CMP	Reading: local digital gate ? 1 if remote gate has communication failure (writing error) Writing: gate cannot be written
excluded	EVN	Reading: local digital gate ? 1 if remote event is excluded Writing: gate cannot be written
acked	EVN	Reading: local digital gate ? 1 if remote event is acknowledged Writing: gate cannot be written
significant	EVN	Reading: local digital gate ? 1 if remote event is significant (active or not acknowledged) Writing: gate cannot be written

var_id is optional. If you omit it (and the separator character) is like you specify *value*. Furthermore it is possible to leave blank the field *Address* of the gate. In this case protocol driver associates the local digital gate to remote digital gate that has the same name.

Examples

DIG:pumpState,3	Value of remote digital gate <i>pumpState,3</i>
DIG:pumpState,3:value	Value of remote digital gate <i>pumpState,3</i>
NUM:itemsProduced,1	Value of remote numeric gate <i>itemsProduced,1</i> is
different from 0	
DIG:pumpState,3:commTxStatus	State of communication (reading error) of remote
digital gate <i>pumpState,3</i>	
EVN:doorOpen,1	State (active or not active) of the remote event
<i>doorOpen,1</i>	
EVN:currentI1Overload,0:acked	Acknowledgement of the remote event
<i>currentI1Overload,0</i>	

Blocks of digital gates

Blocks of digital gates are supported and suggested. Maximum block length is 512 gates. Gates must not observe any particular order and in the remote station can have any type.

33.4 String gates

In legacy mode, field *Address* of gates is ignored. Strictly, protocol driver, associates each local string gate to the remote string gate that has the same name, namely the same *Id* and the same *nId*.

In standard mode, you can associate a local string gate to any remote string gate, this allows to read and write value of remote gate. It is not possible to associate string gates to remote gates of different type.

String gates address in standard mode

Address field of numeric gates must comply to the following format:

```
STR:id,nId
STR:id,nId:value
```

where

- *id, nId* are field *Id* and field *nId* of remote gate that you want to associate to the local gate.

It is possible to leave blank the field *Address* of the gate. In this case protocol driver associates the local string gate to remote string gate that has the same name.

Examples

STR:errorString,0	Value of remote string gate <i>errorString,0</i>
STR:programName,1:value	Value or remote string gate <i>programName,1</i>

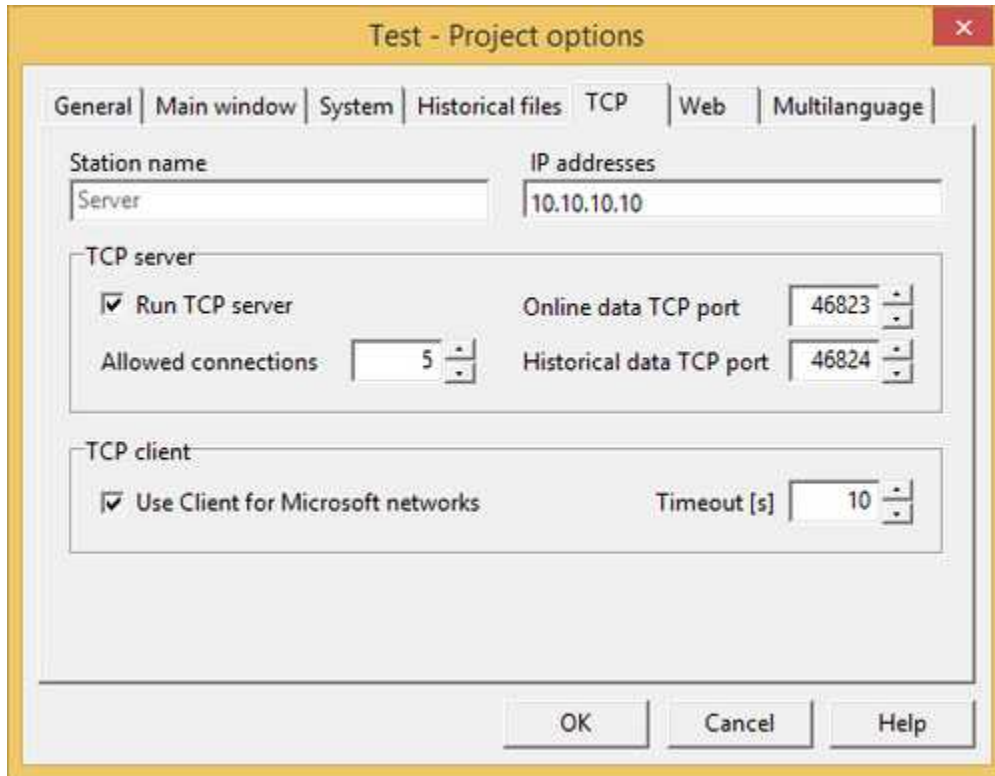
Blocks of string gates

Blocks of string gates are supported and suggested. Maximum block length is 512 gates. Gates must not observe any particular order.

33.5 Configuration

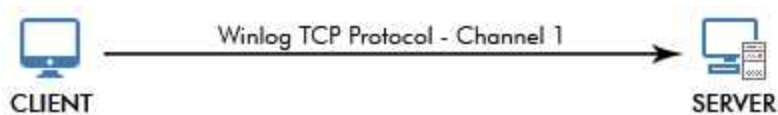
Server station

On the server station, under project *Configuration* | *Options* | *TCP*, check the box *Run TCP server*.



Project configuration dialog on the server station

Client station with one connection per channel



Client station establishes connection with only one server station. All gates of the channel are read and written from the server station. The field *Device* of the gates is ignored.

Driver configuration dialog on the client station

Connection type: if the client communicates only with one server you have to choose *One connection for all devices*.

Server IP address or hostname: IP address or network name of the server station.

TCP port: TCP port of the server. It must be the same as the value *Online data TCP port* of the server project configuration.

Connection timeout: maximum time (in ms) allowed to the client to establish the connection with the server.

Re-connection delay: delay (in ms) between a failed connection and the following attempt.

Transfer timeout: maximum time (in ms) allowed to receive the response message.

Query pause: delay (in ms) between an answer and the following request.

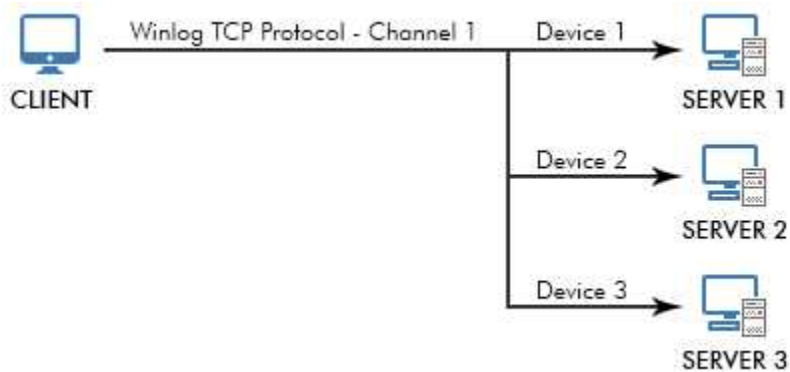
Keep connection active: to avoid waste of resources, server station closes inactive connections. Check this box, to maintain live connections even with slow sampling rate.

Legacy mode: this option setup driver to operate in *legacy mode*. Reffer to introduction to know differences between *standard mode* and *legacy mode*.

Inherit communication status: if you check this box, local gate "inherit" communication status of the related remote gate.

Create communication error log: if set, during runtime, protocol driver will create an error log file. To display error log press *Errors* button (during runtime as well). Error log is useful to debug projects helping you to find causes of communication errors.

Client station with a connection for each device



Client station establishes a connection for each device belonging to the channel. For each device you need to specify the server from which the driver will read and write related gates. Please use the table in the right side of the dialog to associate each *Device* (a number from 1 to 255, as specified in the gate configuration) to the server; insert his IP address (or network name) and optionally listening TCP port.

So, all gates belonging to *Device 1* of the channel will be sampled from the server inserted in the first row of the table, and so on up to 255 device per channel. In this way a server act as a device.

Device	IP Address or hostname
1	192.168.0.110:50003
2	localhost
3	
4	server.mydomain.com:17004
5	
6	
7	
8	
9	
10	

Driver configuration dialog on the client station

Connection type: choose *One connection for each device* if you want the client communicates with more servers.

Server IP address or hostname: IP address (or network name) associates to each device. You can also indicate the TCP port, if left unspecified driver will use that inserted in the field *TCP port*.

Other settings are the same as described for the case with only one connection.