MODBUS-RTU for PLUSR EXPERT DL8

MODBUS-RTU protocol specifications
for LAN control of
PLUSR EXPERT DL8
series devices

Document: **MODBUS-RTU_PLUSRDL8_02-16_ENG**
Installed Software: **PLUSR_DL8 Rev. 0**

REED AND KEEP

INDEX

1: GENERAL DESCRIPTION

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 this controller.
The data serial transmission mode implemented on the 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:	***1200, 2400, 4800, 9600, 14400, 19200, 38400***
Data lenght:	***8 bit***
Parity:	***none, even or odd***

Serial transmission of characters in RTU format

Start	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	Parity (optional)	Stop

1.3 *MESSAGE FORMAT (FRAME)*

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

Start	*Device address*	*Function code*	*Data*	*CRC16*		*Stop*
pause (3.5 times the character transmission period)	Byte	Byte	n x Byte	LSByte	MSByte	pause (3.5 times the character transmission period)

- **Start / Stop :**
  Message starts with pause higher than 3.5 times the character transmission period.
  See chap. 1.4 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 this controller). 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 device are acteve codes 0x03 (register reading), 0x06 (single register writing) and 0x2B/0x0E (identification data reading).

- **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, calculates CRC16 on the message and compares it with the value learnt by the master; if the values do 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 MESSAGES SYNCHRONIZATION

Message synchronization between transmitter and receiver is made placing a pause on the messages at least 3.5 times the character transmission period. If the receiver does not receive any Byte for 3.5 times the character transmission period, 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, decodes it and, if there are no errors, sends the answer message to the master. To send the answer, slave keeps RS485 line busy, wait a pause of 3.5 times the character transmission period, send the complete message, wait 3.5 times the character transmission period and then release the RS485 line.
The master unit will have to consider these periods to avoid risks of transmission overlap; in particular must be set a proper answer reception time-out before starting a new transmission (typical time-out value: 500msec or higher, for a baud rate = 9600).

1.5 ERROR MESSAGES (EXCEPTIONS)

The device, if not possible to complete the required operation, answers 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 (for 0x03 reading) or 0x86 (for 0x06 writing)
- **Exception Code**:
  Exception codes handled by the device are the following:

Exception code	Description	Exception cause
0x01	Function not implemented	A function code not available was requested, different from 0x03, 0x06 and 0x2B/0x0E.
0x02	Address not valid	It's generated in several situations: - a not implemented register has been requested (or a not-existing area) - a reading of a number of registers that goes further on the implemented area has been requested (starting from requested address) - tried to write on a read-only area
0x03	Value not valid for datum	It's generated in several situations: - message 0x2B/0x0E DeviceIdCode is not correct - has been tried to write a parameter with an out of range value

Error control field based on the CRC16 algorithm.

Note:
In case the device identifies in the received message an error on format or in CRC16, the 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 contained.

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 is 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 this 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*	*Bytes of datum No.*	*Datum 1*		*Datum 2*		*Datum 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)
- **Bytes' number of datum**:
  Contains the total Bytes number of data.
  Consider that the Bytes' number of datum is the double of the number of registers (because we talk about word). I.e. if in the polling message 2 registers are requested, in the answer message Bytes' number of datum must be set as 4.
- **Datum 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***		***Datum***		***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***		***Datum***		***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 has been modified.

2.3 DATA READING OF DEVICE IDENTIFICATION (0x2B / 0x0E)

Format of command sent by the Master:

Device address	Function Code	MEI type	Read Device Id Code	Object Id	CRC16	
Byte	Byte	Byte	Byte	Byte	LSByte	MSByte

- **Device address**:
  Address of slave device to be polled
- **Function Code**:
  Function code to be executed, in this case identification data reading (0x2B)
- **MEI type**:
  Modbus Encapsulated Interface type: it must be 0x0E.
- **Read Device Id Code**:
  Indicates the access type to data: it must be 0x01.
- **Object Id**:
  Indicates the starting object for data reading (range: 0x00 – 0x02).
- **CRC16:**
  Error control field based on the CRC16 algorithm.

Format of answer message from slave:

Device address	Function code	MEI Type	Read Device Id Code	Conformity level	More Follows	Next Object Id	Number Of Object	Object Id (n)	Object Length (n)	Object Value (n)	CRC16	
Byte	Byte	Byte	Byte	Byte	Byte	Byte	Byte	Byte	Byte	ASCII String	LSByte	MSByte

- **Device address**:
  Address of slave device answering
- **Function Code**:
  Function code to be executed, in this case identification data reading (0x2B)
- **MEI type**:
  Modbus Encapsulated Interface type: it must be 0x0E.
- **Read Device Id Code**:
  Indicates the access type to data: it must be 0x01.
- **Conformity level**:
  indicates the slave conformity level: it is always 0x01.
- **More Follows**:
  indicates the number of additional transactions requested: it is always 0x00.
- **Next Object Id**:
  indicates the object that has to be requested in the eventual following transaction: it is always 0x00

- **Number Of Object**:
  number of objects that follow (1, 2 o 3).
- **List of**:
  - **Object Id**:
    current object number .
  - **Object Length**:
    length of following string.
  - **Object Value**:
    ASCII string that contains the identification information.
- **CRC16:**

Error control field based on the CRC16 algorithm.

Reading example of all controllers identification information with software PLUSR_DL8 rel. 0 (address 1)

Demand message: (01 2B 0E 01 00 70 77)

- **Device address**: 0x01
- **Function code**: 0x2B
- **MEI type**: 0x0E
- **Read DeviceIdCode**: 0x01
- **ObjectId**: 0x00
- **CRC16:** to be calculated on previous values

Answer message: (:01 2B 0E 01 01 00 00 03 00 04 50 45 47 4F 01 08 50 4C 55 53 52 44 4C 38 02 03 30 30 30 68 F2)

- **Device address**: 0x01
- **Function code**: 0x2B
- **MEI type**: 0x0E
- **Read DeviceIdCode**: 0x01
- **Conformity level**: 0x01
- **More Follows**: 0x00
- **Next ObjectId**: 0x00
- **Number Of Object**: 0x03
- **ObjectId**: 0x00
- **Object Length**: 0x04
- **Object Value**: 'PEGO' (Vendor Name field)
- **ObjectId**: 0x01
- **Object Length**: 0x08
- **Object Value**: 'PLUSRDL8' (Product Code field)
- **ObjectId**: 0x02
- **Object Length**: 0x03
- **Object Value**: '000' (Revision field)
- **CRC16:** to be calculated on previous values

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.
At the beginning of each table it has been indicated in the first row if its data could be only read (READ-ONLY) or written and read (READ/WRITE).

TABLE COLUMNS DESCRIPTION:

- **Register** :
  It indicates the register address that has to be used in the structure of Modbus command for reading or writing the data into device. It is expressed on two Bytes: (MSByte) and (LSByte).
- **Description** :
  Description of the register and possible corresponding programming variable of the device.
- **Meaning and Bytes range:**
  Dimension (MSByte and LSByte), allowed range and notes about register.
- **U.M. :**
  Unit of measure of datum contained in the register.
- **Conv. :**
  Values contained in the registers that represent signed variables require a conversion and they are marked from **X** sign in the following column.
  Conversion procedure:
  - If the value contained in the register is included between 0 and 32767, it represents a positive or null number (the results is the value itself)
  - If the value contained in the register is included between 32768 and 65535, it represents a negative number (the results is the register value - 65536)
- **Molt :**
  It indicates the multiplication factor that has to be mapped to register's datum and that coupled to columns U.m and Conv permits the right interpretation of the value to convert.
  Esempi:
  A datum **(0x0012)** = 18 with Molt =**0,1** / U.m= **°C** / Conv=**C** corresponds to a temperature of (18x0,1)= **1,8 °C**
  A datum **(0xFFF0)** = 65520 with Molt =**0,1** / U.m= **°C** / Conv=**C** corresponds to a temperature [(65520 – 65536) x0,1] = **-1,6 °C**
  A datum **(0x0078)** = 120 with Molt =**1** / U.m= **min** / Conv=**C** corresponds to a time of (120x1)= **120 minutes**
  A datum **(0x0014)** = 20 with Molt =**0,1** / U.m= **°C** / Conv=**C** corresponds to a temperature of (20x0,1)= **2,0 °C**

3.1 *ANALOG INPUTS*

READ-ONLY						
Register	***Description***	***Bytes meaning and range***		***U.M.***	***Conv***	***Molt***
256	Channel 1 temperature	MSByte	Resolution 0,1°C range: -45°C .. +99°C Values > +99°C indicate broken probe	°C	X	0,1
		LSByte				
257	Channel 2 temperature	MSByte	Resolution 1°C range: -45°C .. +99°C Values > +99°C indicate broken probe	°C	X	0,1
		LSByte				
258	Channel 3 temperature	MSByte	Resolution 1°C range: -45°C .. +99°C Values > +99°C indicate broken probe	°C	X	0,1
		LSByte				

READ-ONLY						
Register	**Description**	**Bytes meaning and range**		**U.M.**	**Conv**	**Molt**
259	Channel 4 temperature	MSByte LSByte	Resolution 0,1°C range: -45°C .. +99°C Values > +99°C indicate broken probe	°C	X	0,1
260	Channel 5 temperature	MSByte LSByte	Resolution 1°C range: -45°C .. +99°C Values > +99°C indicate broken probe	°C	X	0,1
261	Channel 6 temperature	MSByte LSByte	Resolution 1°C range: -45°C .. +99°C Values > +99°C indicate broken probe	°C	X	0,1
262	Channel 7 temperature	MSByte LSByte	Resolution 1°C range: -45°C .. +99°C Values > +99°C indicate broken probe	°C	X	0,1
263	Channel 8 temperature	MSByte LSByte	Resolution 1°C range: -45°C .. +99°C Values > +99°C indicate broken probe	°C	X	0,1

3.2 PARAMETERS

READ / WRITE						
Register	**Description**	**Bytes meaning and range**		**U.M.**	**Conv**	**Molt**
768	**A11** Minimum temperature T1 alarm	MSByte LSByte	0.1 °C steps, with sign range: -45.0°C..(A2-0.1°C)	°C	X	0.1
769	**A12** Maximum temperature T1 alarm	MSByte LSByte	0.1 °C steps, with sign range: (A1+0.1°C)..+99.0°C	°C	X	0.1
770	**A21** Minimum temperature T2 alarm	MSByte LSByte	0.1 °C steps, with sign range: -45.0°C..(A2-0.1°C)	°C	X	0.1
771	**A22** Maximum temperature T2 alarm	MSByte LSByte	0.1 °C steps, with sign range: (A1+0.1°C)..+99.0°C	°C	X	0.1
772	**A31** Minimum temperature T3 alarm	MSByte LSByte	0.1 °C steps, with sign range: -45.0°C..(A2-0.1°C)	°C	X	0.1
773	**A32** Maximum temperature T3 alarm	MSByte LSByte	0.1 °C steps, with sign range: (A1+0.1°C)..+99.0°C	°C	X	0.1

READ / WRITE						
Register	**Description**	**Bytes meaning and range**		**U.M.**	**Conv**	**Molt**
774	**A41** Minimum temperature T1 alarm	MSByte LSByte	0.1 °C steps, with sign range: -45.0°C..(A2-0.1°C)	°C	X	0.1
775	**A42** Maximum temperature T1 alarm	MSByte LSByte	0.1 °C steps, with sign range: (A1+0.1°C)..+99.0°C	°C	X	0.1
776	**A51** Minimum temperature T2 alarm	MSByte LSByte	0.1 °C steps, with sign range: -45.0°C..(A2-0.1°C)	°C	X	0.1
777	**A52** Maximum temperature T2 alarm	MSByte LSByte	0.1 °C steps, with sign range: (A1+0.1°C)..+99.0°C	°C	X	0.1
778	**A61** Minimum temperature T3 alarm	MSByte LSByte	0.1 °C steps, with sign range: -45.0°C..(A2-0.1°C)	°C	X	0.1
779	**A62** Maximum temperature T3 alarm	MSByte LSByte	0.1 °C steps, with sign range: (A1+0.1°C)..+99.0°C	°C	X	0.1
780	**A71** Minimum temperature T3 alarm	MSByte LSByte	0.1 °C steps, with sign range: -45.0°C..(A2-0.1°C)	°C	X	0.1
781	**A72** Maximum temperature T3 alarm	MSByte LSByte	0.1 °C steps, with sign range: (A1+0.1°C)..+99.0°C	°C	X	0.1
782	**A81** Minimum temperature T3 alarm	MSByte LSByte	0.1 °C steps, with sign range: -45.0°C..(A2-0.1°C)	°C	X	0.1
783	**A82** Maximum temperature T3 alarm	MSByte LSByte	0.1 °C steps, with sign range: (A1+0.1°C)..+99.0°C	°C	X	0.1
784	**ALd** temperature alarm signaling delay	MSByte LSByte	1 minutes steps range: 0..240 minutes	min		1
785	**ALr** Delay in alarm buzzer reactivation	MSByte LSByte	1 minutes steps range: 0..240 minutes 0 = disabled	min		1
786	**BEE** Buzzer enable	MSByte LSByte	0 = disabled 1 = enabled	num		1

3.2a ***PARAMETERS***

READ-ONLY						
Registro	***Descrizione***	***Significato e range Bytes***		***U.M.***	***Conv***	***Molt***
512	**t1** temperature channel enabling	MSByte	0= Excluded 1= Enabled	num		1
		LSByte				
513	**t2** temperature channel enabling	MSByte	0= Excluded 1= Enabled	num		1
		LSByte				
514	**t3** temperature channel enabling	MSByte	0= Excluded 1= Enabled	num		1
		LSByte				
515	**t4** temperature channel enabling	MSByte	0= Excluded 1= Enabled	num		1
		LSByte				
516	**t5** temperature channel enabling	MSByte	0= Excluded 1= Enabled	num		1
		LSByte				
517	**t6** temperature channel enabling	MSByte	0= Excluded 1= Enabled	num		1
		LSByte				
518	**t7** temperature channel enabling	MSByte	0= Excluded 1= Enabled	num		1
		LSByte				
519	**t8** temperature channel enabling	MSByte	0= Excluded 1= Enabled	num		1
		LSByte				
520	**tA** Status changeover NA – NC alarm relays	MSByte	0= Contact closed with alarm presence 1= Contact opened with alarm presence	num		1
		LSByte				
521	**SAv** Automatic backup of recorder plus memory on USB device.	MSByte	0 = Excluded 1 = Every day at 12.00. 2 = Every first day of the month at 12.00.	num		1
		LSByte				
522	**int** Temperature registration interval,	MSByte	range: 0 ÷ 60 minutes If int =0 registration disabled	min		1
		LSByte				
523	**ASr** Asynchronous registration	MSByte	0 = disabled 1 = enabled	num		1
		LSByte				
524	**BAt** Backup battery state	MSByte	*Power supply off:* 0 … 100 % (level) *Power supply on:* 0 : battery disconnected or broken 1 : battery charging 2 : battery charged	num		1
		LSByte				

3.3 INPUTS / OUTPUTS / ALARMS STATUS

READ-ONLY							
Register	**Description**	**Bytes meaning**			**U.M.**	**Conv**	**Molt**
1280	output status	MSByte	bit 7 (MSb)	Not used	num		1
			bit 6				
			bit 5				
			bit 4				
			bit 3				
			bit 2				
			bit 1				
			bit 0 (LSb)				
		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	Not used			
			bit 0 (LSb)	Alarm output			

READ-ONLY							
Register	**Description**	**Bytes meaning**			**U.M.**	**Conv**	**Molt**
1281	alarms status 1	MSByte	bit 7 (MSb)	T3 high limit alarm (EH3)	num		1
			bit 6	T2 high limit alarm (EH2)			
			bit 5	T1 high limit alarm (EH1)			
			bit 4	Probe T8 error (E8)			
			bit 3	Probe T7 error (E7)			
			bit 2	Probe T6 error (E6)			
			bit 1	Probe T5 error (E5)			
			bit 0 (LSb)	Probe T4 error (E4)			
		LSByte	bit 7 (MSb)	Probe T3 error (E3)			
			bit 6	Probe T2 error (E2)			
			bit 5	Probe T1 error (E1)			
			bit 4	RTC Battery alarm (E6)			
			bit 3	FLASH data write error (E5)			
			bit 2	EEPROM error(E0)			
			bit 1	Alarm EP1 Power supply alarm			
			bit 0 (LSb)	Alarm EP2 low battery			
1282	alarms status 2	MSByte	bit 7 (MSb)	Bluetooth – Error in date range configuration (Eb3)	num		1
			bit 6	Bluetooth – Error in the printing process (Eb2)			
			bit 5	Bluetooth – Connection module absent (Eb1)			
			bit 4	T8 low limit alarm (EL8)			
			bit 3	T7 low limit alarm (EL7)			
			bit 2	T6 low limit alarm (EL6)			
			bit 1	T5 low limit alarm (EL5)			
			bit 0 (LSb)	T4 low limit alarm (EL4)			
		LSByte	bit 7 (MSb)	T3 low limit alarm (EL3)			
			bit 6	T2 low limit alarm (EL2)			
			bit 5	T1 low limit alarm (EL1)			
			bit 4	T8 high limit alarm (EH8)			
			bit 3	T7 high limit alarm (EH7)			
			bit 2	T6 high limit alarm (EH6)			
			bit 1	T5 high limit alarm (EH5)			
			bit 0 (LSb)	T4 high limit alarm (EH4)			

3.4 DEVICE STATUS

READ / WRITE							
Register	**Description**	**Bytes meaning**			**U.M.**	**Conv**	**Molt**
1536	device status	MSByte	bit 7 (MSb)	enabling of mod. stand-by status 8	num		1
			bit 6	enabling of mod. stand-by status 7			
			bit 5	enabling of mod. stand-by status 6			
			bit 4	enabling of mod. stand-by status 5			
			bit 3	enabling of mod. stand-by status 4			
			bit 2	enabling of mod. stand-by status 3			
			bit 1	enabling of mod. stand-by status 2			
			bit 0 (LSb)	enabling of mod. stand-by status 1			
		LSByte	bit 7 (MSb)	stand-by status 8 1 = stand-by 0 = ON			
			bit 6	stand-by status 7 1 = stand-by 0 = ON			
			bit 5	stand-by status 6 1 = stand-by 0 = ON			
			bit 4	stand-by status 5 1 = stand-by 0 = ON			
			bit 3	stand-by status 4 1 = stand-by 0 = ON			
			bit 2	stand-by status 3 1 = stand-by 0 = ON			
			bit 1	stand-by status 2 1 = stand-by 0 = ON			
			bit 0 (LSb)	stand-by status 1 1 = stand-by 0 = ON			

For asking the modification of one of device status bits, the master has to send into LSByte the requested value for the bit and into MSByte the corresponding bit set to 1. i.e.: for stand-by staus forcing, the master has to send MSByte = 00000001 and LSByte = 00000001.

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)

NOTE

NOTE

PEGO S.r.l.
Via Piacentina, 6/b
45030 OCCHIOBELLO –ROVIGO-
Tel : 0425 762906
Fax: 0425 762905
www.pego.it
e-mail: info@pego.it

Dealer: