How to use the serial port debugging assistant to send commands to the servo control board

The servo control board is serial (TTL level) and can be used to control the servo by sending commands to it from any serial device.

to control the servo using any serial device.

Requirements for the serial port:

1. must be TTL level (not 232 level)

2. TXD to RXD, RXD to TXD, GND to GND (inverted, common ground)

3. baud rate of 9600, no parity bit, 8 data bits, 1 stop bit, no control flow.

Command Format Description :

Such as #1P1500T100\r\n

The number 1 is the S1 channel on the servo control board.

The number 1500 is the angle of the control servo (range 500-2500), and the number 100 is the time, which means the time needed to rotate from the current position to the position in the command.

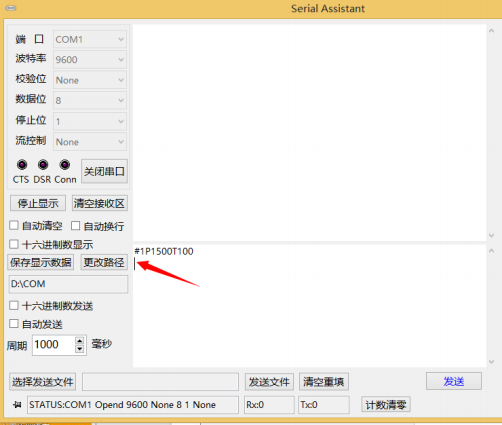
The number 100 is the time, time means the time needed to rotate from the current position to the position in the command.

Lastly, \r\n is a special character that cannot be sent out directly by the serial port debugging assistant. (Note 1)

The above command means: the servo inserted on S1, within 100ms. rotate to 90 degrees from the current position.

rotate to 90 degrees from the present position within 100ms.

**annotate 1**



**In the figure above, we have used our own serial port debugging assistant. Note that the following commands are sent to the following area**

**#1P1500T100**

**followed by a blank line (arrow)**

**Because our correct command is #1P1500T100\r\n , \r\n this is a special character, meaning that the meaning of the carriage return to line feed, if you directly send "#1P1500T100\r\n ", then certainly**

**If you send "#1P1500T100\r\n" directly, then there will definitely be no response because the command is wrong, the reason is:**

**What you mean by this is to take \r\n as four characters, '\',' r ',' \',' n '.**

**Because our serial port debugging assistant will parse the "blank lines" in the send area as "\r\n" in the command.**

**"\r\n" in the command, so it can be used normally.**

**If you must use the serial port debugging assistant downloaded from the Internet, then you must convert all the commands into hexadecimal numbers to send. This can also be used normally (the above is the form of strings sent**

**commands are very intuitive)**

**Hexadecimal number description:**

**#1P1500T100\r\n**

**Converted to hexadecimal number is :**

**23 31 50 31 35 30 30 54 31 30 30 30 0D 0A**

**The last 0D 0A means \r \n.**

**A special note:**

**If you know the difference between a string and a hexadecimal number, and you know what an escape character is, then I**

**can guarantee that you can use any one of the serial port debugging assistant to send commands to our products.**

**But if you do not understand, then for unnecessary trouble, you should use our serial debugging assistant.**

**hand it, the specific method, see page two.**