MODBUS-RTU for HPX

MODBUS-RTU protocol specifications for HPX device net controlling

READ AND KEEP

REV. 09-02
ENG

INDEX

1: DESCRIZIONE GENERALE

1.1 *MODBUS PROTOCOL*

The data communication system based on Modbus protocol allows to connect up to 247 devices in a common RS485 line with standard format and communication mode.
Communication takes place in half duplex by frame (transmitted continuously); only master (PC , PLC ...) can start polling with slaves as question/answer (only one slave addressed) and the polled slave answers. The slave answers after a minimum pause of 3,5 characters between received frame and the one to be transmitted.
Also broadcast communication mode exists where the master send a request to all the slaves simultaneously, and they give no answer back; this mode it's not available with HPX controllers.
The data serial transmission mode implemented on HPX controller is RTU type (Remote Terminal Unit), where data are exchanged in binary format (8 bit characters).

1.2 *SERIAL CONFIGURATION*

Serial line:	***RS485***
Baud rate:	***9600***
Data lenght:	***8 bit***
Parity:	***none***
Stop bit:	***1***

Serial transmission of characters in RTU format

Start	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	Stop

1.3 *MESSAGGE FORMAT (FRAME)*

Each message (Frame) is made, based on MODBUS-RTU standard, by the following parts:

Start	*Device address*	*Function code*	*Data*	*CRC16*		*Stop*
4,5msec pause	Byte	Byte	n x Byte	LSByte	MSByte	4,5msec pause

- **Start / Stop :**
  Message starts with a 4,5ms pause (time higher than 3.5 times the character transmission period).
  See chap. 4.1 for further clarifications.

- **Device address**:
  Device address with whom the master established the polling; it's a value between 1 and 247. Address 0 is reserved to the broadcast, message sent to all slave devices (not active on HPX). RS485 line allows to connect together up to 32 devices (1 Master + 31 slaves), but with appropriate "bridges" or relay devices it is possible to use the whole logical addressing field.

- **Function Code:**
  Code of the function to be execute or already executed; On HPX device codes 0x03 (register reading) and 0x06 (single register writing) are active.

- **Data:**
  Data that must be exchanged.

- **CRC16:**
  Error checking field based on CRC16 algorithm. CRC16 is calculated on the whole message by the master device which is trasmitting and attached to the message itself. The slave, at the end of reception, calculate CRC16 on the message and compare it with the value added by the master; if the values does not match the message will be considered not valid and will be discarded without sending any answer to the master.
  The following fragment of C code shows the CRC16 calculation mode:

```
unsigned int CRC16
void Modbus_CRC(unsigned char *Frame, unsigned char FrameLength)
{
unsigned char ByteCount;
unsigned char i;
unsigned char bit_lsb;
CRC16 = 0xFFFF;
for (ByteCount=0;ByteCount<FrameLength;ByteCount++)
  {
  CRC16^=Frame[ByteCount];
  for (i=0;i<8,i++)
    {
    bit_lsb = CRC16 & 0x0001;
    CRC16 = CRC16>>1;
    if (bit_lsb == 1)
      CRC16 ^= 0xA001;
    }
  }
```

}

1.4 *MESSAGGES SYNCHRONIZATION*

Message synchronization between transmitter and receiver is made placing a pause on the messages at least 4 ms. If the receiver does not receive any Byte for a 4 ms time, consider the last message completed and set the next Byte received ad the first one of a new message.
The slave, once received the complete message, decode it and, if there are no errors, send the answer to the master. To send the answer, slave keeps RS485 line busy, wait a 4,5 ms pause, send the complete message, wait a 4,5 ms pause and then release the RS485 line.
The master will have to consider these periods to avoid risks of trasmission overlap; in particular must be set a proper answer reception time-out before starting a new transmission (typical time-out value: 500msec or higher).

1.5 *ERROR MESSAGGES (EXCEPTIONS)*

The HPX device, if not possible to complete the required operation, answer with an error message, in the following format:

Device address	*Function Code*	*Exception Code*	*CRC16*	
Byte	Byte	Byte	LSByte	MSByte

- **Device address**:
  Address of slave device answering
- **Function Code**:
  Function code Msb =1 (to show exception); i.e. 0x83 (reading) or 0x86 (writing)
- **Exception Code**:
  Exception codes handled by HPX device are the following:

Exception code	*Description*	*Exception cause*
0x01	Function not implemented	A function code not available was requested, different from 0x03 and 0x06.
0x03	Address not valid	It's generated in several situations: - a not implemented register was requested (or a not-existing area) - a reading of more than 10 registers was requested - a reading of a number of registers more than the implemented are was requested (starting from requested address) - tried to write on a read-only area - tried to write an 8-bit parameter with a 16-bit value

- **CRC16:**
  Error control field based on the CRC16 algorithm.

Note:
In case th HPX device identify in the received message an error on format or CRC16, message is discarded (considered not valid) and no answer is sent.

2: COMMANDS DESCRIPTION

All the registers, to equalize the interpretation, are handled in a Word format (16 bit), even if an 8-bit parameter is cointained.

2.1 *REGISTER READING (0x03)*

Format of command sent by the Master:

Device address	*Function Code*	*Register address*		*Number of registers*		*CRC16*	
Byte	Byte	MSByte	LSByte	MSByte	LSByte	LSByte	MSByte

- **Device address**:
  Address of slave device to be polled
- **Function Code**:
  Function code to be executed, in this case register reading (0x03)
- **Register address**:
  Starting register address for reading expressed with two Bytes; (MSByte) and (LSByte).
- **Number of registers**:
  indicates the number of Word required from the starting address. If a number of registers more than 1 are requested, the answer message will provide all the registers required with consecutive addresses starting from the address shown on the "register address" field.
  The number of registers to read is expressed on two Bytes, particularly for HPX controller (MSByte) must always be 0x00 and (LSByte) with range 1-10.
- **CRC16:**
  Error control field based on the CRC16 algorithm.

Format of answer message from slave:

Device address	*Function Code*	*No. Bytes of data*	*Data 1*		*Data 2*		*Data n*		*CRC16*	
Byte	Byte	Byte	MSByte	LSByte	MSByte	LSByte	MSByte	LSByte	LSByte	MSByte

- **Device address**:
  Address of slave device answering
- **Function Code**:
  Function code to be answered to, in this case register reading (0x03)
- **No. Bytes of data**:
  contains the total Bytes number of data.
  Consider that the number of Bytes of data is double the number of registers (because we talk about word). I.e. if in the polling message 2 registers are requested, in the answer message number of data Bytes must be setted as 4.
- **Data n** :
  contains data sequences each expressed on two Bytes; (MSByte) and (LSByte).
- **CRC16:**

Error control field based on the CRC16 algorithm.

2.2 *SINGLE REGISTER WRITING (0x06)*

Format of command sent by the Master:

Device address	*Function Code*	*Register address*		*Data*		*CRC16*	
Byte	Byte	MSByte	LSByte	MSByte	LSByte	LSByte	MSByte

- **Device address**:
  Address of slave device to be polled
- **Function Code**:
  Function code to be executed, in this case single register writing (0x06)
- **Register address**:
  address of register to write expressed with two Bytes; (MSByte) and (LSByte).
- **Data**:
  Value to be assigned to the register expressed with two Bytes; (MSByte) and (LSByte).
- **CRC16:**
  Error control field based on the CRC16 algorithm.

Format of answer message from slave:

Device address	*Function Code*	*Register address*		*Data*		*CRC16*	
Byte	Byte	MSByte	LSByte	MSByte	LSByte	LSByte	MSByte

The answer message is a simple echo of the polling message to confirm that the variable was modified.

WARNING :

On the HPX device is not present a control on parameter's range (apart the clock 512-513 registers). Master must have all the controls to avoid setting of out-of-range values. The range of parameters are shown on the tables in the next chapters.

Please do not send commands of writing on single register in a continuously way to avoid that 100-Master controller keeps being busy on management of Modbus commands received instead of communication with the console.

3: REGISTERS AND ADDRESSES DESCRIPTION

Each register has a 16 bit dimension. It has been formed some blocks of variables (each with a different MSByte address) basing on the the type of these variables. In the followings paragraphs are described in the detail all the available blocks and, for each block, the implemented variables.

3.1 *ANALOG INPUTS (READ-ONLY)*

Register	*Description*	*Bytes range and meanings*	
256	ambient temperature (resolution 0,5°C)	MSByte	whole part of temperature in °C with offset 45
		LSByte	decimal part of temperature in °C with offset 45 range: 0..9
257	external temperature (resolution 1°C)	MSByte	whole part of temperature in °C with offset 45
		LSByte	decimal part of temperature in °C with offset 45 range: 0..9
258	defrost probe temperature (resolution 1°C)	MSByte	whole part of temperature in °C with offset 45
		LSByte	decimal part of temperature in °C with offset 45 range: 0..9
259	fans current absorption (A) (resolution 0,1A)	MSByte	16-bit value high part
		LSByte	16-bit value low part

Examples for the temperature:
(0x5A00) MSByte = 90, LSByte = 0 correspondent to a temperature of 90,0 – 45 = +45,0°C
(0x4105) MSByte = 65, LSByte = 5 correspondent to a temperature of 65,5 – 45 = +20,5°C
(0x2D00) MSByte = 45, LSByte = 0 correspondent to a temperature of 45,0 – 45 = 0,0°C
(0x2B05) MSByte = 43, LSByte = 5 correspondent to a temperature of 43,5 – 45 = -1,5°C
(0x0F05) MSByte = 15, LSByte = 5 correspondent to a temperature of 15,5 – 45 = -29,5°C

Example for the current:
(0x0030) MSByte = 0, LSByte = 48 correspondent to a current of 4,8 A
Note: The visualization on display of the current is limited to 25.5A while register 259 reach 32A

3.2 *CLOCK (READ / WRITE)*

3.3

Register	*Description*	*Bytes range and meanings*	
512	hour	MSByte	range: 0..59
	minutes	LSByte	range: 0..23
513	day of the week	MSByte	range: 1..7
		LSByte	

PARAMETERS (READ / WRITE)

Register	*Description*	*Bytes range and meanings*	
768	manual temperature setpoint	MSByte	in °C with offset 45 range: 50..80, correspondent to +5..+35°C
		LSByte	
769	**r0** temperature differential	MSByte	range: 1..4 °C
		LSByte	
770	**r1** free-cooling differential	MSByte	range: 1..4 °C
		LSByte	
771	**tg** anti-freeze resistance insertion differential	MSByte	range: 0..5 °C
		LSByte	
772	**t1** first band P1 temperature setpoint	MSByte	in °C with offset 45 range: 50..80, correspondent to +5..+35°C
		LSByte	
773	**t2** second band P1 temperature setpoint	MSByte	in °C with offset 45 range: 50..80, correspondent to +5..+35°C
		LSByte	
774	**t3** P2 band temperature setpoint	MSByte	in °C with offset 45 range: 50..80, correspondent to +5..+35°C
		LSByte	
775	**tS1** first band P1 start hour	MSByte	step of 10 minutes range: 0..143, correspondent to 0:00..23:50
		LSByte	
776	**tF1** first band P1 end hour	MSByte	step of 10 minutes range: 0..143, correspondent to 0:00..23:50
		LSByte	
777	**tS2** second band P1 start hour	MSByte	step of 10 minutes range: 0..143, correspondent to 0:00..23:50
		LSByte	
778	**tF2** second band P1 end hour	MSByte	step of 10 minutes range: 0..143, correspondent to 0:00..23:50
		LSByte	

Register	*Description*	*Bytes range and meanings*	
779	**tS3** P2 band start hour	MSByte	step of 10 minutes range: 0..143, correspondent to 0:00..23:50
		LSByte	
780	**tF3** P2 band end hour	MSByte	step of 10 minutes range: 0..143, correspondent to 0:00..23:50
		LSByte	
781	**G1** active program in AUTO mode on day 1	MSByte	range: 0..3 0 = P1, 1 = P2, 2 = P3, 3 = P4
		LSByte	
782	**G2** active program in AUTO mode on day 1	MSByte	range: 0..3 0 = P1, 1 = P2, 2 = P3, 3 = P4
		LSByte	
783	**G3** active program in AUTO mode on day 1	MSByte	range: 0..3 0 = P1, 1 = P2, 2 = P3, 3 = P4
		LSByte	
784	**G4** active program in AUTO mode on day 1	MSByte	range: 0..3 0 = P1, 1 = P2, 2 = P3, 3 = P4
		LSByte	
785	**G5** active program in AUTO mode on day 1	MSByte	range: 0..3 0 = P1, 1 = P2, 2 = P3, 3 = P4
		LSByte	
786	**G6** active program in AUTO mode on day 1	MSByte	range: 0..3 0 = P1, 1 = P2, 2 = P3, 3 = P4
		LSByte	
787	**G7** active program in AUTO mode on day 1	MSByte	range: 0..3 0 = P1, 1 = P2, 2 = P3, 3 = P4
		LSByte	
788	**d0** defrost delay	MSByte	range: 1..30 minutes
		LSByte	
789	**d1** defrost start setpoint	MSByte	in °C, with offset 45 range: 25..50, correspondent to -20..+4°C Note: upper limit of d1 is (d2-1)
		LSByte	

Register	*Description*	*Bytes range and meanings*	
790	**d2** defrost end setpoint	MSByte LSByte	in °C, with offset 45 range: 25..50, correspondent to -19..+5°C Note: lower limit of d2 is (d1+1)
791	**d3** max. defrost duration	MSByte LSByte	range: 1..254 minutes
792	**F5** fan stop duration	MSByte LSByte	range: 1..254 seconds
793	**F1** min. time between two start of each compressor	MSByte LSByte	range: 1..15 minutes
794	**F2** min. time of compressor functioning	MSByte LSByte	range: 1..15 minutes
795	**du** cycle-inversion valves activation delay	MSByte LSByte	range: 10..20 seconds
796	**rC** second compressor activation delay	MSByte LSByte	step of 5 seconds range: 1..120, correspondent to 5..600 secondi

3.4 *INPUTS / OUTPUTS / ALARMS STATUS (READ-ONLY)*

Register	*Description*	*Bytes meanings*		
1280	outputs status	MSByte	bit 7 (MSb)	Not used
			bit 6	
			bit 5	
			bit 4	
			bit 3	
			bit 2	
			bit 1	
			bit 0 (LSb)	
		LSByte	bit 7 (MSb)	by-pass shutter relay (K9)
			bit 6	cycle-inversion valve 2 relay (K8)
			bit 5	compressor 2 relay (K7)
			bit 4	cycle-inversion valve 1 relay (K5)
			bit 3	anti-freeze resistance relay (K4)
			bit 2	fans relay (K3)
			bit 1	aux. heater resistances relay (K2)
			bit 0 (LSb)	compressor 1 relay (K1)

Register	*Description*	*Bytes meanings*		
1281	inputs status	MSByte	bit 7 (MSb)	Not used
			bit 6	
			bit 5	
			bit 4	
			bit 3	
			bit 2	
			bit 1	
			bit 0 (LSb)	
		LSByte	bit 7 (MSb)	not used
			bit 6	remote OFF input
			bit 5	(EP) maintenance alarm
			bit 4	(E1) compressor 2 protection
			bit 3	(EF) missing ventilation alarm
			bit 2	(E8) supply dirty filters alarm
			bit 1	(E7) exhaust dirty filters alarm
			bit 0 (LSb)	(EC) compressor 1 protection

Register	*Description*	*Bytes meanings*		
1282	alarms status	MSByte	bit 7 (MSb)	Not used
			bit 6	
			bit 5	
			bit 4	
			bit 3	
			bit 2	
			bit 1	
			bit 0 (LSb)	
		LSByte	bit 7 (MSb)	not used
			bit 6	master - console disconnection error
			bit 5	not used
			bit 4	(E9) too cold external air alarm
			bit 3	(E2) EEPROM error
			bit 2	(E6) defrost probe anomaly
			bit 1	(E5) external probe anomaly
			bit 0 (LSb)	(E0) ambient probe anomaly

3.5 *DEVICE STATUS (READ / WRITE)*

Register	*Description*	*Bytes meanings*		
1536	device status	MSByte	bit 7 (MSb)	not used
			bit 6	not used
			bit 5	not used
			bit 4	not used
			bit 3	not used
			bit 2	not used
			bit 1	enable chrono-thermostat status modification
			bit 0 (LSb)	enable stand-by status modification
		LSByte	bit 7 (MSb)	not used
			bit 6	not used
			bit 5	not used
			bit 4	not used
			bit 3	not used
			bit 2	not used
			bit 1	chrono-thermostat status 1=AUTO (chrono-thermostat enabled) 0=MAN (chrono-thermostat disabled)
			bit 0 (LSb)	stand-by status 1 = stand-by 0 = ON

To request modification of one of the device status bit, the master must send on LSByte the required value for the bit and on MSByte the correspondent bit setted to 1.
Example: to force stand-by status, master must send MSByte = 0b00000001 and LSByte = 0b00000001. To disable chrono-thermostat must send MSByte = 0b00000010 e LSByte = 0b00000000.

4: GLOSSARY

- **Binary Number:**
  It is used in computer science for the internal representation of numbers, thanks to the simplicity to physically realize an element with two state (0,1) instead an higher number, but also with the matching with the logic values TRUE and FALSE.

- **Decimal Numer:**
  On decimal system all whole numbers can be represented using the ten digits that indicates the first ten natural numbers, included zero. The value of each of these digits depends on the position occupied inside the number, and it increases in powers of 10, from right to left.

- **Hexadecimal Number:**
  It is part of a positional numeric system with base 16, that means it uses 16 symbols instead usual 10 of the traditional numerical deciaml system. Hexadecimal generally uses symbols from 0 to 9 and then letters from A to F, for a total 16 symbols. Conventionally an hexadecimal number is preceded by 0x (i.e. 0x03) or by H (i.e. H03).

- **bit:**
  A bit is a binary digit that is one of the two symbols of numerical binary system, usually called zero (0) and one (1). It represents the definition unit of a logic state.
  It's defined also as elementary unit of the information used by a computer.

- **Byte:**
  It's the quantity of bit needed to define an alphanumeric character; particularly a Byte is made by a sequence of 8 bit (i.e. 10010110).

- **Word:**
  Unit of measure that fixes information lenght at 16 bits that is equivalent to 2 Bytes (i.e. 10010110 01101011).

- **LSb:**
  Less significant bit of a binary digit (first bit on the right of the indicated number)

- **MSb:**
  Most significant bit of a binary digit (first bit on the left of the indicated number)

- **LSByte:**
  Less significant Byte of a Word (Byte on the right of the indicated Word)

- **MSByte:**
  Most significant Byte of a Word (Byte on the left of the indicated Word)