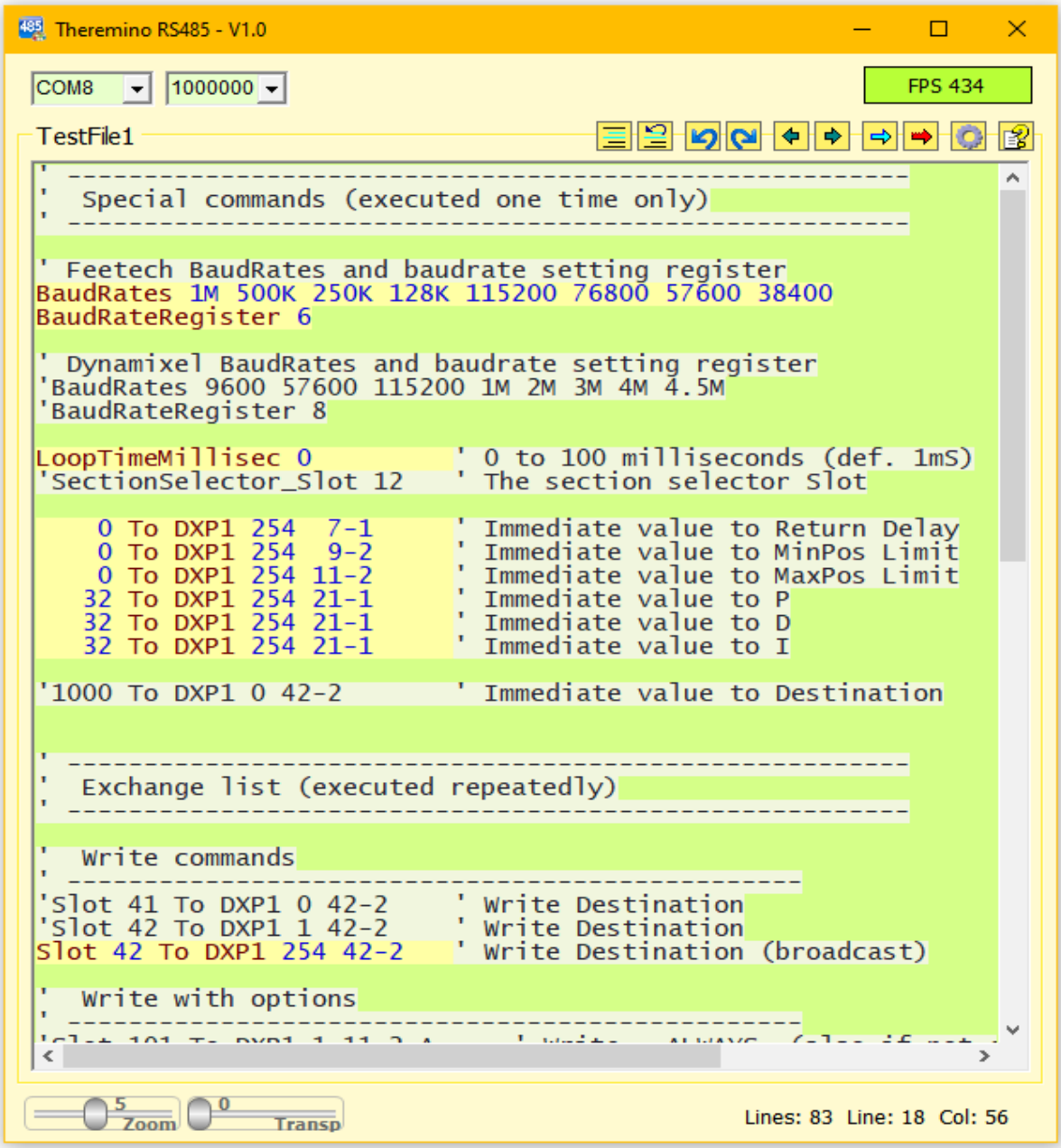


Theremino RS485 V1.3

The Theremino_RS485 application

This application connects the Theremino system Slots with devices connected to a serial line ***Half-Duplex*** (RS485 with two balanced wires, or TTL with a single wire).

The devices are connected in a chain by means of a four-wire cable (two signals plus power supply and GND) or even just three wires in the TTL versions.

The devices that normally connect are ***FeeTech*** and ***Dynamixel***

These servo-motors, also called Robot-Motor or Smart-Motor, contain all the control electronics, a 4096 step encoder and a configurable PID algorithm that allows to control the rotation with 0.09 degrees precision.

You can also adjust the speed, acceleration and torque, as well as read the position reached, the temperature, the current (which is related to the torque) and many other parameters.

Some Dynamixel motors can be controlled in a range of +/- 256 rpm, while currently the FeeTechs ***(Note 1)*** they have a smaller range, of only +/- 7.5 turns.

(Note 1) New FeeTech models with improved torque control are in preparation. True SmartMotors for industrial controls but at a highly competitive price.

Characteristics of servo motors

On this page we report the characteristics of some servomotors that we found particularly interesting.

We tried the top five on this list and definitely recommend the top three (3032, 3215 and 3046) which, in addition to being very economical, also run smoothly and quietly.

Motor	Voltage	Gears	Speed (RPM)	Torque (kg-cm)	Encoder bits	Size (mm)	Weight (g)	Euro	Processor
STS3032	4V-7.4V	205:1	111	4	12	32*12*27	21	20	GD32F130F8P6TR
STS3215	4V-7.4V	345:1	54	19	12	45*25*35	55	10	GD32F130F8P6TR
STS3046	6-7.4V	378:1	52	40	12	40*20*43	89	25	
SM29BL	12-24V	241:1	110	24	12	40*28*42	102	50	STM32F030K6T6
SM45BL	12-24V	353:1	110	25	12	46*28*34	100	90	STM32F030K6T6
SM120BL	9-25V	232:1	50	120	12	*78*43*65*	485	420	
eRob70	48V		40	367	19	70*81	1000	550	

The data in this table are approximate, consider them as a rough indication to roughly compare the motors. The millimeters are rounded to the nearest integer and the prices are those found in Europe. Prices in China are 30..40% lower.

All the motors considered here are coreless or brushless and the encoders are magnetic or optical. We have previously discarded all models with brush motors and potentiometer feedback.

The indicated "torque" value is the stall value. Under normal working conditions it must be kept considerably lower.

Communication protocols

The protocol used by this application is the ***Dynamixel 1.0*** (abbreviated to DXP1) which allows communication with all servomotors ***Dynamixel*** is ***FeeTech***.

There is also a protocol ***Dynamixel 2.0***but we didn't implement it because it would only work with some Dynamixel models (MX and PRO) and with none of the FeeTechs. Furthermore, the 2.0 protocol does not contain any particular advantages. Its only major improvement would be the SyncRead instruction, which is superseded by our implementation of***Buffered transfers***

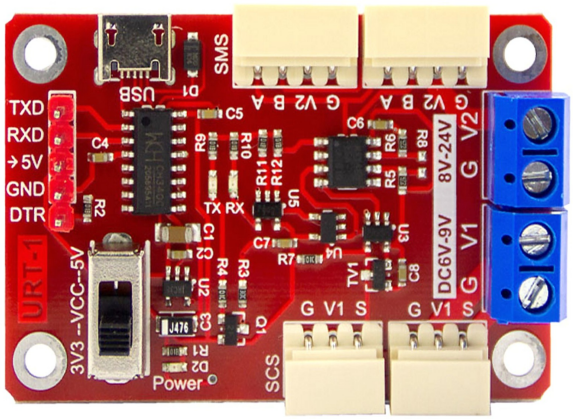
Use DXP1 or Modbus protocols

FeeTech servos can be programmed to use either the DXP1 protocol or the ModBus protocol. The latter is familiar to PLC users, but has lower speed performance and fewer commands.

To use Modbus devices we have written an application very similar to this one, but called ***Theremino_Modbus*** In its page you will also find the related documentation files.

Program FeeTech servo motors for DXP1 or Modbus

We have prepared in ***This Page*** a ZIP archive containing everything needed to reprogram the FeeTech servos with the two protocols. You will also find programming instructions in the ZIP file.



To reprogram the FeeTech motors you will need a USB connection module, like the one in the image on the right, a USB cable and a connection cable for the motors.

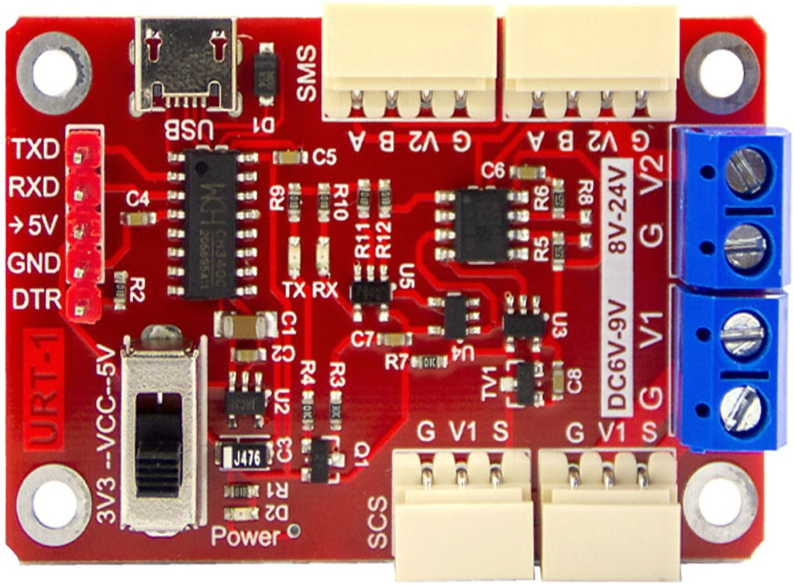
You will also need a 24 volt power supply which connects with the negative to terminal "G" and with the positive to V2.

The power supply must have a convenient power switch located near the PC keyboard and the Mouse. With the switch you have to give power exactly at the same time you press the programming button on the software. If this action is not done synchronously, the programming will not start.

To make the Dynamixels work you could use this same adapter, but not we still tried.

The Feetech URT1 board

With this card you connect the motors to the USB port and power them with the blue connectors on the right.

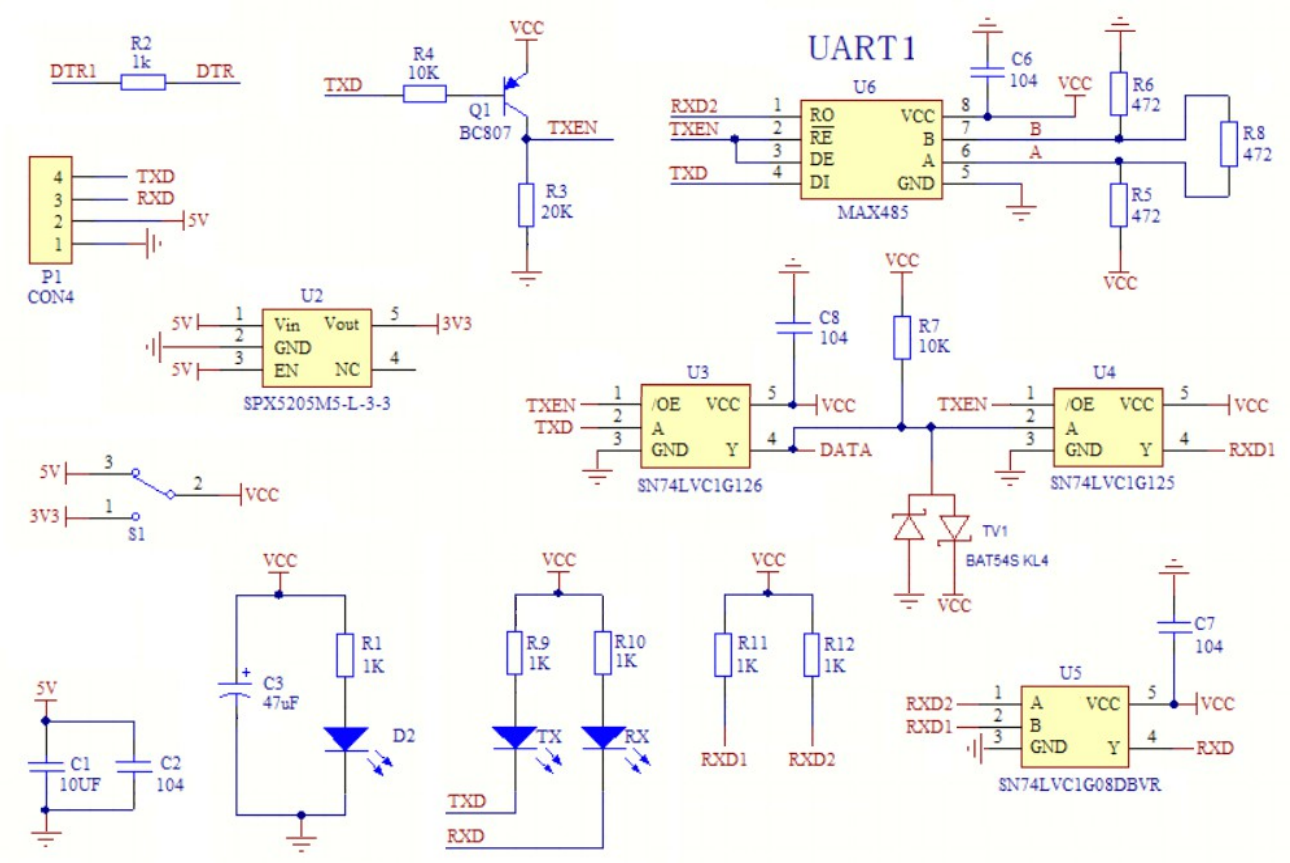


Motors with four wires connect to the top connectors and feed 8 to 24 volts on connectors G and V2

Motors with three wires plug into the connectors below and feed 5 to 7 volts on connectors G and V1

The low voltage motors (we recommend the excellent 3032 and 3215) can be powered by connecting the 5 volts arriving from a second USB port (preferably USB3) to G and V1, or from a 5 or 6 volt power supply and at least 2 or 3 amps.

Wiring diagram



Edit the command list

In the central part of the application there is the list of commands. The colored parts on a yellow background are active and without errors. The parts in light green are commented and do not act.

```
' ========================================================
'  Special commands (executed one time only)
' ========================================================

' Feetech BaudRates and baudrate setting register
BaudRates 1M 500K 250K 128K 115200 76800 57600 38400
BaudRateRegister 6

LoopTimeMillisec 0          ' 0 to 100 milliseconds (def. 1mS)

    0 To DXP1 254  7-1      ' Immediate value to Return Delay
    0 To DXP1 254  9-2      ' Immediate value to MinPos Limit
    0 To DXP1 254 11-2      ' Immediate value to MaxPos Limit
   32 To DXP1 254 21-1      ' Immediate value to P
   32 To DXP1 254 22-1      ' Immediate value to D
   32 To DXP1 254 23-1      ' Immediate value to I


' ========================================================
'  Exchange list (executed repeatedly)
' ========================================================

' ----------------------
'  Write with SYNC option
' ----------------------
Slot 11 To DXP1 0 42-2  S   ' Write Destination but WAIT the SYNC command
Sync To DXP1                ' Exec all the SYNC commands

' ----------------------
'  Read with buffer
' ----------------------
DXP1 0 56-15 To Buffer      ' Read 15 bytes (56 to 70) to buffer
Buffer 56-2 To Slot 100     ' Position    from buffer to Slot
Buffer 58-2 To Slot 101     ' Velocity    from buffer to Slot +4000 -4000
Buffer 60-2 To Slot 102     ' Torque      from buffer to Slot  +900 -900
Buffer 62-1 To Slot 103     ' Voltage     from buffer to Slot
Buffer 63-1 To Slot 104     ' Temperature from buffer to Slot
Buffer 64-1 To Slot 105     ' Sync Flag   from buffer to Slot
Buffer 65-1 To Slot 106     ' Hard.Error  from buffer to Slot
Buffer 66-1 To Slot 107     ' Moving      from buffer to Slot
Buffer 69-2 To Slot 108     ' Current     from buffer to Slot +40 -40
```

Everything written in this list becomes immediately operational. Every time even a single character is changed, the whole list is checked again and the parts without errors are immediately activated.

For which be careful when editing commands not to accidentally create commands that write to registers other than those desired.
Or disable the communication as explained on the next page.

```
' ----------------------
'  Write with SYNC option
' ----------------------
Slot 11 To DXP1 0 42-2  E   ' Write Destination but WAIT the SYNC command
Sync To DXP1                ' Exec all the SYNC commands
```

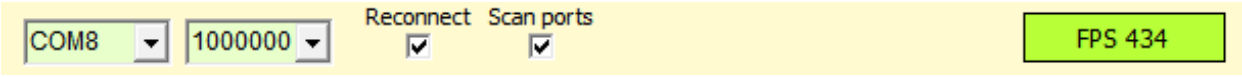
In case of errors the command turns red. Here you see a line containing an error, the "E" option which does not exist.

Disable communication

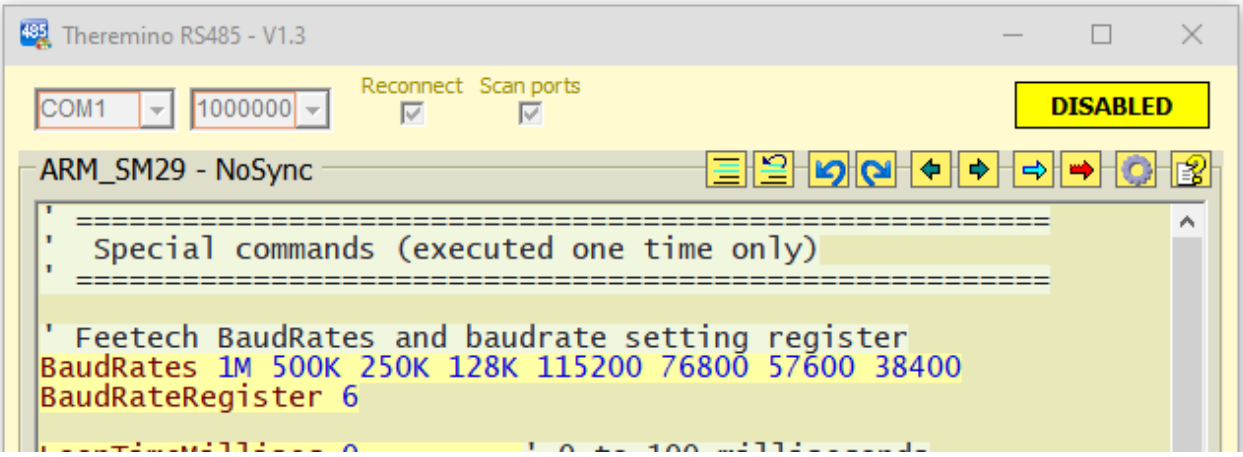
Everything you write in the command list becomes immediately operational. Every time even a single character is changed, the whole list is checked again and the parts without errors are immediately activated.

This is handy for quickly testing the effect of variations, for example when adjusting PID values. But if you are not careful it could happen that you write to other registers without meaning to. For example, if you write to the identifier register, then the engines will no longer work and you will have to disconnect them and reset their identifier one at a time.

Therefore, during substantial changes, it is advisable to disable communication. You could remove power from the motors but it is more convenient to press the button at the top right with the mouse that indicates the communication speed in FPS.



When the button is pressed, the message "DISABLED" appears on a yellow background and the communication port commands are disabled.



The command list also changes color to highlight the disablement.

After finishing the changes, press the button again which will return to mark the speed on a green background.

Command List (DXP1 Protocol)

Initialization commands
Run once, at startup and when editing program text

```
BaudRates nnn, nnn, nnn          ' BaudRate List
BaudRateRegister r               ' Register to set the BaudRate
LoopTimeMillisec nnn             ' Delay to slow down the execution loop
SectionSelectorSlot s            ' Setting the Sections Slot
nnn To DXP1 d r-b                ' "Immediate" number to a register
```

Write commands
Performed continuously, as often as possible

```
Slot s To DXP1 d r-b             ' Read the Slot and write in a register
Slot s To DXP1 d r-b     TO      ' Always write (A = Always)
Slot s To DXP1 254 r-b           ' Write to all devices (254 = Broadcast)
Slot s To DXP1 d r-b     W       ' Send but wait for the Action command
Slot s To DXP1 d r-b     W       ' Send but wait for the Action command
Action To DXP1                   ' Execute all previous (Wait) commands
Slot s To DXP1 d r-b     S       ' Accumulate and wait for the Sync command
Slot s To DXP1 d r-b     S       ' Accumulate and wait for the Sync command
Sync To DXP1                     ' Send all commands accumulated with Sync
```

Read commands
Performed continuously, as often as possible

```
DXP1 d rb To Slot s              ' Read 1 to 4 bytes and send them to a Slot
DXP1 d rb To Buffer              ' Read from 1 to 253 bytes and send them to buffer
Buffer rb To Slot s              ' From 1 to 4 bytes from the buffer to a Slot
Buffer rb To Slot s              ' 1 to 4 bytes from the buffer to another Slot
```

Sections setting command, also executed continuously

```
Section nnn                      ' Start marker of a section
```

Meaning of abbreviations

nnn = Number
s = Slot (1 to 999)
d = Device identifier (0 to 253) (254 = "Broadcast")
r = Device register (0 to 253)
b = Number of bytes that make up the register (from 1 to 4)

Special commands

These commands are used to initialize the device with fixed values, they are mainly used to adjust the speed of the serial port "Baud Rate", the minimum and maximum movement limits and the PID parameters.

These commands are sent only once when the application is started and are sent again each time any character in the program is changed.

These transfers they do not affect the number of exchanges per second (FPS) because they act only once and with very short times. Therefore, you don't have to worry about using them carefully and limiting their use to the essentials as you should do with all other transfers.

Examples of special commands

```
BaudRates nnn, nnn, nnn          ' BaudRate List
BaudRateRegister r               ' Servo register to set the Baud Rate
LoopTimeMillisec nnn             ' Delay to slow down the execution loop
SectionSelector_Slot s           ' Setting the Sections Slot
```

Among the special commands, which are executed only once, there are also the transfers of the "Immediate" numbers, which will be explained in more detail on the next page.

Example of "Immediate" transfer

```
nnn To DXP1 d r-b                ' "Immediate" number to a register
```

Meaning of abbreviations

nnn = Number
s = Slot (1 to 999)
d = Device identifier (0 to 253) (254 = "Broadcast")
r = Device register (0 to 253)
b = Number of bytes that make up the register (from 1 to 4)

Log tables

Each device has different characteristics and even the control tables are not all the same. Therefore it is always advisable to consult the documentation of the individual devices.

FeeTech Servo Motors

Fortunately FeeTech servo motors have only one table that applies to everyone. The differences in the behavior of the individual motors are of little importance and the registers are always the same.

So we were able to collect all the tables in a convenient PDF file that you download from ***This Page***

All the examples you will find on the next pages
use the FeeTech engine register scheme.

For Dynamixel servos the examples have to be adapted
to the tables of the single engine, correcting the numbers of the registers.

Dynamixel Servo Motors

We haven't checked them all but Dynamixels from the same group (XL, XC, XM, XH, XW, AX, EX, DX, RX, MX, PH, PM, L, M, H) should have almost identical tables.

Dynamixel documentation is convenient and well specified for each engine, so we have not prepared any documentation on them.

Go to consult the characteristics and registers to use for each engine, you can find them all on this page:

https://emanual.robotis.com/docs/en/dxl

On the left there is a menu with the categories of the engines, click to open them and inside you will find all the engines with their images.

Then clicking on the single engine opens a page that contains everything about it, technical specifications, registers to use and advice for communication.

Immediate transfers

These commands are used to initialize the device with fixed values, they are mainly used to adjust the "Return-Delay" response time, the minimum and maximum movement limits and the PID parameters.

Immediate transfers are sent only once when the application is started and are sent again whenever any character in the program is changed.

These transfers they do not affect the number of exchanges per second (FPS) because they act only once and with very short times. Therefore, we do not have to worry about using them carefully and limiting their use to the essential, as we must do with all other transfers.

Examples of immediate transfers

```
0 To DXP1 254 7-1          'Immediate value to Return Delay
0 To DXP1 254 9-2          'Immediate value to MinPositionLimit
0 To DXP1 254 11-2         'Immediate value to MaxPositionLimit
32 To DXP1 254 21-1        'Immediate value to P
32 To DXP1 254 21-1        'Immediate value to D
32 To DXP1 254 21-1        'Immediate value to I
500 To DXP1 254 48-2       'Immediate value to MaxTorque
```

Note that in these examples the target device is 254 ie "Broadcast" which sends the same value to all connected devices. With the "Broadcast" method you save from repeating the instruction for all devices and you get the added benefit of not having to change the program even if you add or remove devices.

- - -

With an immediate transfer you could also control the destination of a single servo motor. This normally doesn't make much sense but it could be useful, in some cases, to make sure that the motor is positioned at a predetermined point when starting.

Example of command that initializes the zero device position

```
1000 To DXP1 0 42-2        'Immediate value to Destination
```

Transfers

These commands transfer numerical values from the Theremino system Slots to the device registers and vice versa.

The following example reads the numerical value of Slot 12 and writes it in four consecutive bytes starting from address 42 of device 3.

Slot 12 To DXP1 3 42-4

- The first part Slot 12 defines the Slot (could be from 0 to 999)
- The word To indicates direction (from Slot to DXP1)
- DXP1 defines a device that uses the Dynamixel V1 protocol
- The number after DXP1 defines the device. In this example the device is the 3 but it could be from 0 to 253.
- The number 254 cannot be used to indicate the devices because it would specify a "broadcast" sending, ie a simultaneous sending to all connected devices.
- The number 255 cannot be used because it is used by the protocol as a transmission start signal.
- The last part 42-4 indicates to write 4 consecutive bytes starting from address 42.

Examples of transfers from a Slot to a device

Slot 22 To DXP1 0 22-2 From Slot 22 towards bytes 22 and 23 of device zero.

Slot 22 To DXP1 254 22-2 From Slot 22 towards "254" ie all connected devices.

Slot 22 To DXP1 0 22-2 A Sending with special option "A" (Always)

Examples of transfers from a device to a Slot

DXP1 2 22-4 To Slot 22 Bytes 22, 23, 24 and 25 of device 2 to Slot 22

DXP1 3 40-2 To Slot 22 Bytes 40 and 41 of device 3 towards Slot 22

Buffered transfers

The transfers that slow down the communication the most are the log readings.

Usually the registers you are interested in are many, for example Position, Speed, Motion Indicator, Temperature, Voltage, Current, etc ... If you read all these registers one by one you have to wait for the answer each time and this can slow down the speed in an unacceptable way. And if there are more than one servos, you can even reach a few exchanges per second and the movements become fragmented, with continuous stops and starts.

When writing registers this problem can be solved by using the SYNC method, which sends data to numerous registers without waiting for responses, but the Dinamixel protocols do not provide efficient methods for reading the registers.

So we've added the ability to read several bytes with a single response packet. The bytes are accumulated in a buffer from which the data of the individual registers can be extracted, at very high speed and without spending time in communication.

To extract the data of a register from the buffer, the address of the register and its size in bytes (1, 2, 3 or 4 bytes) must be indicated

Buffered read example

```
DXP1 0 56-15 To Buffer          'Read 15 bytes (56 to 70) to buffer

Buffer 56-2 To Slot 100         'Position    bytes 56 and 57 from buffer to Slot 100
Buffer 58-2 To Slot 101         'Velocity    bytes 58 and 59 from buffer to Slot 101
Buffer 60-2 To Slot 102         'Torque      bytes 60 and 61 from buffer to Slot 102
Buffer 62-1 To Slot 103         'Voltage     bytes 62       from buffer to Slot 103
Buffer 63-1 To Slot 104         'Temperatures bytes 63      from buffer to Slot 104
Buffer 64-1 To Slot 105         'Sync Flag   bytes 64       from buffer to Slot 105
Buffer 65-1 To Slot 106         'Hard.Error  bytes 65       from buffer to Slot 106
Buffer 66-1 To Slot 107         'Moving      bytes 66       from buffer to Slot 107
Buffer 69-2 To Slot 108         'Current bytes 69 and 70 from buffer to Slot 108
```

"Broadcast" programs

You can send commands with the same value, to all connected devices, and these commands are called "Broadcast" -

These commands save lines of code as well as transmission time. They are also commands that do not require a response, so they are practically instantaneous.

The "Broadcast" commands are used mainly for initializations, but can also be used during the continuous exchange of information, for example to change the same setting on all devices at the same time.

The "Broadcast" commands are just write commands, because reading from many devices at the same time would cause the data to collide.

To send a "Broadcast" command to all devices, the special number 254 is used instead of the device identifier.

Examples of broadcast transmissions

```
0 To DXP1 254 7-1        'Immediate value to Return Delay
0 To DXP1 254 9-2        'Immediate value to MinPos Limit
0 To DXP1 254 11-2       'Immediate value to MaxPos Limit
32 To DXP1 254 21-1      'Immediate value to PID-Proportional
32 To DXP1 254 22-1      'Immediate value to PID-Derivative
 32 To DXP1 254 23-1     'Immediate value to PID-Integral
```

With this series of instructions, which is executed only once at start-up, all connected devices are initialized in the same way.

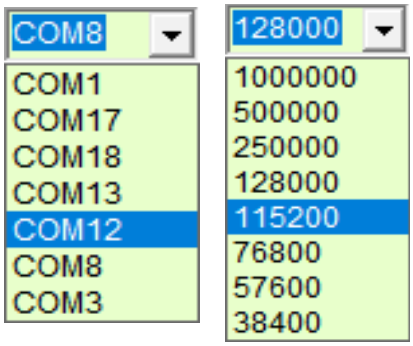
Example of a broadcast transmission that moves all the servos

```
Slot 100 To DXP1 254 42-2
```

This instruction moves the destination of all connected servos. By varying the value of Slot 100 all servos will move together.

COM port settings

With the two boxes at the top left you can choose the serial port and the communication speed.



Usually the maximum speed is used but if the cable is very long it can be limited.

To locate the port you disconnect and reconnect the USB cable and each time you close and reopen the box on the left.

If the port does not work, or is already in use, the two boxes turn red.

To make the servo and application understand each other it is essential that the following two setting lines are at the beginning of the program and that they are correct.

Settings for FeeTech servos:

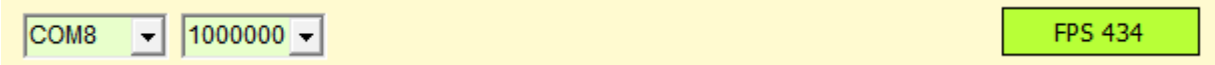
```
BaudRates 1M 500K 250K 128K 115200 76800 57600 38400
BaudRateRegister 6
```

Dynamixel Servo Settings:

```
BaudRates 9600 57600 115200 1M 2M 3M 4M 4.5M
BaudRateRegister 8
```

The line `BaudRates` must contain **exactly** all the speeds that the servos accept and the speeds must be in the right order, otherwise the servos will be programmed with the wrong speed and will not work.

At each start, and each time any character in the program is changed, the communication speed is sent again to all servos. So the box on the right should turn green and indicate good FPS speed.



If the box on the right does not turn green, or if it flashes and indicates very low FPS, then you will need to check the communication lines with the servos and perhaps also program the identifiers of the servos, as explained on the following pages.

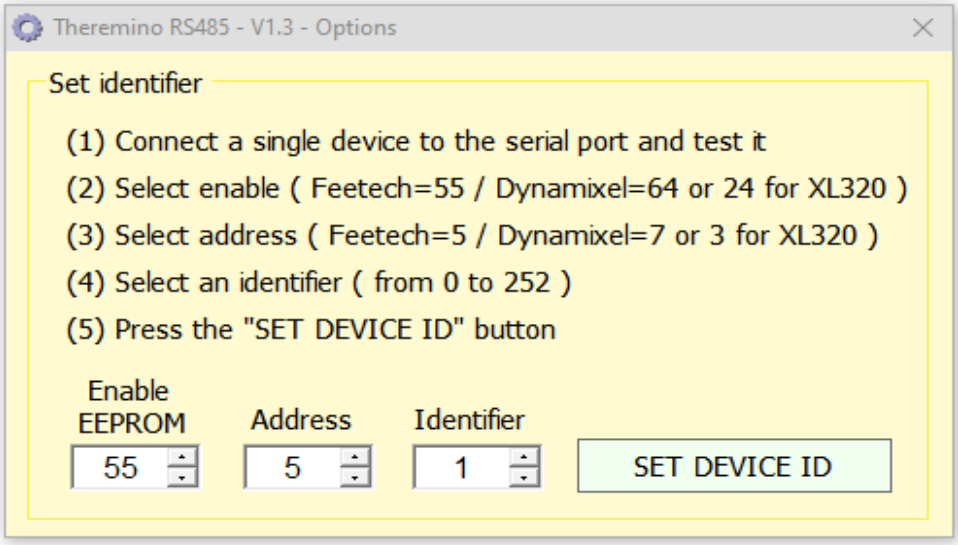
Note that at least one command that expects a response packet must be used, otherwise the text "Inactive" will appear instead of the FPS

Identifier setting

If you connect two or more devices on the same line and communication errors occur with the same ID. In these cases the communication speed indicator may indicate very low FPS and cause the word "Disconnected" to blink.

Therefore to use more than one device it is necessary to prepare them by connecting them one at a time and assigning each of them a different Identifier.

- Open the options panel using the gear tool located at the top right.



- Set the register to enable writing to the EEPROM. In FeeTech servos it is 55, while in Dynamixel it is 64, or 24.
- Set the address (Address) of the identifier. In FeeTech servants it is 5, while nei Dynamixel is 7, or 3 (see servo specifications).
- Connect a single device e make sure it is communicating. Possibly open and close the serial port box and also try to move it using a Broadcast (254) command with option A, such as this: **Slot 1 To DXP1 254 42-2 A**
- Choose an identifier from 0 to 252 to assign to the device, making sure to use a different number for each of them.
- Press the SET DEVICE ID button.
- Repeat for all devices.

Enable writing of the EEPROM

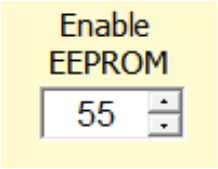
Both FeeTech and Dynamixel servos have a memory location in which to write a zero to enable writing to the EEPROM. Normally the value of this location is "1" and the EEPROM is blocked.

If the EEPROM is blocked then when some parameters are changed it only acts in the RAM memory and this setting is then lost as soon as the supply voltage is turned off.

Commands that automatically enable EEPROM

The commands of the previous two pages, "Setting the communication speed" and "Setting the identifier", automatically enable the EEPROM and disable it at the end.

To make the previous two commands work, the "Enable EEPROM" box (see previous page) must contain the correct value (55 for FeeTech servos, or 64 or 24 for Dynamixels).



Commands that DO NOT automatically enable EEPROM

The commands that are written in the main list act only in the temporary memory unless they are preceded by a line that enables the EEPROM.

It is advisable to group all the commands to be written in the EEPROM at the beginning and to precede them with an enabling line.

And it is recommended that this group of commands be followed with an EEPROM disable line.

See the example on the next page.

Enable and disable the EEPROM

In this example we see the first lines that are normally used. These lines set all engines to speed up communication (zero Return Delay and no Return Packets). In addition, in these lines they also ensure that the range of motion is as desired (in this case from 0 to 4095). Other commands could be added between these lines.

```
    0 To DXP1 254 55-1      ' Zero to enable EEPROM writing
    0 To DXP1 254  7-1      ' Zero to Return Delay
    1 To DXP1 254  8-1      ' ReplyMode=1 (NoReturnPackets)
    0 To DXP1 254  9-2      '       0 to MinPos Limit
 4095 To DXP1 254 11-2      ' 4095 to MaxPos Limit
    1 To DXP1 254 55-1      ' 1 to disable EEPROM writing
```

It is important to note that the first line **0 To DXP1 254 55-1** enables writing to the EEPROM and that the last line **1 To DXP1 254 55-1** disables it again.

It is important to disable the EEPROM after the last command that has to write in it. This reduces the risk of changing important parameters by mistake. For example if you changed the motor ID then you would have to disconnect all motors and reset the IDs one by one.

This is an initialization line so you put it between the first lines of the program, indeed it is advisable to place it right at the beginning so that you can see it well and be sure that it is there.

This example is valid for all FeeTech servos that have the EEPROM lock register in position 55.

For Dynamixel servos, instead, it is necessary to consult the characteristics of the individual devices. In the next few lines we have collected the register numbers of some Dynamixel servos. However we recommend that you always check the characteristics table of your motor.

Dynamixel servos using register 24: *XL-320, AX-12, AX-18, EX-106 DX-113, DX-116, DX-117, RX-10, RX24F, RX-28, RX-64. MX-12W, MX-28, MX-64, MX-106*

Dynamixel servos using register 64: *XL-430, XC430, XM430, XM540, XH430, XH540*

Finally, it is necessary to pay attention that the Dynamixels use the same memory location both to enable the EEPROM (with the value zero) and to enable the engine itself (with the value one). Therefore, after setting the identifier, it will also be necessary to reset the value "1" to make the motor work again.

The Reply Mode

FeeTech servos have the "Replay mode" option which is useful for speeding up communication.

To use it, you must first enable writing to the EEPROM and then send a "ReplyMode = 1" command which disables the sending of confirmation packets.

In the next sequence we see the commands that are normally sent to the servos to initialize them.

```
    0 To DXP1 254 55-1      ' Zero to enable EEPROM writing
    0 To DXP1 254  7-1      ' Zero to Return Delay
    1 To DXP1 254  8-1      ' ReplyMode=1 (NoReturnPackets)
    0 To DXP1 254  9-2      '    0 to MinPos Limit
 4095 To DXP1 254 11-2      ' 4095 to MaxPos Limit
    1 To DXP1 254 55-1      ' 1 to disable EEPROM writing
```

It is always advisable to leave the writing on the EEPROM commented to avoid accidentally changing the IDs of the devices.

In fact, it could happen to send by mistake a command with identifier 254 (therefore broadcast) on the ID register and then write identical IDs on all the servos. In this case you will have to disconnect all the cables and reset the IDs by connecting one motor at a time.

The line that deletes the return packets is as follows:

```
    1 To DXP1 254  8-1      ' ReplyMode=1
```

By eliminating them, communication speeds of 130 FPS can be achieved with four servos, and 60 FPS with an entire COBOT arm with six motors plus gripper.

Secondary identifier

The secondary identifier (Shadow) does not exist on FeeTech devices, while on Dinamixels it is located at address 12.

The secondary ID can be from 0 to 252, if you set it with 253 or a higher number its function is disabled.

You can configure the same secondary ID on many devices and with this configuration you can send identical messages to a group of devices.

With the secondary IDs you get the same result that you would get by sending a "Broadcast" message (with id 254), but with the difference that you can send the message to a group of devices instead of all.

Speed up communication

When you send a message to a secondary ID the devices do not respond and you do not have to wait for the reply packet. So you can use this method to speed up communication and you can do it even when you have to control a single device.

With FeeTech devices you can get the same advantage (no waiting for the answer) by setting "ReplayMode = 1", as explained on the previous page or by sending the commands with the "S" (Sync) option, as explained on the next page.

Consult the Dynamixel documentation

Read more about secondary IDs on Dynamixel device features pages, for example this: ***Secondary ID***

The options of the Slot To DXP command

You can change the behavior of type commands **Slot x To DXPn x xx-x** adding some letters (A, S or W) at the end of the command. You cannot combine S and W with each other, only A with S or A with W. So the valid options are: A / S / W / AS / AW

Option A ("Always" in English)

The commands that read the value from the Slots and send it to the devices are executed only if the Slot value changes by at least one unit, but using the A option they are always executed. The communication speed decreases but a continuous exchange of data is obtained, useful in some cases to test communication.

Example:

```
Slot 1 To DXP1 1 11-2 TO        'ALWAYS (also if not changed)
```

Option W ("Wait" in English)

Commands ending in W are sent immediately to devices, but with a warning not to execute them. They will then be executed all together when the "Action" command is sent to the devices.

Example:

```
Slot 2 To DXP1 0 42-2 W         'Send but WAIT the ACTION command
Slot 3 To DXP1 1 42-2 W         'Send but WAIT the ACTION command
Action To DXP1                  'Execute all the commands with the WAIT option
```

Option S ("Sync")

The commands ending with S are not sent, but accumulated in a list and sent to the devices only when the "Sync" command is executed. Consecutive lines with the Sync option must have the same address and the same length.

The advantages of this option are to synchronize operations and speed up communication, because you do not have to wait for a response. You can also use the Sync option to send single commands (a single line and then immediately the Sync line).

Example:

```
Slot 2 To DXP1 0 42-2 S.        'Accumulate and send with the SYNC command
Slot 3 To DXP1 1 42-2 S.        'Accumulate and send with the SYNC command
Sync To DXP1                    'Execute all the commands with the SYNC option
```

The Sections and the Sections Slot

Using the sections may be necessary when the connected motors are numerous, say more than four, or maybe a dozen, or even a hundred. In these cases the communication speed may drop so much that it causes obvious problems. If you go below 30 FPS the movements become erratic and you can experience rocking or even uncontrollable oscillations.

The writing commands (from the software to the motors) can be almost instantaneous, so they can always be sent to all the motors at each communication cycle. Furthermore, these commands are sent only if the value changes (check on the previous pages which commands to use to obtain the maximum speed).

On the other hand, the reading commands are very expensive in terms of time. While using the best commands, each engine must respond and it takes at least a few milliseconds to do so. Therefore, if dozens of motors are interrogated, the communication speed is too low.

To solve this problem we have added the possibility to activate only some parts of the program. It is therefore possible to keep the instructions that move the motors always in operation and only occasionally read the data of the motors or, better still, read them one at a time. This method could therefore be useful for occasionally checking the temperature of the motors or the supply voltage.

To use sections, you establish a Control Slot and set it with the statement **SectionSelectorSlot**. If this instruction is missing, or if it is commented out, or if the Slot number is invalid, then all sections of the program are executed and the instructions**Section** they no longer have an effect.

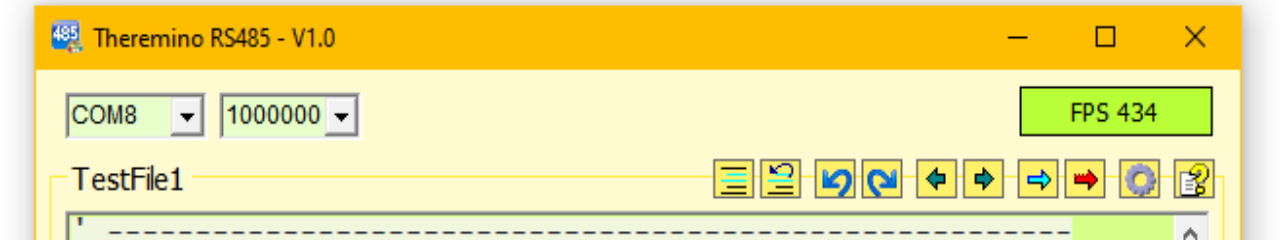
Example of using sections

```
SectionSelectorSlot 11          ' Slot setting for controlling sections
instructions to always follow
Section 1                       ' Section start marker 1
instructions in section 1
Section 2                       ' Section start marker 2
instructions in section 2
```

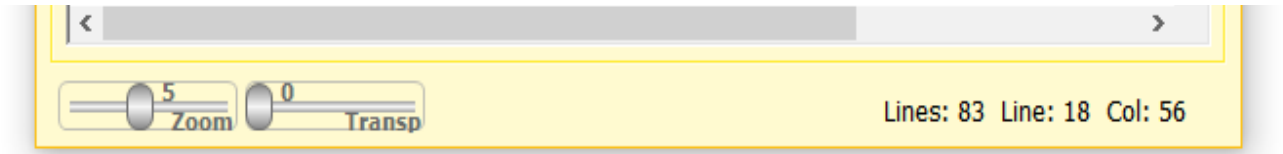
Depending on the current numeric value of Slot 11, only the instructions in the corresponding section are executed.

All statements preceding the command **Section 1**, are always performed. If you want you can write a**Section 0** at the beginning but it is not necessary.

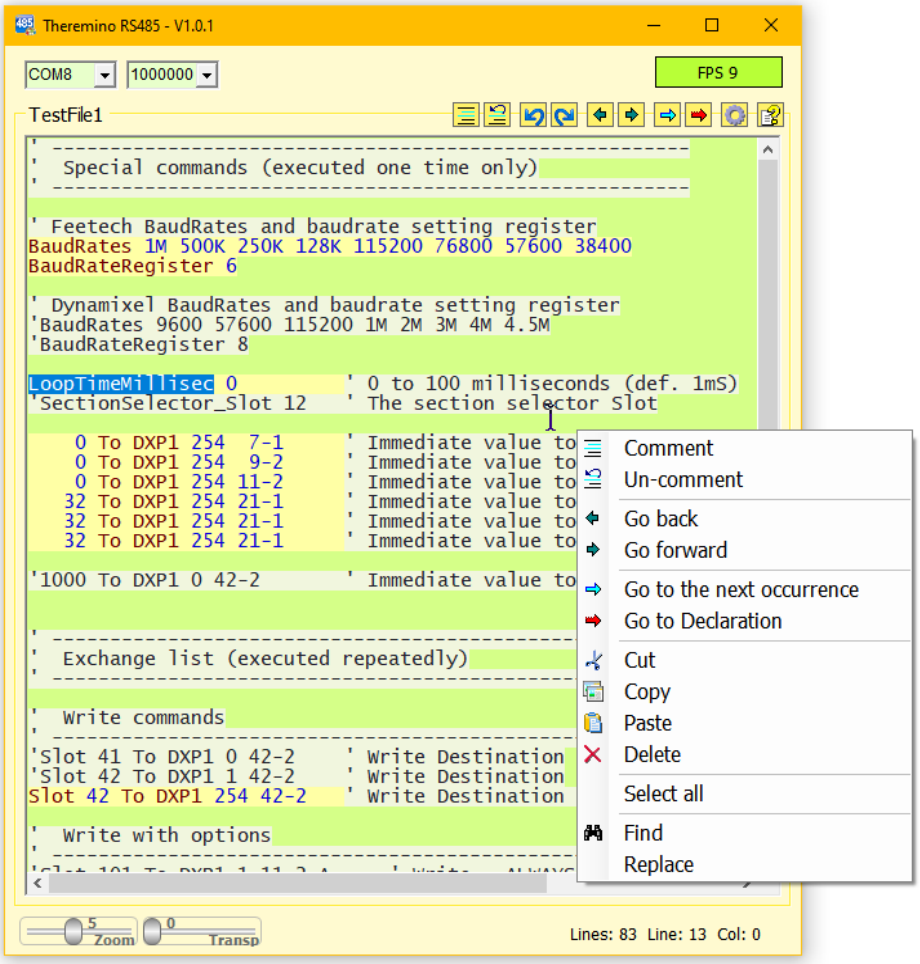
Controls of the application



In the upper area we find the controls for communication and the buttons of the instruments.



In the lower area we find the size and transparency controls and information on the cursor position.



By clicking with the right mouse button on the program area (or by touching the touch screen for two seconds), the menu shown below opens.

These checks will be explained one by one on the following pages.

The controls on the top bar

With the two controls on the left you set the communication port and its speed in Baud (bits per second). With the box on the right you check that no errors occur and you check the communication speed in FPS (Frames per second). "Frame" means sending and receiving all command lines that have been programmed.

With the right settings you should get to write a register and read the response at a speed of 400 ... 500 FPS. In some cases it is possible to read multiple registers towards the Buffer up to 900 FPS and beyond.

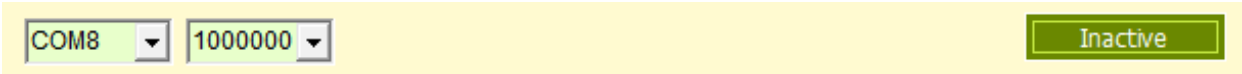
When using multiple motors the speed necessarily drops, but if you use the proper instructions you can check a dozen devices before dropping below 50 FPS and starting to have problems.

Communication errors

Errors are divided into various types:

In this image the two boxes on the left with a red background indicate that the COM1 port is not working.

Here instead we see that the COM8 port is connected but that one or more devices are not responding with valid packets. This can happen if you interrogate devices with wrong ID, or not connected, or without power.

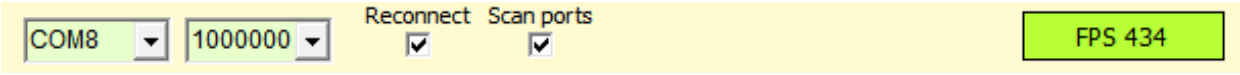


In this case the COM8 port is connected but the word "Inactive" indicates that no return packets are being received.

The "Inactive" condition may also not be due to an error but only to the fact that none of the commands expects a response. This can happen if only Broadcast or Sync write commands are used and no read commands are used.

The automatic reconnection options

Since version 1.2 we have added the possibility to automatically reconnect the COM port and the devices connected to it.



The "Reconnect" option continuously checks that there is communication with the devices and if they do not respond, it closes and reopens the COM port and re-initializes the connected devices.

The "Scan Ports" option also adds the COM port change. The ports are then tested all in sequence, until one is found working and with the devices responding.

Use automatic reconnection with caution

In some cases the devices may be programmed not to respond and none of the communication commands may expect a response.

In these cases the automatic re-connection may believe that there are communication errors and cause continuous re-connections, thus blocking the communication.

Use port scanning carefully

In some cases the "Scan Ports" option may slow down the system boot and go through all the ports before finding the right one.

So if you know the communication port and it never changes, it might be better not to enable the "Scan Ports" option.

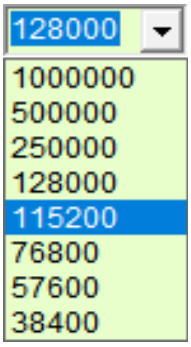
Increase the communication speed

To get smooth movements the number of exchanges per second (FPS) must be at least 20, but if possible it is better to go beyond 50.

Furthermore, if the applications carry out checks and changes in real time, it is good to minimize reaction times and have an exchange rate of at least 100 FPS.

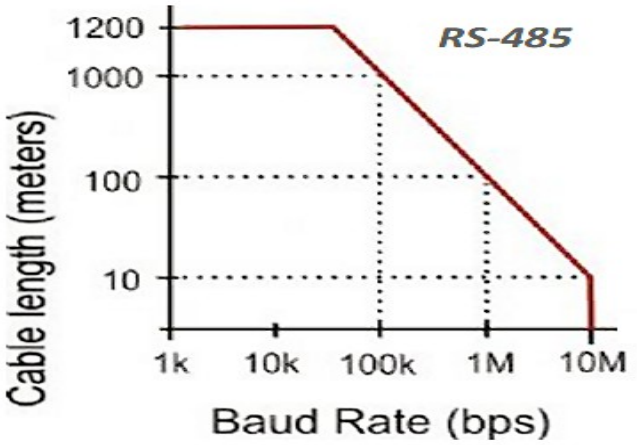
There are no problems when controlling one device, but achieving these speeds with three or more connected devices requires careful programming.

First of all it is better to increase the Baud Rate to the maximum (see previous page "COM Port Settings")



FeeTech servos work up to 1 mega bit and Dynamixel up to 4 mega bit.

The graph on the right indicates to limit the speed only if the cable is over 100 meters long (or over 30 meters in the case of 4 Mb Dynamixel),



Then you have to lower the LoopTime and the Return Delay to zero

Example: **LoopTimeMillisec 0** 'LoopTime = 0

Example: **0 To DXP1 254 7-1** 'Immediate value to Return Delay

Finally, only the communication instructions that allow fast data exchange must be used

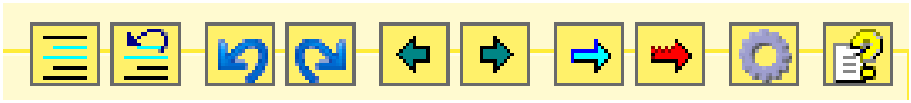
Only instructions with the Sync option will be used to write the registers, which do not include a return packet.

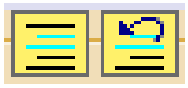
Example: **Slot 2 To DXP1 0 42-2 S.**

Only instructions that read to the buffer will be used to read. These instructions read an entire block of bytes from a device (even many tens of bytes), using a single return packet.

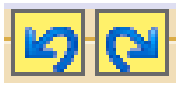
Example: **DXP1 1 50-8 To Buffer**

The toolbar

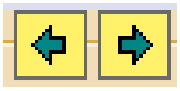




The first two buttons comment and de-comment the selected text.



The two blue arrows they are used to go back in the program changes and to rebuild the deleted changes.

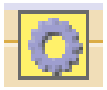


The two dark ARROWS move the cursor, and also the visible page, to the previously visited program sections.

The blue ARROW searches for all occurrences of the selected word, or even just indicated by the text cursor.

The red ARROW only searches for active words (which are not in the commented areas).

The search functions are convenient, just select a word or just place the cursor on it and then press the arrow repeatedly.



The gear opens the options window to assign the identifier to the devices.

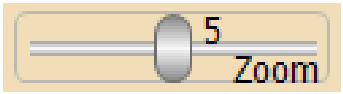
The question mark opens the instruction file (Help) in the chosen language. For this command to work you must copy the Help file of your preferred language into the "Docs" folder.
The latest Help files are downloaded from ***This Page***.

If the Help file is not found then a message appears suggesting to open the Docs folder and copy the file into it.

Or you can choose to select a Help file in your preferred language located in the "Docs" folder or any other folder. *To change the selected file click the button with the right mouse button*.

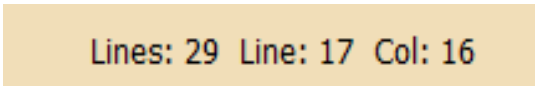
The controls of the lower bar

The cursors are adjustable with the mouse and using the right mouse button return to the default position.



The ZOOM slider determines the size of the text.

This slider adjusts the transparency of the main window and allows you to see below it as well.



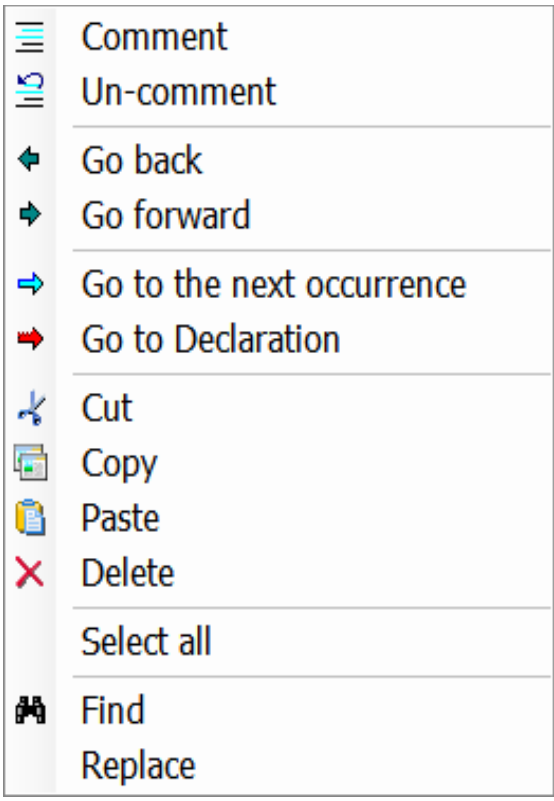
The right part of the bottom bar shows information about the program:

- The total number of lines
- The line where the cursor is (starting from line 1)
- The column where the cursor is (starting from column 1)

The context menu

This menu shows some commands already available with the tool buttons (the buttons at the top right) but integrates them with other useful commands.

By clicking on the program area, with the right mouse button (or by touching the touch screen without removing your finger for two seconds), the menu shown below opens.



Comment is **Uncomment**they are used to comment (add the initial superscript) to entire program areas. Or to delete comments.

Go back is **Go forward** they move the cursor and the visible page to previously visited program areas.

Go to the next occurrence search for other occurrences of the selected word.

Go to declaration search for the selected word but only in the active areas (the parts not commented on).

Cut, **Copy** **is** **Pastes** cut, copy and paste selected parts of the text.

Delete deletes the selected part of the text.

Select all select all text.

Find is **Replace,** they open a window to search and replace words and phrases.

Some of the commands in this menu can also be reached with the keyboard, using the CONTROL key in combination with some letters.

CTRL-X, **CTRL-C**, **CTRL-V** for Cut, Copy and Paste

DELETE for Delete

CTRL-A for Select all

CTRL-F for Find

CTRL-R for Replace