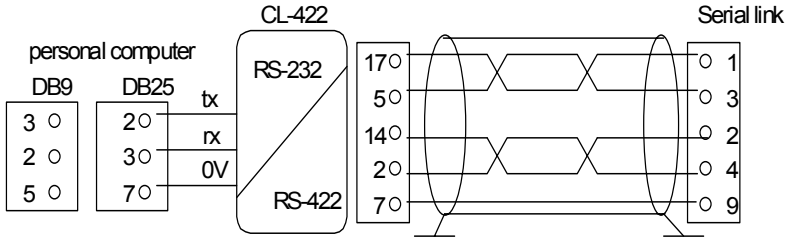
4.2 Programming with MotionWiz

The **sLVD serial kit** is supplied to enable communication between a PC and the drive. The kit includes an RS-422/RS-232 converter, relative 230V~ power supply and serial connection cable. The enclosed communication software (supplied free of charge) designated **MotionWiz** has the following HW/SW requirements: 486 microprocessor or higher, *Windows* 3.1* or more recent version, mouse and serial port for drive connections. The main features of MotionWiz are:

- serial connection of up to 32 drives
- reading and setting of basic parameters and drive commands
- reading and setting of commands and parameters of operating modes
- functional block diagrams
- pico-PLC program displayed as ladder diagrams
- display of pico-PLC program status during operation
- I/O status
- file storage of parameterisation including pico-PLC program
- uploading of parameterisation including pico-PLC program from a file that can be selected from among those previously stored
- oscilloscope function

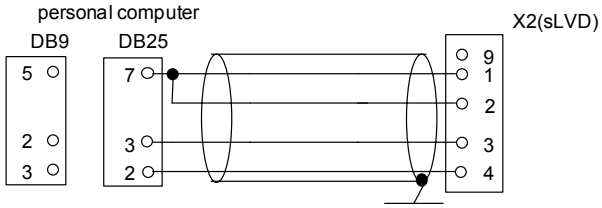
PC - sLVD connection layout (sLVD serial kit):



Refer to text for burden resistors

To create line load (burden) resistances, jumper pins 2 and 6, and pins 4 and 7 on connector X2 of the last drive on the serial line.

If the PC is a battery powered laptop (i.e. not connected to ground) use the following connection layout:



To install MotionWiz open *WINDOWS**, insert the diskette in drive A, select the *File* menu in Program Manager and then the *Run.*option. Now run Setup.exe on [A:\] by typing the following string on the *Command line*: *A:\setup.exe* or by selecting the file with the *Browse...*button. The following installation procedure automatically creates a new icon for MotionWiz. Once the program is installed, launch it by double clicking the relative icon (or select the icon and then press ENTER). Set the parameters for activate the serial line.

- *Windows* and the Windows logo are registered trade marks or trade marks owned by Microsoft Corporation in the United States of America and/or other countries.

5 SERIAL INTERFACE

The serial communication of the converter is half-duplex, master-slave, using an asynchronous RS-485/RS-422 line. The converters take control of the line only if interrogated by the master.

The same serial line can be connected to up to 32 converters by setting a different serial address in each to the Pr27 parameter. It is also possible to set the transmission speed by using the Pr26 parameter as specified in the table below:

Pr26 (decimal base)	b/s	time-out (ms)
0	600	512
1	1200	256
2	2400	128
3	4800	64
4	9600 (*)	32
5	9600	32
6	19200	16
7	38400	12
8	57600	8

For the connection diagrams, see the section, *Connecting the serial line*.

5.1 Communication protocol

The column on the right in the table above shows the time-out value expressed in milliseconds for each communication speed. This is the time within which the message must be sent, beginning from the start of each message (STX). If a message is interrupted after this time, the converter ignores what has been received and waits for the beginning of a new message.

The message consists of several consecutive bits. The format of the bits is the following:

1 start bit
8 bit of data defined by a following byte enclosed within brackets
1 parity bit (even)
1 stop bit

The structure of the message is the following:

[STX] [CMD+ADDR] [BK+LUN] [PAR] [D0]... [Dn] [CHK]

where:

[STX] = $7E indicator of transmission start. If a field in the message different from STX assumes the value $7E, this field is followed by a 0 ($00) so that it will not be interpreted as an [STX].

[CMD+ADDR] = command and address of the peripheral device. This is never 0. This data is composed in the following way: the first 5 bits (bits 0-4) define the address of the converter (from 0 to 31); the remaining 3 bits (bits 5-7) define the type of message sent, as described in the following table:

CMD	bit 7	bit 6	bit 5	type of message
1	0	0	1	converter response
2	0	1	0	reading a pico-PLC instruction
3	0	1	1	writing a pico-PLC instruction
4	1	0	0	reading a parameter
5	1	0	1	writing a parameter
6	1	1	0	bit modification
7	1	1	1	writing a parameter to all the slaves

[BK+LUN] = the LUN field (first 3 bits) indicates the number of bytes of the data transmitted (a parameter or a PLC instruction); the values can range from 1 to 4. This value does not include eventual 0 characters ($00) which are inserted after the values that coincide with the character of transmission start ($7E). The length of each parameter is two bytes.
The BK field is represented by the 5 most significant bits and represents the 5 most significant bits of the parameter address.
[PAR] = write/read address of the parameter or PLC instruction.
The parameter address is the number of the parameter * 2 with 13 bit field: PAR represents the least significant eight bits of the address, the 5 most significant bits must be written in the BK field. The table used by electronic cam starts at the address 4096. The PLC instructions have the address from 0 up to 255.
[D0]... [Dn] = data transmitted.
[CHK] = 256 module sum of all the fields excluding the [STX] (checksum).

Message types
[CMD1] = is the response message of the converter to a data request. The response message has the following format:

[STX] [001+ADDR] [BK+LUN] [PAR] [D0]... [Dn] [CHK]

or it can be the confirmation message to a data write or data modify. In this case, the format is the following:

[STX] [001+ADDR]

where ADDR always identifies which converter is answering.

[CMD2] = is the read message of an instruction in the PLC area. The message has the following format:

[STX] [010+ADDR] [BK+LUN] [PAR] [CHK]

[CMD3] = is the write message to an instruction in the PLC area. The message has the following format:

[STX] [011+ADDR] [BK+LUN] [PAR] [D0]... [Dn] [CHK]

[CMD4] = is the read message of a parameter. The message has the following format:

[STX] [100+ADDR] [BK+LUN] [PAR] [CHK]

[CMD5] = is the write message of a parameter. The message has the following format:

[STX] [101+ADDR] [BK+LUN] [PAR] [D0]... [Dn] [CHK]

[CMD6] = is the change bit message of a byte parameter. The message has the following format:

[STX] [110+ADDR] [BK+LUN] [PAR] [D0] [D1] [CHK]

In this case LUN=2 or else two bytes are sent for the data. The first byte is the mask containing the 0s in the positions of the bits to be changed and 1s in the other positions, while the second byte contains 1s in the positions of the bits that are to be set to 1 and 0s in the other positions. The PAR address is that of the parameter (byte) where one or more bits are to be modified. If the parameter is a word and the bit to be modified is one of the first 8 (b0...b7): PAR = the parameter address; otherwise, if the bit to be modified is one of the upper 8 (b8...b15): PAR = the address parameter + 1.

[CMD7] = is the write message of a parameter to all converters connected to the serial line. The message has the following format:

[STX] [11100000] [BK+LUN] [PAR] [D0]... [Dn] [CHK]

The address of the peripheral device (ADDR) must be 0.

Notes:

- The parameters that are represented on the screen with decimals must be treated as complete values. For example, a value of 978.5 is read and written as 9785.
- All values that are receded by the $ symbol are to be understood as hex numbers.
- The value included inside the brackets identifies the base unit (byte) of the message.
- All messages must be terminated with a time-out which is a function of the speed, well defined to be considered valid and must have the exact parity and checksum.
- The converter responds to a request or to a data send only if the message has been received correctly. In the case of an error in the message, no response is transmitted. The only exception is message type 7 that is used to send data with a single message to all the converters connected to the serial line.

Initializing and managing the serial line

The converter is delivered with a 0 address (Pr27=0) and a speed of 9600 bps (Pr26=5).

To modify the configuration, first set the speed in Pr26, then the serial address in Pr27 and finally initialise it by issuing the command b42.3. Use the b99.15 command to store the configuration.

Each pico-PLC instruction occupies 2 or 4 bytes whose format is the following:

Since the maximum length of each instruction in 2 bytes and the total area available in the PLC is 256 bytes, the PLC program can have at the most 128 instructions.

Instruction	Code	Length (bytes)
LD Pa.y	0	2
LDN Pa.y	1	2
OUT Pa.y	2	2
OUTN Pa.y	3	2
AND Pa.y	4	2
ANDN Pa.y	5	2
OR Pa.y	6	2
ORN Pa.y	7	2
ADD Pa, Pb, Pc	8	4
SUB Pa, Pb, Pc	9	4
MUL Pa, Pb, Pc	10	4
DIV Pa, Pb, Pc	11	4
SET Pa.y	12	2
RES Pa.y	13	2
FIN Pb40.y/Pb150.y	14	2
END	15	2

The first 4 bits (b0..b3) of the first byte in each instruction contain the instruction code.

In the first 8 instructions in the table (LD… ORN) and the SET and RES instructions, the remaining 4 bit of the first byte (b4..b7) contain the value y, while the second byte contains the value Pa.

In the ADD, SUB, MUL e DIV instructions, the second byte contains the value Pa, the third byte the value Pb, and the fourth byte the value Pc.

In the END instruction, the second byte is not used.

In the FIN instruction, the fifth bit (b4) of the first byte selects the parameter: b4=0 if it refers to Pb40, b4=1 if it refers to Pb150; the sixth bit (b5) of the first byte is used for logical negation: b5=0 the bit is copied, b5=1 the bit is negated before being copied. The second byte of the FIN instruction contains the value of y.

If the FIN instructions are used, they must be the first instructions of the program and cannot be more than 2. They occupy the addressed from 0h to 3h. If a FIN instruction is inserted beginning at the 4h address or after any other instruction, the FIN instruction does not function and is ignored (NOP).

The instructions must follow each other beginning at address 0h and no byte can be left empty.

There is only one program and it is terminated with the END instruction.

Examples of using the serial line

In order to better understand how to implement the communication protocol on the serial line, some examples of each type of message are given below.

The values indicated are only indicative as examples.

First example: reading a 1 byte parameter

Suppose we want to read the value of the parameter Pr25 (release software) and that its value is 43. Suppose also that the converter has the serial address 0. The message to be sent is the following.

[$7E][$80][$01][$32][$B3]

The converter responds with the message:

[$7E][$20][$01][$32][$2B][$7E][$00]

Second example: reading a 2 byte parameter

Suppose we want to read the reference speed (Pr7) and that its value is 2000. Suppose also that the converter has the serial address 1. The message to be sent is the following:

[$7E][$81][$02][$0E][$91]

The converter responds with the message:

[$7E][$21][$02][$0E][$D0][$07][$08]

Third example: writing a 1 byte parameter

Suppose we want to select operating mode 1 (Pr31). Suppose also that the converter has the serial address 3. The message to be sent is the following:

[$7E][$A3][$01][$3E][$01][$E3]

The converter responds with the message:

[$7E][$23]

Fourth example: writing a 2 byte parameter

Suppose we want to set the rated current to 2.5 A (Pr33). Suppose also that the converter has the serial address 3. The message to be sent is the following:

[$7E][$A3][$02][$42][$19][$00][$00]

The converter responds with the message:

[$7E][$23]

Fifth example: setting a bit to 1

Suppose we want to send the command to save the PLC program (b99.14=1). Suppose also that the converter has the serial address 0. The message to be sent is the following:

[$7E][$C0][$02][$C7][$BF][$40][$88]

The converter responds with the message:

[$7E][$20]

Sixth example: setting a bit to 0

Suppose we want to disable the converter via software (b40.9=0). Suppose also that the converter has the serial address 0. The message to be sent is the following:

[$7E][$C0][$02][$51][$FD][$00][$10]

The converter responds with the message:

[$7E][$20]

Seventh example: writing a PLC instruction

Suppose we want to set the first instruction of the PLC as: LD 90.4. Suppose also that the converter has the serial address 0. The message to be sent is the following:

[$7E][$60][$02][$00][$40][$5A][$FC]

The converter responds with the message:

[$7E][$20]