Specifiche generali protocollo MODBUS-RTU Rev. 10 (inglese)

Written: U. T. ing. M. Bandini / A. Sangiorgi _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Verified: A. Q. sig. F. Adinolfi _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Approved: D. T. ing. P. Amato _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Date: 17/09/04

Status : Final

REVISIONS

REV.	VER	DATE	CHANGES	WRITTEN	VERIFIED	APPROVED
2		17/07/98	Adding to pages 7	Goffredo	Donati	Cantobelli
3		01/12/98	Multivariable Read	Canini	Goffredo	Cantobelli
4		18/02/00	Errata corrige	Zoletto	Donati	Amato
5		30/05/00	New memory map function 06	Zoletto	Donati	Amato
6		21/11/00	Functions 01 06	Zoletto	Donati	Amato
7		07/12/01	New command for Smart Più	Bandini	Adinolfi	Amato
8		17/12/02	New ID for Smart 96 Più; Voltage and Current ThdF for Smart Più and Smart96 Più. Different management of the requests with memory overflow (from V3.03)	Bandini Sangiorgi	Adinolfi	Amato
9		28/11/03	New ID and Apparent energy measurement for Smart (96) Più V3.11(KVAh)	Bandini	Adinolfi	Amato
10		17/09/04	For Smart(96) Più from version V3.12. Update new instruments ID and new range for CT ratio	Sangiorgi	Adinolfi	Amato

Contents

1. Communication frame in RTU mode (Remote Terminal Unit)

A modbus frame is composed of:

T1 T2 T3	Address (8 bits)	Function (8 bits)	Data (N x 8 bits)	CRC (16 bits)	T1 T2 T3

in which

a) the Address field contains the address of the Slave to which the message is sent

b) the Function field contains the code of the function that must be carried out by the Slave

c) the Data field contains the information needed by the Slave to carry out a specific function or contains data collected from the Slave in response to a question

d) the CRC field allows both the Master and the Slave to check a message in order to detect any errors in transmission. Sometimes, due to electrical "noise" or other interference, a message may be changed during the transmission from one unit to another. The error check ensures that neither the Master nor the Slave react to messages that have been haltered

e) the T1 T2 T3 sequence represents the time that separates one frame from another, and corresponds to at least 3 and a half characters: during this period no one is allowed to talk on the bus, to let the instruments detect that a frame is over and another one is starting

In RTU mode, the synchronisation of the frame can be maintained only by simulating a synchronous message. The receiving device, a MachSmart for example, measures the time that separates the reception of one character and the reception of the subsequent one (for example, between address and function). If this time is longer than the time needed to transmit three and a half characters, then the message is considered lost and the next character arriving is considered to be an address, in other words the beginning of a new frame.

2. Activation of Modbus protocol and available functions

To activate the MODBUS protocol in the MACH SMART, "**1**" must be selected in the last field of the Setup indicated by "**PROT**" (0 = DUCBUS , 1 = MODBUS).

The available MODBUS functions are reported in the following table:

Modbus Functions	Action performed in the analyser
01 = READ OUTPUT STATUS	Read data relative to the output status
03 = READ HOLDING REGISTERS	Reads data relative to the Measurements and the Setup
05 = FORCE SINGLE COIL	Set the output state
06 = PRESET SINGLE REGISTER	Sets Setup parameters
07 = READ EXCEPTION STATUS	Reads Instrument status
17 = REPORT SLAVE ID	Reads the identification of the instrument type

IMPORTANT NOTE REGARDING FUNCTIONS 3 AND 6 DESCRIBED BELOW:

Whenever the user employs commercial programs for reading the data from the Slaves, and these programs are already designed for handling the Modbus protocol, it is necessary to use the addresses of the storage locations indicated in the first left-hand column of the tables relative to functions 3 and 6 illustrated below.

Instead, whenever the user writes his own program to read the data it is necessary to decrease by one the addresses reported.

For example: the Master wishes to read from MachSmart with address 3 the value of the three-phase equivalent current, available at memory address 18 (= 12Hex).

The communication frame of the Master to the Slave, with hexadecimal values, will be the following:

03	03	00	11	00	02	95	EC

where:

03 = address of Mach Smart (Slave) no. 03

03 = function requested by the Master, in this case measure reading request

00 11 = address, reduced by one unit (12H - 1H = 11H), of the storage location to be read and containing the value of the three-phase equivalent current

00 02 = number of registers to be read beginning with address 11H

95 EC = CRC

The Slave response frame to the Master in hexadecimal values will be the following:

03	03	04	00	00	01	E0	D9	EB

in which:

03 = address of Mach Smart (Slave) no. 03

03 = function requested by the Master, in this case measure reading request

04 = number of data bytes following in the frame

00 00 01 E0 = hex value of the three-phase equivalent current (01E0H = 480 hundredths of A = 4.80 A)

D9 EB = CRC

2.1 Function 01 : "READ OUTPUT STATUS"

This function is used to read the status of the Smart 96 or Smart Più output; the output is treated like a coil, following Modbus standard.

The request and answer frame are described below:

Read request (master):

Addr	Func	Coil Addr Start H	Coil Addr Start L	Number of Coils H	Number of Coils L	CRC	CRC
1Fh	01h	00h	00h	00h	08h	3Eh	72h

In the example above the "read output function" **Func = 01** is sent to the slave with address **Addr = 1Fh**, starting from register **Coil Addr Start = 0000h** (compulsory) (5th bit set to one) for **Number of coils = 08h** (compulsory) adjacent coils. The frame is closed with **CRC=3E72h**.

Reply (slave):

Addr	Func	Byte Count	Data Coil Byte 0H	CRC	CRC
1Fh	01h	01h	00h	57h	A0h

The answer fields, as described above, are the slave address **Addr = 1Fh**, the function executed **Func = 01**, the number of data bytes following **Byte Count = 01h**, the coil value **00h**: the 5th bit is related to the Smart 96 output, all the others are forced to 0 otherwise for the Smart Più are bit 0 and bit 1 (1st and 2nd bit respectively for output 1 and output 2), so in this example the output status is 0 (open). The frames ends with the CRC.

This function is available in the Smart 96 from version 1.01 onwards and for model 2P of Smart Più.

2.2 Function 03 : "READ HOLDING REGISTERS"

This function reads one or more memory adjacent locations, each one being 2-word sized. It is possible to read up to 12 or 14[i] consecutive measures[*]. Moreover Smart Più can read a Mix of measures previously configured (see 2.4).

Below are described the read request format (from master to slave) and the reply format (from slave to master).

[i] Smart Più: 14 measures

[*] Mach Smart with firmware versions earlier than 1.07 allow only one value (2 words) per enquiry to be read.

Read request (master):

Addr	Func	Data Start Register H	Data Start Register L	Data # of Regs H	Data # Regs L	CRC	CRC
1Fh	03h	00h	11h	00h	08h	17h	B7h

In the example above, the master sends the 'read function' **Func = 03** to the slave with address **Addr = 1Fh**, starting from base register address **Data Start Register = 0011h** for **Data Regs = 08h** consecutive registers. So the command reads all registers from address **0011h** to **0018h**. The **CRC = 17B7h** closes the data stream.
(Note: the physical address is always obtained from measure address reduced of 1 unit, see note above).

Reply (slave):

Addr	Func	Byte count	Data Out Reg 0012 H	Data Out Reg 0012 L	……	Data Out Reg 0018H	Data Out Reg 0018L	CRC	CRC
1Fh	03h	10h	10h	EFh	……..	3Bh	40h	xxh	yyh

The table above shows the fields in the MACH SMART reply, which are :
-Addressed Slave **Addr = 1Fh**
-Function code request **Func = 03**
-Number of data byte following **Byte Count = 10h**
-Data byte fields requested by the master
-CRC closes the reply data stream

There are **three particular cases** that can happen using this command; the first is related to the quantity of requested memory, the second is related to the beginning of the requested segment and the last is related to the quantity of the requested words.
In particular, if the quantity of the requested bytes is greater than the Smart's memory extension, the instrument will answer an "INVALID DATA" for the not available values; for example, if are requested 20 byte from the last fourth valid address, a part of the request overflows in the non available memory. The exceeded bytes will be filled with the value FFh[ii], indicating a non managed value for those memory cells.
The second particular case is related to a request starting from a non valid address, when the request starts from an address not present in the following table. In this case the instrument will answer with an exception "02 ILLEGAL DATA ADDRESS" (see 3).
The last particular case is the request of a number of word greater than the maximum for the instrument: in this case the instrument will answer with an exception "02 ILLEGAL DATA address" (see 3).

[ii] SmartPiù from V. 3.03 on; in the V. 3.00 the slave answer at this request was an exception 02

Memory map

The following table indicates the correspondence between the address of the location, the number of accessible words beginning with that address, the description of the measurement value, the unit of measurement of the measurement value and the binary format.

Add.	Word	Measurement description	Unit	Format
0002	2	Frequency	Tenths of Hz	Unsigned Long
0004	2	Three-phase Equivalent Voltage	Volt	Unsigned Long
0006	2	Line Voltage (line 1 - line 2)	Volt	Unsigned Long
0008	2	Line Voltage (line 2 - line 3)	Volt	Unsigned Long
0010	2	Line Voltage (line 3 - line 1)	Volt	Unsigned Long
0012	2	Voltage between Phase and Neutral line 1	Volt	Unsigned Long
0014	2	Voltage between Phase and Neutral line 2	Volt	Unsigned Long
0016	2	Voltage between Phase and Neutral line 3	Volt	Unsigned Long
0018	2	Three-phase Equivalent Current	Hundredths of A	Unsigned Long
0020	2	Current Line 1	Hundredths of A	Unsigned Long
0022	2	Current Line 2	Hundredths of A	Unsigned Long
0024	2	Current Line 3	Hundredths of A	Unsigned Long
0026	2	Three-phase equivalent power factor*	Hundredths	Signed/Unsigned Long[iii]
0028	2	Power factor line 1*	Hundredths	Signed Long
0030	2	Power factor line 2*	Hundredths	Signed Long
0032	2	Power factor line 3*	Hundredths	Signed Long
0034	2	Three-phase equivalent active power	Watt	Unsigned Long
0036	2	Average three-phase equivalent active power	Watt	Unsigned Long
0038	2	Maximum three-phase equivalent active power	Watt	Unsigned Long
0040	2	Active power line 1	Watt	Unsigned Long
0042	2	Active power line 2	Watt	Unsigned Long
0044	2	Active power line 3	Watt	Unsigned Long
0046	2	Average active power line 1	Watt	Unsigned Long
0048	2	Average active power line 2	Watt	Unsigned Long
0050	2	Average active power line 3	Watt	Unsigned Long
0052	2	Maximum active power line 1	Watt	Unsigned Long
0054	2	Maximum active power line 2	Watt	Unsigned Long
0056	2	Maximum active power line 3	Watt	Unsigned Long
0058	2	Three-phase equivalent apparent power	VA	Unsigned Long

[iii] For Smart Più this value is a signed long

0060	2	Average three-phase equivalent apparent power	VA	Unsigned Long
0062	2	Maximum three-phase equivalent apparent power	VA	Unsigned Long
0064	2	Apparent power line 1	VA	Unsigned Long
0066	2	Apparent power line 2	VA	Unsigned Long
0068	2	Apparent power line 3	VA	Unsigned Long
0070	2	Average apparent power line 1	VA	Unsigned Long
0072	2	Average apparent power line 2	VA	Unsigned Long
0074	2	Average apparent power line 3	VA	Unsigned Long
0076	2	Maximum apparent power line 1	VA	Unsigned Long
0078	2	Maximum apparent power line 2	VA	Unsigned Long
0080	2	Maximum apparent power line 3	VA	Unsigned Long
0082	2	Three-phase equivalent reactive power	VAr	Unsigned Long
0084	2	Average three-phase equivalent reactive power	VAr	Unsigned Long
0086	2	Maximum three-phase equivalent reactive power	VAr	Unsigned Long
0088	2	Reactive power line 1	VAr	Unsigned Long
0090	2	Reactive power line 2	VAr	Unsigned Long
0092	2	Reactive power line 3	VAr	Unsigned Long
0094	2	Average reactive power line 1	VAr	Unsigned Long
0096	2	Average reactive power line 2	VAr	Unsigned Long
0098	2	Average reactive power line 3	VAr	Unsigned Long
0100	2	Maximum reactive power line 1	VAr	Unsigned Long
0102	2	Maximum reactive power line 2	VAr	Unsigned Long
0104	2	Maximum reactive power line 3	VAr	Unsigned Long
0106	2	Three-phase equivalent active energy	Tens of Wh	Unsigned Long
0108	2	Active energy line 1	Tens of Wh	Unsigned Long
0110	2	Active energy line 2	Tens of Wh	Unsigned Long
0112	2	Active energy line 3	Tens of Wh	Unsigned Long
0114	2	Three-phase equivalent reactive/apparent energy [iv]	Tens of VArh/VAh	Unsigned Long
0116	2	Reactive/Apparent energy line 1 [iv]	Tens of VArh/VAh	Unsigned Long

0118	2	Reactive/Apparent energy line 2 [iv]	Tens of VArh/VAh	Unsigned Long
0120	2	Reactive/Apparent energy line 3[iv]	Tens of VArh/VAh	Unsigned Long
0122	2	Max. average three-phase active power	Watt	Unsigned Long
0124	2	Line 1 normal format voltage ThdF (only for Smart Più and Smart96 Più from revision V. 3.10)[x]	Hundredths	Unsigned Long
0126	2	Reserved for line 2 normal format voltage ThdF, but *not used*[x]	-	Unsigned Long
0128	2	Reserved for line 3 normal format voltage ThdF, but *not used*[x]	-	Unsigned Long
0130	2	Line 1 normal format current ThdF (only for Smart Più and Smart 96 Più form revision V. 3.10)[x]	Hundredths	Unsigned Long
0132	2	Reserved for line 2 normal format current ThdF, but *not used*[x]	-	Unsigned Long
0134	2	Reserved for line 3 normal format current ThdF, but *not used*[x]	-	Unsigned Long
0136	2	*Not used*	-	Unsigned Long
0138	2	*Not used*	-	Unsigned Long
0140	2	*Not used*	-	Unsigned Long
0142	2	*Not used*	-	Unsigned Long
0144	2	*Not used*	-	Unsigned Long
0146	2	*Not used*	-	Unsigned Long
0148	2	*Not used*	-	Unsigned Long
0150	2	KV constant	Unit	Unsigned Long
0152	2	KA constant	Unit	Unsigned Long
0154	2	Time of Average	Minute	Unsigned Long
200	2	1st measure configured for the Mix+	Depend	Long
202	2	2nd measure configured for the Mix+	Depend	Long
204	2	3rd measure configured for the Mix+	Depend	Long
206	2	4th measure configured for the Mix+	Depend	Long
208	2	5th measure configured for the Mix+	Depend	Long
210	2	6th measure configured for the Mix+	Depend	Long

(*) Regarding the **power factor**, please note that:

[iv] The instrument return the apparent energy when the CFG parameter is set to 28, 33 or 38. Only for Smart Più, Smart96 Più Version 3.11 or later. See the instrument's manual for more information.

- power factor of the three lines may be negative; the MACH SMART reports negative numbers with the most significant bit (the 32nd) high (1), while positive numbers have the same bit low (0)
- when the power factor for a line is undefined (current is zero), the MACH SMART returns the value "2" to report about this situation

(+) For the **Mix of measure** pay attention to the follows:

- it is available only for Smart Più / Smart 96 Più
- after a global reset, the measures are set to the first 6 measures of the list (frequency, three-phase equivalent voltage...)

(x) For the voltage and current **ThdF** pay attention to the followings:

- the ThdF represents the normalised voltage and current crest factor
- it is available only for Smart Più and Smart 96 Più from revision V. 3.10
- if the instrument is configured with PAR function, it responds to the measure's request in the same way as the measure didn't exist
- the memories 126, 128 and 132, 134 are reserved for line 2 and line 3 ThdF, but they are not used, since the Smart Più and Smart 96 Più measure ThdFs only the line 1
- even if the instrument is set to display the ThdF % (percentage format), when the memory values corresponding to the addresses from 124 to 130 are requested, it answers always with the corresponding ThdF in normal format
- in case the ThdF isn't computable (e.g. when current = 0), the instrument provides two words equal to FFFFh, corresponding to an INVALID DATA

Example of reading a block of 6 Mixed measure:

Addr	Func	Start Register H	Start Register L	Num. Data H	Num. Data L	CRC	CRC
1Fh	03h	00h	C7h	00h	0Ch	F7h	8Ch

In the example it set a reading of 12 bytes from index 200 (minus 1 therefore 199).

Note: "**INVALID DATA**" is shown = FFFFh.[v]

[v] From V. 3.03 on

2.3 Function 05 : "FORCE SINGLE COIL"

Using this function it is possible to set the status of the Smart 96 and Smart Più outputs; the output is treated like a Modbus coil.

The value for "Coil number" is the number of corespondent Output minus 1 therefore for Smart Più outputs 1 or 2 are respectively 00h and 01h; for Smart 96 output 5 is 04h.

The query and answer frames are described below.

Write request (master):

Addr	Func	Coil Num. H	Coil Num. L	Coil Status H	Coil Status L	CRC	CRC
1Fh	05h	00h	04h	FFh	00h	CEh	45h

In the example above the master forces the output to 1: the "force single coil" **Func = 05** is sent to the slave with address **Addr = 1Fh**, followed by **Byte Number = 00 04h** bytes (compulsory). The **Coil Status** value must be one of the following:

- to set the output: **FF 00h**
- to reset the output: **00 00h**

The CRC **CE45h** ends the frame.

Reply (slave):

Addr	Func	Coil Num. H	Coil Num. L	Coil Status H	Coil Status L	CRC	CRC
1Fh	05h	00h	04h	FFh	00h	CEh	45h

If the query is correct, the query frame itself is sent back from the slave to the master, otherwise the exception is reported in the second byte (which becomes **85h**).

This function is available in the Smart 96 from version 1.01 onwards, and in the Smart Più model "2P".

2.4 Function 06 : "PRESET SINGLE REGISTER"

This function lets the user set the Setup parameters of the instrument.

Add.	Word	Parameter description	Min.	Max.	Format
0002	1	V.T. Ratio	1	400	Unsigned int
0004	1	C.T. Ratio	1	1250[vi]	Unsigned int
0006	1	Average period	1	60	Unsigned int
0008	1	MACH reset, may be one of the following: [vii] ***a)*** the value "5" resets the average and max. powers ***b)*** the value "10" resets average powers, max. powers and energies	-	-	Unsigned int
0014	1	MACH address	1	247	Unsigned int
0016	1	Enables (if value=0) or disables (if value=1) the setup menu	-	-	Unsigned int
0018	1	Index 1st value for Mix of measure[viii]	0002	154	Unsigned int
0020	1	Index 2nd value for Mix of measure [iv]	0002	154	Unsigned int
0022	1	Index 3rd value for Mix of measure [iv]	0002	154	Unsigned int
0024	1	Index 4th value for Mix of measure [iv]	0002	154	Unsigned int
0026	1	Index 5th value for Mix of measure [iv]	0002	154	Unsigned int
0028	1	Index 6th value for Mix of measure [iv]	0002	154	Unsigned int
0150	1	V.T. Ratio	1	400	Unsigned int
0152	1	C.T. Ratio	1	1250[vi]	Unsigned int
0154	1	Average period	1	60	Unsigned int

Please note that the write addresses 150, 152 and 154 are just a copy of the addresses 2, 4, 6 in this order, and that there is no difference between the "low" address and "high" address. The high addresses are available for Smart96 version 1.0 onwards, Mach Smart version 1.11 onwards and Smart Più.

A table variable table of measure can be can be configured in Smart più. So with one reading it send the six measure of a pre-set Mix of value (index 18 - 28).

[vi] From version V3.12. With versions V3.11and previous, the maximum value for CT ratio was 1000.

[vii] The reset 15 in not available since the default setting will set the instrument in *DUCATI protocol* and so the instrument will not be resumed any more.

[viii] This value are available only for Smart Più. After a global reset this measure are set to the first six measure of the list (frequency, three-phase equivalent voltage...)

Frame format:

Addr	Func 06	Register H	Register L	DATA H	DATA L	CRC	CRC

Example 1:

Addr	Func	Register H	Register L	DATA H	DATA L	CRC	CRC
1Fh	06h	00h	11h	00h	6Ah	5Ah	5Eh

In the last example the first index(0018) of Mix is set to three-phase active energy (0106).

Notes:

The index must be the real index minus 1 (0018-1 = 0017), while the value of the measure's index must be the actual one (0106).

Example 2:

Addr	Func	Register H	Register L	DATA H	DATA L	CRC	CRC
1Fh	06h	00h	0Fh	00h	01h	7Bh	B7h

In the example the instrument at address 31 receive a keyboard lock (data = 0001) by function 06 at address 0016 (minus 1 = 0015).

2.5 Function 07 : "READ EXCEPTION STATUS"

This function makes it possible to read the status of the instrument. It gives a byte in which each bit (when equal to 1) has the following meaning:

Bit	Meaning
0	Indicates that the instrument has undergone at least one HW Reset since the last reading of that register
1	Not used
2	Not used
3	Not used
4	Not used
5	Not used
6	Not used
7	Indicates that the instrument has the Setup menu active

2.6 Function 17 : "REPORT SLAVE ID"

This function makes it possible to read the instrument identifier.

Read request (master):

Addr	Func	CRC	CRC
02h	11h	C0h	DCh

In this example the id request is sent using **Func = 11h** to the slave with address **Addr = 02h**, the CRC **C0DCh** ends the frame.

Reply (slave):

Addr	Func	Len	Inst. type	///	///	Fw rel.	CRC	CRC
02h	11h	04h	09h	FFh	00h	C8h	F8h	7Bh

The answer contains address and function, the number of data bytes **Len = 04h**, the analyser description **Inst. type = 09h**, and the firmware version **Fw rel. = C8h**: this value must be divided by 100. CRC **F87Bh** ends the frame.

For version 2.56 onwards the field Fw is filled with 00 and the fields that wasn't unused are filled with the hex value of release:

Addr	Func	Len	Inst. type	Fw rel. H	Fw rel. L	///	CRC	CRC
02h	11h	04h	15h	01h	2Ch	00h	h	h

In the example:

Fw rel. 012Ch = V 3.00

Identifier	Analyser model
09	MachSmart three-phase (5A)
10	MachSmart three-phase (50A)
11	MachSmart Dark three-phase (5A)
12	MachSmart Dark three-phase (50A)
13	MachSmart single phase (5A)
14	MachSmart single phase (50A)
15	Smart 96
16	MachSmart Dark single phase (230/240Vac - 5A)
20	Smart Più 5A
21	Smart Più 5A with 2 output as pulse or alarms (models 2P)
22	Smart Più 5A (KVAh)[ix]
23	Smart Più 5A (KVAh)[ix] with 2 output as pulse or alarms (models 2P)
25	Smart 96 Più 5A
26	Smart 96 Più 5A with 2 output as pulse or alarms (models 2P)
27	Smart 96 Più 5A (KVAh)[ix]
28	Smart 96 Più 5A (KVAh)[ix] with 2 output as pulse or alarms (models 2P)
29	Smart Più 1A
30	Smart Più 1A with 2 output as pulse or alarms (models 2P)
31	Smart Più 50A
32	Smart Più 50A with 2 output as pulse or alarms (models 2P)
33	Smart96 Più 1A
34	Smart96 Più 1A with 2 output as pulse or alarms (models 2P)

[ix] You will obtain this ID from Smart Più and Smart 96 più, Ver 3.11 or later, if you set CFG parameter to value 28, 33 or 38. See the instrument's manuals for more information.

35	Smart96 Più 50A
36	Smart96 Più 50A with 2 output as pulse or alarms (models 2P)
60	Smart Più 1A KVAh^{ix}
61	Smart Più 1A KVAh^{ix} with 2 output as pulse or alarms (models 2P)
62	Smart Più 50A KVAh^{ix}
63	Smart Più 50A KVAh^{ix} with 2 output as pulse or alarms (models 2P)
64	Smart96 Più 1A KVAh^{ix}
65	Smart96 Più 1A KVAh^{ix} with 2 output as pulse or alarms (models 2P)
66	Smart96 Più 50A KVAh^{ix}
67	Smart96 Più 50A KVAh^{ix} with 2 output as pulse or alarms (models 2P)

3. Exceptions on the Bus

Below is a table of the exceptions handled for errors regarding access to the bus.

Exception	Description
01 ILLEGAL FUNCTION	An unsupported function code has been sent
02 ILLEGAL DATA ADDRESS	Illegal address
03 ILLEGAL DATA VALUE	A setup datum is outside of the acceptable limits

4. CRC algorithm

The CRC used in modbus follows the standard CRC-16 defined by CCITT. Many algorithms are ready off-the-shelf, below one is reported, written in C, which uses a look-up table.

```
/*    CRC-16 (reverse) table lookup for Modbus CRC-16
 *      Project:  Modbus
 *      Author:   Lynn August Linse, based on method used by XMODEM
 * 16Feb94 LAL Create from book about XMODEM
*/
```

```
word crc16_rev_table[256] =
{     0x0000, 0xC0C1, 0xC181, 0x0140, 0xC301, 0x03C0, 0x0280, 0xC241,
      0xC601, 0x06C0, 0x0780, 0xC741, 0x0500, 0xC5C1, 0xC481, 0x0440,
      0xCC01, 0x0CC0, 0x0D80, 0xCD41, 0x0F00, 0xCFC1, 0xCE81, 0x0E40,
      0x0A00, 0xCAC1, 0xCB81, 0x0B40, 0xC901, 0x09C0, 0x0880, 0xC841,
      0xD801, 0x18C0, 0x1980, 0xD941, 0x1B00, 0xDBC1, 0xDA81, 0x1A40,
      0x1E00, 0xDEC1, 0xDF81, 0x1F40, 0xDD01, 0x1DC0, 0x1C80, 0xDC41,
      0x1400, 0xD4C1, 0xD581, 0x1540, 0xD701, 0x17C0, 0x1680, 0xD641,
      0xD201, 0x12C0, 0x1380, 0xD341, 0x1100, 0xD1C1, 0xD081, 0x1040,
      0xF001, 0x30C0, 0x3180, 0xF141, 0x3300, 0xF3C1, 0xF281, 0x3240,
      0x3600, 0xF6C1, 0xF781, 0x3740, 0xF501, 0x35C0, 0x3480, 0xF441,
      0x3C00, 0xFCC1, 0xFD81, 0x3D40, 0xFF01, 0x3FC0, 0x3E80, 0xFE41,
      0xFA01, 0x3AC0, 0x3B80, 0xFB41, 0x3900, 0xF9C1, 0xF881, 0x3840,
      0x2800, 0xE8C1, 0xE981, 0x2940, 0xEB01, 0x2BC0, 0x2A80, 0xEA41,
      0xEE01, 0x2EC0, 0x2F80, 0xEF41, 0x2D00, 0xEDC1, 0xEC81, 0x2C40,
      0xE401, 0x24C0, 0x2580, 0xE541, 0x2700, 0xE7C1, 0xE681, 0x2640,
      0x2200, 0xE2C1, 0xE381, 0x2340, 0xE101, 0x21C0, 0x2080, 0xE041,
      0xA001, 0x60C0, 0x6180, 0xA141, 0x6300, 0xA3C1, 0xA281, 0x6240,
      0x6600, 0xA6C1, 0xA781, 0x6740, 0xA501, 0x65C0, 0x6480, 0xA441,
      0x6C00, 0xACC1, 0xAD81, 0x6D40, 0xAF01, 0x6FC0, 0x6E80, 0xAE41,
      0xAA01, 0x6AC0, 0x6B80, 0xAB41, 0x6900, 0xA9C1, 0xA881, 0x6840,
      0x7800, 0xB8C1, 0xB981, 0x7940, 0xBB01, 0x7BC0, 0x7A80, 0xBA41,
      0xBE01, 0x7EC0, 0x7F80, 0xBF41, 0x7D00, 0xBDC1, 0xBC81, 0x7C40,
      0xB401, 0x74C0, 0x7580, 0xB541, 0x7700, 0xB7C1, 0xB681, 0x7640,
      0x7200, 0xB2C1, 0xB381, 0x7340, 0xB101, 0x71C0, 0x7080, 0xB041,
      0x5000, 0x90C1, 0x9181, 0x5140, 0x9301, 0x53C0, 0x5280, 0x9241,
      0x9601, 0x56C0, 0x5780, 0x9741, 0x5500, 0x95C1, 0x9481, 0x5440,
      0x9C01, 0x5CC0, 0x5D80, 0x9D41, 0x5F00, 0x9FC1, 0x9E81, 0x5E40,
      0x5A00, 0x9AC1, 0x9B81, 0x5B40, 0x9901, 0x59C0, 0x5880, 0x9841,
      0x8801, 0x48C0, 0x4980, 0x8941, 0x4B00, 0x8BC1, 0x8A81, 0x4A40,
      0x4E00, 0x8EC1, 0x8F81, 0x4F40, 0x8D01, 0x4DC0, 0x4C80, 0x8C41,
      0x4400, 0x84C1, 0x8581, 0x4540, 0x8701, 0x47C0, 0x4680, 0x8641,
      0x8201, 0x42C0, 0x4380, 0x8341, 0x4100, 0x81C1, 0x8081, 0x4040};

unsigned fast_crc16( unsigned char *ucpBuf, int nSize){
      register word x;
      register word crc;

      int i;
      crc = 0xFFFF;      /* start with all 1's for a reverse CRC */

      for( i = 0; i < nSize; ++i) {
            /* process each character in the message - 2 steps per char only! */
            x = crc ^ ucpBuf[i];
            crc = (crc >> 8) ^ crc16_rev_table[x & 0x00FF];
      }
      return( crc);
}
```