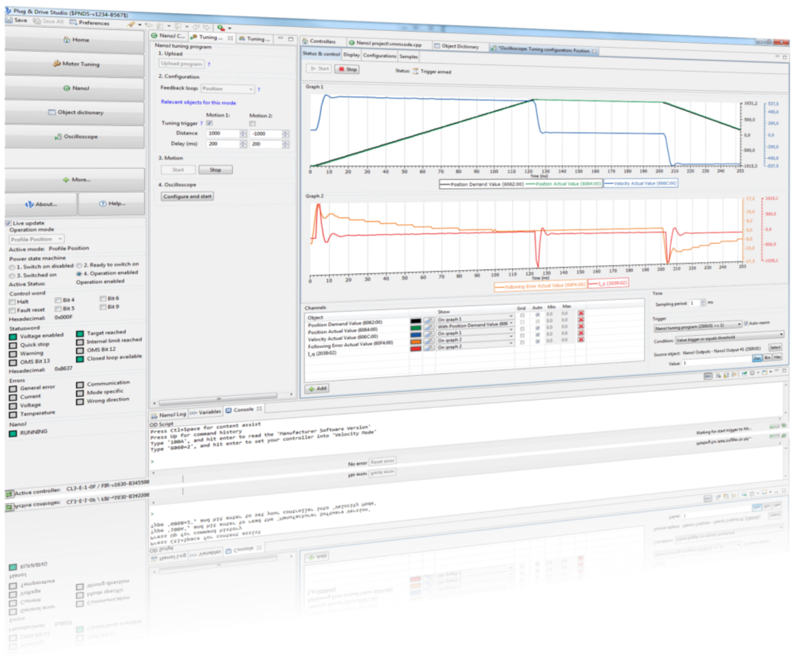
Plug & Drive Studio Quick Start Guide



For Plug & Drive Studio Version 0.10

NANOTEC ELECTRONIC GmbH & Co. KG
Kapellenstraße 6
85622 Feldkirchen bei München, Deutschland

Tel. +49 (0)89-900 686-0
Fax +49 (0)89 900 686-50
support@nanotec.de

Contents

1 Introduction

Plug & Drive Studio is a free software for easy commissioning of the Nanotec controllers, including motor tuning.

If you have questions about Plug & Drive Studio, you can contact us at the following email address:

support@nanotec.de

Please include the keyword "PNDS" in the email subject.

In case of errors, please attach a screenshot and the `.log` file located in the `workspace/.metadata` directory.

2 Prerequisites

2.1 Firmware version

It is recommended to use the firmware version "FIR-v1650-B434164" with this version of Plug & Drive Studio.

The firmware version will be displayed in Plug & Drive Studio when connecting to a controller. For USB controllers, it can be found in the file "INFO.BIN".

2.2 Supported interfaces and products

Plug & Drive Studio currently supports the following interfaces:

CANopen
: Recommended when possible. Currently, only IXXAT adapters are supported.

Ethernet TCP/IP
: Only supported for the N5 EtherCAT (N5-x-1) and CANopen (N5-x-2).

USB Virtual COM Port (USB VCP)
: Requires specific activation, see below for details.

Modbus RTU over RS232 or RS485
: CL3-E only.

Here is an overview for every product:

Product name	Supported field bus
C5-01	USB virtual COM port
C5-E-x-09	USB virtual COM port, CANopen
CL3-E-x-0F	USB virtual COM port, CANopen, RS232, RS485
N5-x-1	Ethernet TCP/IP
N5-x-2	CANopen, Ethernet TCP/IP
N5-x-3	Not supported yet
N5-x-4	Not supported yet
NP5-08	CANopen
NP5-20	Not supported yet
NP5-40	Not supported yet

Product name	Supported field bus
PD2-Cx-E-01	USB virtual COM port
PD2-Cx-E-08	CANopen
PD4-Cx-E-01	USB virtual COM port
PD4-Cx-E-08	CANopen
PD6-Cx-E-09	USB virtual COM port

2.3 Setting up the controller and the communication

It is not recommended to have other applications or devices interacting with your controller while using it with Plug & Drive Studio.

2.3.1 CANopen

Note

- Plug & Drive Studio needs the version **3** of the driver for the IXXAT adapter, versions 2 and 4 is not supported.
- It is recommended to disable all PDOs on the bus to reduce the load.

1. Connect the IXXAT adapter to the computer.
2. Connect the IXXAT adapter to the controller with the appropriate cable. Please refer to the product manual for the details about the CAN connector.
3. Make sure the controller is using the correct CAN settings: node id, baud rate. Please refer to the product manual for more details.

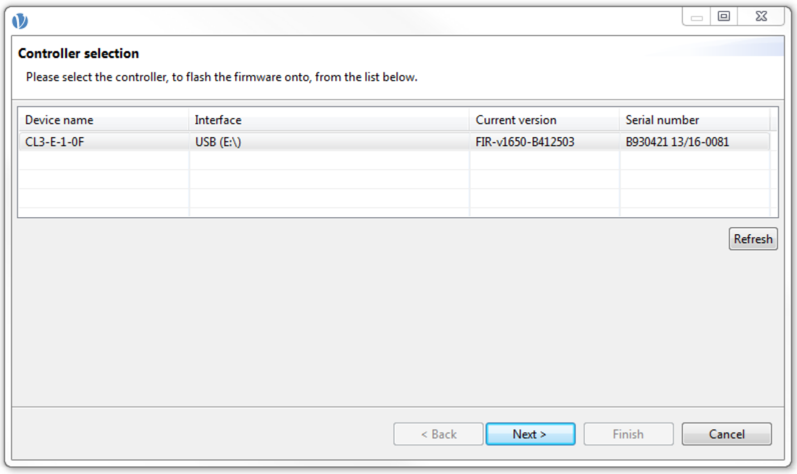
2.3.2 USB mass storage

Most controller used with a clock/direction input are fitted with a USB mass storage connection. Plug & Drive Studio only supports the firmware update, all other features are not available.

If you need to know the firmware verion of such a controller, please follow these steps:

1. Connect the controller to the computer using a USB cable.
2. Open Plug & Drive Studio and move to **Setup** > **Firmware** and click on the button **Flash firmware...** .

   This will open a new window showing the controller name and the firmware version as shown in the picture below.



Controller selection

Please select the controller, to flash the firmware onto, from the list below.

Device name	Interface	Current version	Serial number
CL3-E-1-0F	USB (E:\)	FIR-v1650-B412503	B930421 13/16-0081

2.3.3 USB virtual COM port

Note

You may have problems reconnecting Plug & Drive Studio to the controller after a restart, especially if the restart was not initiated by Plug & Drive Studio (e.g. power cycle). In this case, unplug the USB cable and plug it in again. You may also have to disconnect and reconnect the controller in Plug & Drive Studio. You might want to use the **Restart** button at the top of the "Operation" tab instead of a power cycle.

1. Depending on your operating system, a special driver may be necessary:

   ⚠ WARNING

   Install the USB driver **before** activating the virtual COM port. The driver is shipped in the Plug & Drive Studio release package in the subfolder `Nanotec_USB_VCP_Driver`. Please see the `README.TXT` file there for more information.

   - Windows 7: please install the driver first.
   - Windows 10: no driver is required.
2. Connect the controller to the computer using a USB cable.
3. Open the drive (e.g. F:\) assigned by Windows to the controller in the Windows Explorer.
4. Open the file CFG.TXT or PD4CFG.TXT with a text editor, for instance Windows Notepad.
5. Depending on the product add the following line(s) at the end of the file:

Product name	line(s) to add

C5-01, PD2-Cx-01, PD4-Cx-01	`2102=0x190001`
	`DD4C=1`

CL3	`2102=0x19000F`
C5-E, PD6-Cx-09	`2102=0x190009`

   This will set the bit 20 of the object 2102$_h$:00, enabling the virtual COM port of the controller. The command "`DD4C=1`" will deactivate the internal program for controlling the state machine.
6. Restart the controller.
7. Open the Windows device manager and make sure the controller has been assigned a COM port.

2.3.4 Ethernet TCP/IP

1. Connect the controller to the Ethernet network.
2. Make sure your controller is reachable, for instance with "ping [ip]" on the command line. Please refer to the product manual for more information about the TCP/IP settings.

2.3.5 Modbus over RS232/RS485

Make sure the appropriate COM port is available in the Windows device manager.

1. Connect the controller to the computer using a serial data cable. Please refer to the product manual for the details about the RS connectors.
2. Make sure the controller is using the correct serial port settings like baud rate, parity, modbus address, and so on. Please refer to the product manual for more details.

3 Installing Plug & Drive Studio

⚠ **CAUTION**

We do not recommend using `C:\Program` or `C:\Program Files` as destination folder, because these paths require admin privileges. Since the installer automatically starts the application after it has been extracted, it would start in elevated mode, making the files created by Plug & Drive Studio only accessible to the admin-user.

1. Download the Plug & Drive Studio installer.
2. Run the downloaded installer PNDS-N.N-x86.exe, e.g. PNDS-0.10-x86.exe.
3. Select the destination folder by clicking on **browse**.
4. Click on "Extract" and wait until the unpacking has finished.

Plug & Drive Studio will start automatically after the installation has completed. Later, you can start it via the link on your desktop.

3.1 Uninstalling

CAUTION

Your data - like NanoJ source code, oscilloscope configuration, etc. - is stored in the installation folder, more specifically in the sub-folder called "workspace". Make sure you have copied all the information you need before deleting it. Copying this directory to another installation is supported as long as it contains the same Plug & Drive Studio version.

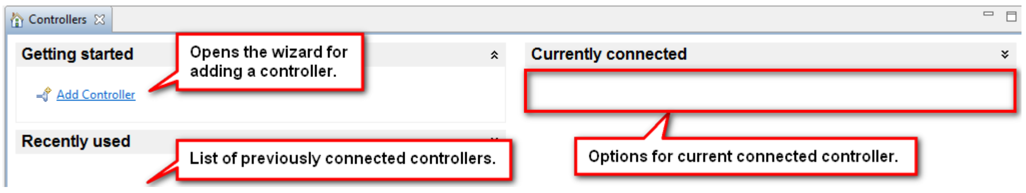
To delete an installation that you don't need anymore, just delete the corresponding folder and the shortcut that Plug & Drive Studio created on the desktop.

3.2 Upgrading

To upgrade Plug & Drive Studio, just install the new version in a new folder. Don't overwrite an existing installation.
Currently, your data (NanoJ source code, oscilloscope configurations) will not be available in the new installation, this feature will be available in a future version of Plug & Drive Studio.

4 First Steps

After the startup of Plug & Drive Studio, you will see the "Home" tab as shown below.


Controllers
Getting started
Add Controller
Recently used
Currently connected
Opens the wizard for adding a controller.
List of previously connected controllers.
Options for current connected controller.

4.1 Connecting to a controller

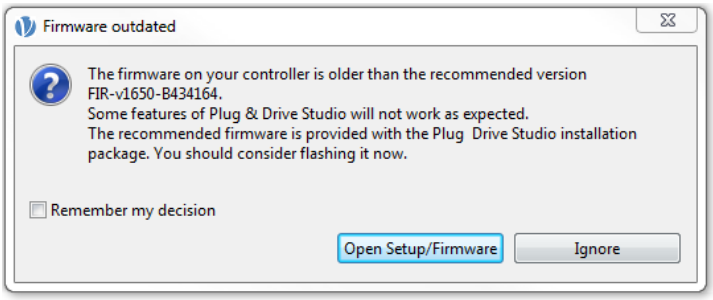
1. Connect the power supply and the cables to the controller if you haven't done so already.

2. Click on **Home** > **Connect Controller** on the "Home" tab. Following this, the wizard **Connect to controller** opens up.
3. Select the relevant interface and click on **Next>**.
4. Change your settings so that they match the configuration on the controller. Click on the **Check Connection** button to test the communication.

   If the connection fails, make sure all cables are properly connected, the device is powered on and that you have selected the correct device and set the appropriate communication parameters.
5. Click on **Finish** when the communication is established.

Note

You may get a dialog as pictured below, reporting that the firmware on the controller is older than the recommended version. We recommend updating the firmware in order to use all the functionality, as described in the chapter Firmware update. **Firmware update**.

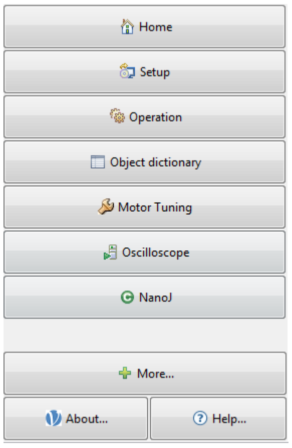

Firmware outdated

The firmware on your controller is older than the recommended version
FIR-v1650-B434164.
Some features of Plug & Drive Studio will not work as expected.
The recommended firmware is provided with the Plug Drive Studio installation package. You should consider flashing it now.

Remember my decision

4.2 Using a controller

In the top left corner of the main window, you can access the main features:



Home (connect and disconnect a controller)

Will open the "Home" tab for connecting or disconnecting a controller. At the moment only one controller can be connected to Plug & Drive Studio at any time. If another controller is to be used, click on "Disconnect", the connection settings of the controller are then stored to disc. The controller is now listed on the welcome page under the section "Recently used".

Setup

Adjust general settings of the controller and the motor. See chapter "**Auto-Setup**" for details.

Operation

Operate the motor in the different modes and adjust the corresponding settings. See chapter "**Operation**" for details.

Object dictionary

Display all objects in object dictionary with their value, and modify them. See chapter **Object dictionary table** for details.

Motor tuning

Tune the motor controller so that it reacts optimally for your application. See chapter "**Motor tuning**" for details.

Oscilloscope

Display the values of the objects in the object dictionary over time. See chapter "**Oscilloscope**" for details.

NanoJ

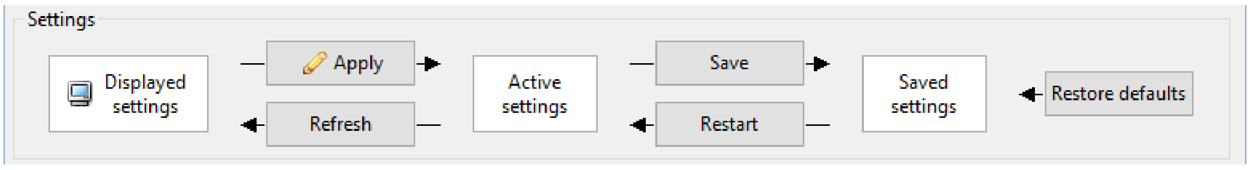
Write programs that are executed on the controller. See chapter "**NanoJ Editor**" for details.

More

All other tools are listed under this menu, for instance a command line to read and write objects in the object dictionary. See chapter "**Object dictionary script console**" for details.

5 Setup

All settings that can be modified in the setup are categorized. Each category is represented by a tab. When you make changes in the graphical interface, the new value is not immediately applied on the controller. Instead, an icon will appear, indicating where changes have been made. The same icon will also appear on the **Apply** button, which is usually what you would press next, unless you want to apply several changes at once. Text fields can also be applied directly, by pressing the return key while they are selected.



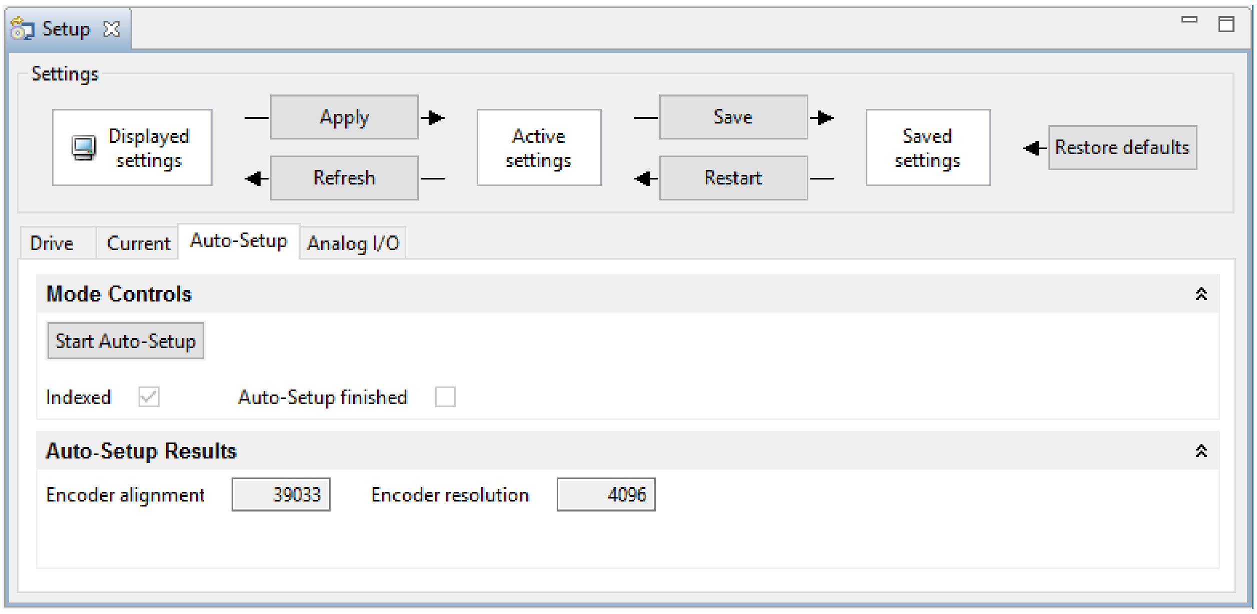
Applying changes means sending the new settings to the controller, making them the active settings. If you are satisfied with the active settings, you can make them permanent by pressing the **Save** button.

5.1 Auto-Setup

CAUTION

Before running the auto setup, make sure all prerequisites for your controller are met. You don't need to perform the auto setup for PD (Plug & Drive) motors, it has been already performed in the factory.

To run the auto setup, open **Setup**, switch to the **Auto-Setup** tab, and press the **Start Auto-Setup** button.



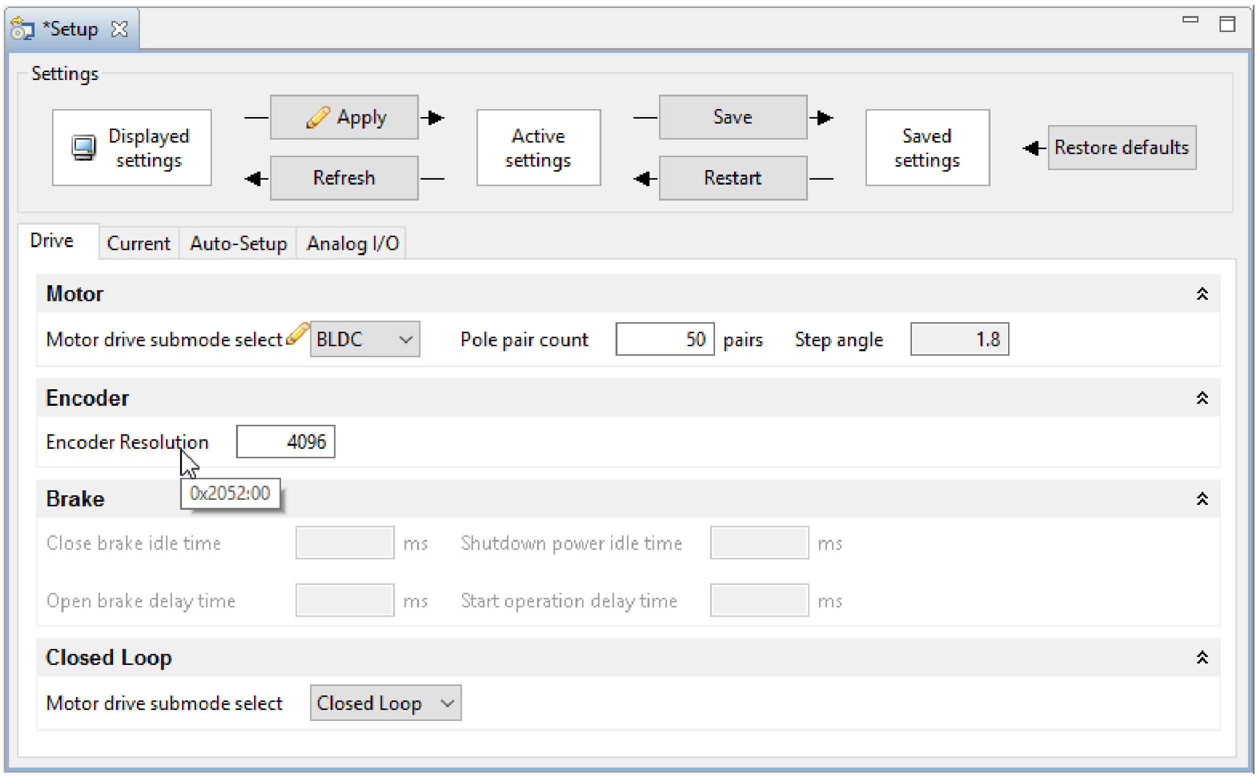
Indexed ✓ Auto-Setup finished ☐

Auto-Setup Results

Encoder alignment 39033 Encoder resolution 4096

5.2 Setup Controls

Each field in the Setup tabs represents an object, or part of an object in the object dictionary. Some of the fields cannot be modified because they are read-only, their value is derived from other actual objects, or because they are currently not relevant. Hovering over the label of a field with the cursor will show you which index and subindex of the object it represents. Additional information about each object can be found in the product manual.



Motor drive submode select BLDC Pole pair count 50 pairs Step angle 1.8

Encoder Resolution 4096 0x2052:00

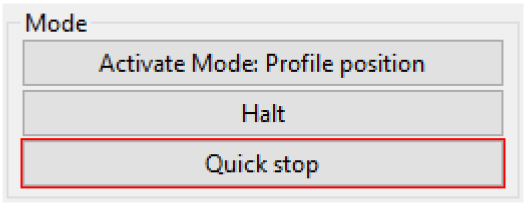
Close brake idle time ms Shutdown power idle time ms

Open brake delay time ms Start operation delay time ms

Motor drive submode select Closed Loop

6 Operation

Configuring the mode of operation of a controller is similar to changing its setup (see chapter **Setup**). However, there are additional controls.



The first button activates the mode that is represented by the currently selected tab, and switches to the operation enabled state in the power state machine.

⚠ CAUTION

Depending on your settings, this can result in immediate motor movement.

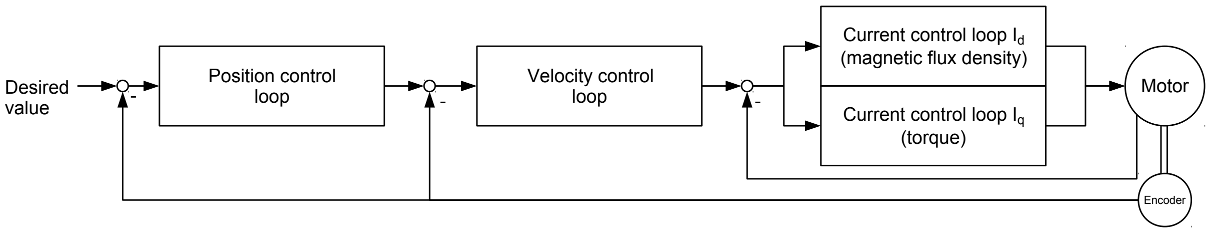
The **Halt** button *toggles* bit 8 in the control word. The **Quick stop** button activates quick stop. Please refer to the product manual for more details.

7 Motor tuning

7.1 Background information

Tuning is the process of finding best possible parameters for the control unit of the motor already mounted in the application.

The control structure of the Nanotec motor controllers consists of three cascading feedback loops (position, velocity, current/torque), each loop being a PI (proportional-integral) controller.


Desired value
Position control loop
Velocity control loop
Current control loop I_d (magnetic flux density)
Current control loop I_q (torque)
Motor
Encoder

Each PI controller is parametrised by:

- a proportional gain (P), which is multiplied by the measured error
- an integral gain (I), which is multiplied by the error accumulated over time

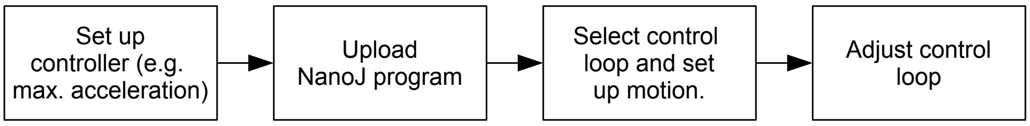
Please refer to the literature for more information about PI controllers.

Tuning is the process of finding the optimal P and I parameters depending on the actual motor, load, and expected behavior (speed vs. precision vs. jerk...). Therefore, it is essential to perform the tuning with the motor mounted in the machine and it moving real load. The reaction of the motor depends also on the rate of change (ramps) of the velocity and torque. Tuning should be done with the steepest ramps required by the application. Obviously, these ramps should be within the range that the motor can actually achieve, based on its data sheet and the available current. You should not set ramps that the motor cannot physically follow. If you do, you will still be able to tune and find the parameters that allow the motor to reach the targets in the minimal possible time by maximizing the torque. However, since the error will still be high, the PI parameter will have to be low, leading to a "soft" regulation, where the controller will be more sensitive to changes of the load compared to a controller set to follow realistic ramps.

Since the input of a control loop depends on the output of the lower-level loop, you should start by tuning the current/torque loop, then continue with the velocity loop and finally with the position loop, if applicable for your application. In other words, if the current/torque or velocity loops are not tuned, it will not be possible to tune the position loop.

7.2 General sequence

The workflow for tuning is as follows:


Set up controller (e.g. max. acceleration)
Upload NanoJ program
Select control loop and set up motion.
Adjust control loop

7.3 Prerequisites

1. Make sure that the controller is connected and available.
2. Ensure that you have read the chapter "**Background information**".
3. Before tuning, Plug & Drive Studio needs to upload a special NanoJ program - any existing program will be overwritten. Therefore, make sure that you have a backup of your own NanoJ program, if any.

7.4 Set up controller

Before starting the tuning, you need to set the relevant objects for the movement itself like "maximum speed", "acceleration", "max. current" and of any such kind.

The link **Relevant objects for this mode** will list some objects that are relevant for the current selected mode. You can also use the "Object Dictionary table" and the appropriate filters (see chapter **Object dictionary table** for more information). For more details about the objects and their values, please refer to the product manual.

7.5 Upload NanoJ program

CAUTION

After uploading a NanoJ program, the controller will restart automatically. This leads to the following effects:

- All unsaved settings will be lost. Plug & Drive Studio will therefore ask before resetting the controller, if *all* objects must be saved first.
- After a restart, at least one full rotation is necessary in order to use Closed Loop. This needs to be done by the user.

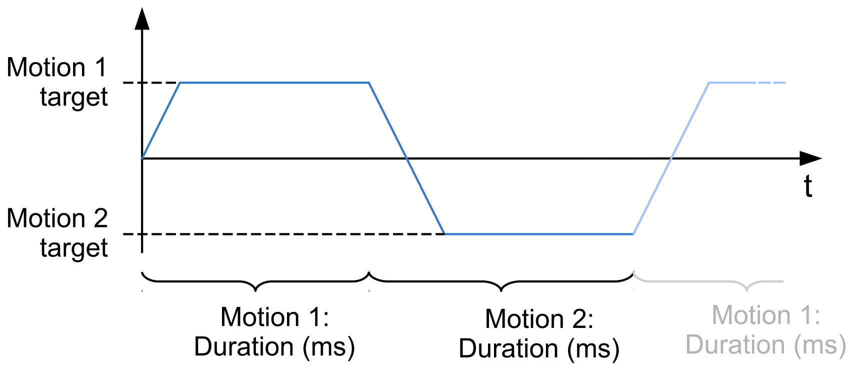
1. Click on the Button **Motor Tuning** to open the view.
2. Click on the Button **Upload program**.
   - If you want to store the objects on the controller, answer the question with **Save and Restart**.
   - If you don't want to save the objects before uploading, answer **Restart**.
3. Wait for the program to be uploaded.

7.6 Setting up motion

1. Select under **Feedback loop** the loop to tune, start with adjusting the Current/Torque loop.

2. Set up the value and the time for the two motions.
3. Click on the button **Start** to start the motion.

   The motor will alternate between the values of motion 1 and motion 2.



Note

The objects

- 6072_h (Max torque)
- 6087_h (Torque Slope)

are set to the value "0" by default and need to be changed. Otherwise the motor will not move when staring the motion for the Current/Torque mode. Also set the object 2031_h (Maximum Current) to the maximum current allowed for this motor.

4. To get a quick setup for the oscilloscope, click on the button **Configure and Start**

Every change in value or duration has effect immediately.

7.7 Adjust control loop

1. Select the tab **Tuning Parameter**.
2. It is suggested to load a default set by clicking on **Presets...** and selecting the current motor type.
3. Use the slider to change the proportional (abbreviated "P") or integral part (abbreviated "I") of the control loop. Sliders labeled with "CL" are responsible for "Closed Loop" control loops and "OL" is used for open loop controls.

7.8 Finding the best parameters

Use the following rules of thumb to find the optimum parameters:

1. You can increase the "P" value until the motor is starting to producing noise. Then reduce by half the value to get a good control loop setting.
2. If the motor is following the desired input parameter too slowly, increase the "P" value.
3. After finding a good "P" value, increase the "I" value to reduce the control error.
4. If the motor is producing overshoots after reaching the desired value, decrease the "I" value.

8 NanoJ Editor

NanoJ is a technology that allows you to run your own program on the controller. The programming language used is C with some extensions.

Note

For a more detailed description of the possibilities of NanoJ please refer to in the product manual.

8.1 General

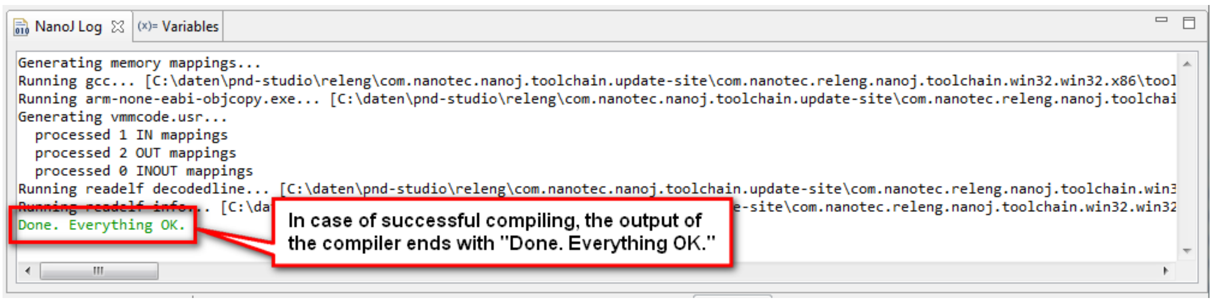
The workflow is as follows:

- Modify the source code, typically vmmcode.cpp.
- Save the source code. The compiler translates (compiles) the files into a format for the controller, the binary file.
- This binary file is then uploaded to the controller.
- Finally, the program is started on the controller.

8.2 Writing a program

The project starts with a small sample program in the file `vmmcode.cpp`. Every time this - or related files - are saved, the compiler will compile the file and will display some information in the console.

The project starts with a sample program in the source file `vmmcode.cpp`. When you modify and save any source file, Plug & Drive Studio will build the project, reporting the result on the NanoJ log. This can also be triggered manually by clicking on the button Build.



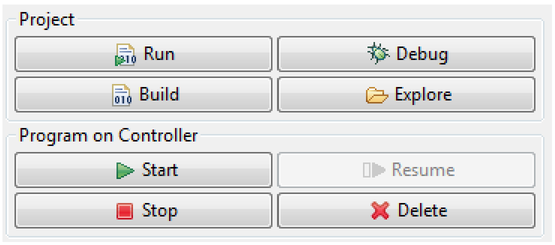
```
Generating memory mappings...
Running gcc... [C:\daten\pnd-studio\releng\com.nanotec.nanoj.toolchain.update-site\com.nanotec.releng.nanoj.toolchain.win32.win32.x86\tool
Running arm-none-eabi-objcopy.exe... [C:\daten\pnd-studio\releng\com.nanotec.nanoj.toolchain.update-site\com.nanotec.releng.nanoj.toolchai
Generating vmmcode.usr...
  processed 1 IN mappings
  processed 2 OUT mappings
  processed 0 INOUT mappings
Running readelf decodedline... [C:\daten\pnd-studio\releng\com.nanotec.nanoj.toolchain.update-site\com.nanotec.releng.nanoj.toolchain.win3
Running readelf info... [C:\da                                   e-site\com.nanotec.releng.nanoj.toolchain.win32.win32
Done. Everything OK.
```

In case of successful compiling, the output of the compiler ends with "Done. Everything OK."

In case of successful compilation, the output of the compiler ends with "`Done. Everything OK.`"

8.3 Running and debugging a NanoJ program

After building, the program can be uploaded and executed by the controller.



Run

The current project wll be written on the controller and started. The NanoJ program on the controller will be overwritten, if any. No breakpoints or variables are checked and debug output will not be collected.

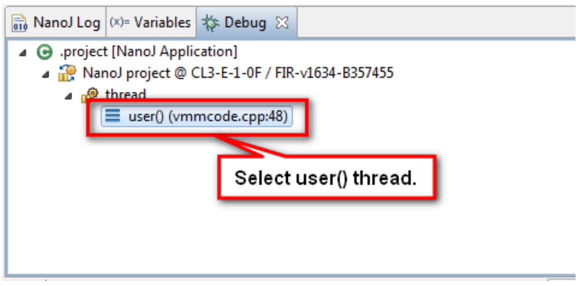
Debug

Same as "Run" but in this mode, the output of the controller via the `VmmDebugOutput` function is collected and printed out in the log line. Furthermore, if a breakpoint is hit, the Editor will stop at the line and show the program variables.

Note

There are some issues with the "Variables" tab in the current version of Plug & Drive Studio:

- The list might be empty, in this case switch to the "Debug" tab and select the user thread as shown below.


NanoJ Log
Variables
Debug
.project [NanoJ Application]
NanoJ project @ CL3-E-1-0F / FIR-v1634-B357455
thread
user() (vmmcode.cpp:48)
Select user() thread.

- In some rare cases the value of the variables are not refreshed.

Build

Rebuilds the project.

Explore

Opens the file explorer at the project location.

Start

Starts the progam already on the controller.

Resume

In case the NanoJ program reached a breakpoint and paused, resumes the execution of the program.

Stop

Stops the execution of the NanoJ program.

CAUTION

When a NanoJ program is stopped, the controller will stay in its current state. If the motor is running, it will continue to do so!

Delete

Deletes the current NanoJ program on the controller. It is not necessary to delete a NanoJ program before using **Run**, it will overwrite any existing program.

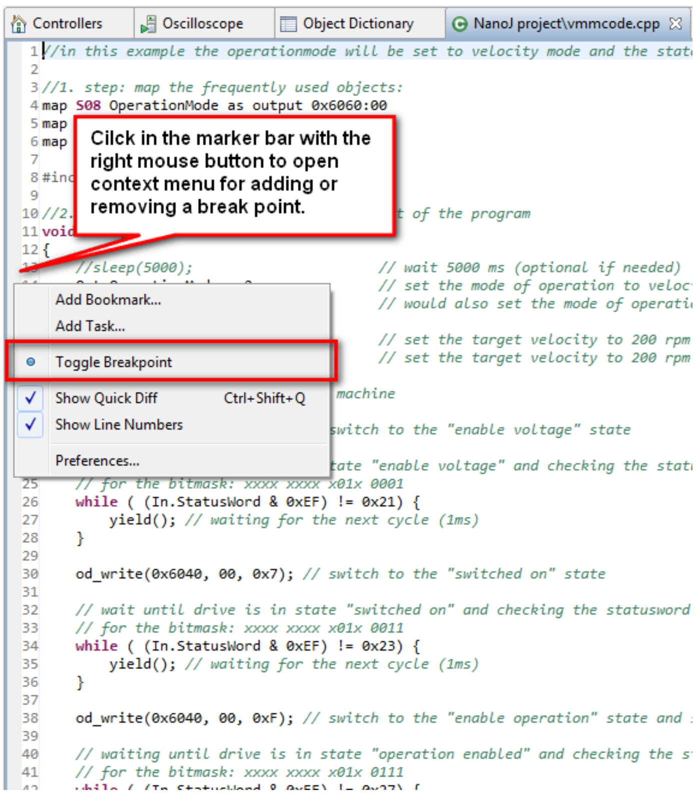
8.4 Additional functions

Breakpoint

A breakpoint can be added or removed by clicking with the right mouse button in the marker bar (see image below) and select the entry "Toggle Breakpoint". When run in Debug mode, the NanoJ program execution will pause when the breakpoint is hit. You can then inspect the variables and resume the execution. Up to three breakpoints are supported."

Note

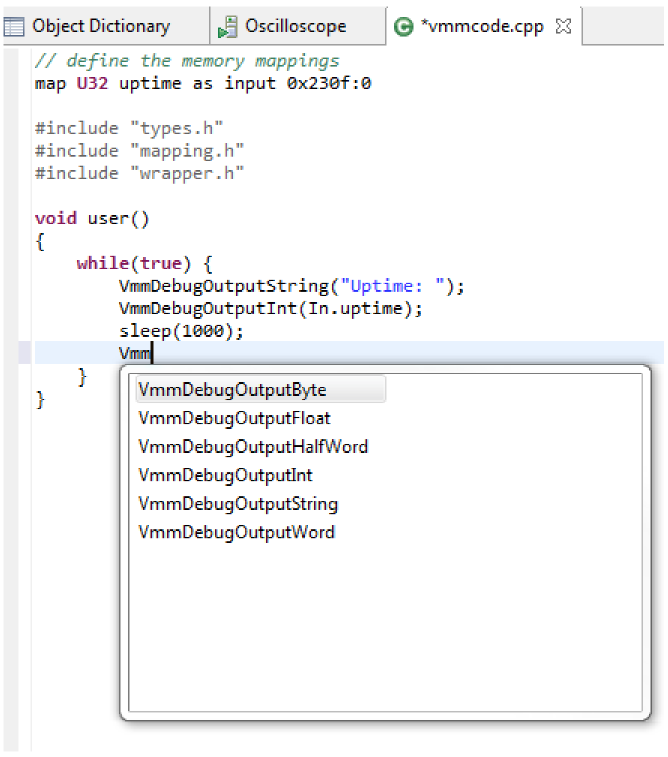
- Run the NanoJ program using "Debug" in order to make Plug & Drive Studio halt at breakpoints.
- Breakpoints can only be set on specific lines, not all.


Controllers
Oscilloscope
Object Dictionary
NanoJ project\vmmcode.cpp
Cilck in the marker bar with the right mouse button to open context menu for adding or removing a break point.
Add Bookmark...
Add Task...
Toggle Breakpoint
Show Quick Diff Ctrl+Shift+Q
Show Line Numbers
Preferences...

```
//in this example the operationmode will be set to velocity mode and the stat

//1. step: map the frequently used objects:
map S08 OperationMode as output 0x6060:00
map
map

#inc

//2.                                                      t of the program
void
{
    //sleep(5000);                                    // wait 5000 ms (optional if needed)
                                                      // set the mode of operation to veloc
                                                      // would also set the mode of operati

                                                      // set the target velocity to 200 rpm
                                                      // set the target velocity to 200 rpm

                                                   machine

                                                  switch to the "enable voltage" state

                                                  tate "enable voltage" and checking the stat
    // for the bitmask: xxxx xxxx x01x 0001
    while ( (In.StatusWord & 0xEF) != 0x21) {
        yield(); // waiting for the next cycle (1ms)
    }

    od_write(0x6040, 00, 0x7); // switch to the "switched on" state

    // wait until drive is in state "switched on" and checking the statusword
    // for the bitmask: xxxx xxxx x01x 0011
    while ( (In.StatusWord & 0xEF) != 0x23) {
        yield(); // waiting for the next cycle (1ms)
    }

    od_write(0x6040, 00, 0xF); // switch to the "enable operation" state and

    // waiting until drive is in state "operation enabled" and checking the s
    // for the bitmask: xxxx xxxx x01x 0111
```

Code completion

Code can be completed by the editor. To start the completion, start typing in the first letters of the code and hit **Ctrl** and **Space**. The context menu offers all known functions, variables and data types fitting the typed letters.



```
// define the memory mappings
map U32 uptime as input 0x230f:0

#include "types.h"
#include "mapping.h"
#include "wrapper.h"

void user()
{
    while(true) {
        VmmDebugOutputString("Uptime: ");
        VmmDebugOutputInt(In.uptime);
        sleep(1000);
        Vmm
    }
}
```

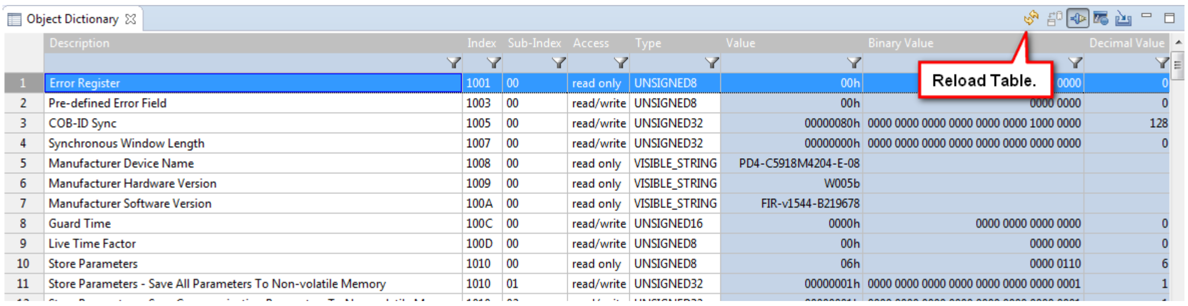
9 Object dictionary table

The object dictionary table lists all object dictionary entries with their names, values and some meta data. You can edit the values in binary, decimal or hexadecimal form.

Only entries with the "Access" set to "read/write" or "write only" are editable.

The value will be written to the controller as soon as you press **Enter** on your keyboard.

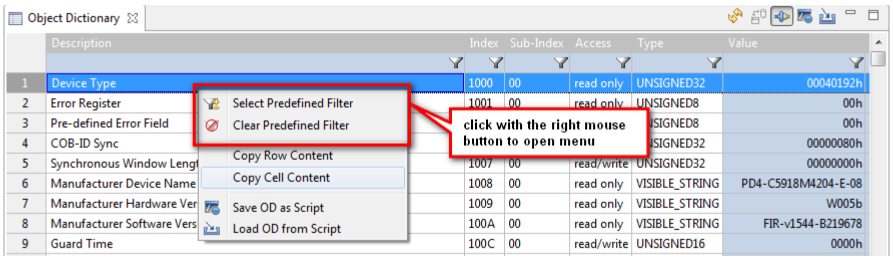
If you want to refresh the whole table, click on the button with the circled double arrow:



	Description	Index	Sub-Index	Access	Type	Value	Binary Value	Decimal Value
1	Error Register	1001	00	read only	UNSIGNED8	00h	0000	0
2	Pre-defined Error Field	1003	00	read/write	UNSIGNED8	00h	0000 0000	0
3	COB-ID Sync	1005	00	read/write	UNSIGNED32	00000080h	0000 0000 0000 0000 0000 0000 1000 0000	128
4	Synchronous Window Length	1007	00	read/write	UNSIGNED32	00000000h	0000 0000 0000 0000 0000 0000 0000 0000	0
5	Manufacturer Device Name	1008	00	read only	VISIBLE_STRING	PD4-C5918M4204-E-08		
6	Manufacturer Hardware Version	1009	00	read only	VISIBLE_STRING	W005b		
7	Manufacturer Software Version	100A	00	read only	VISIBLE_STRING	FIR-v1544-B219678		
8	Guard Time	100C	00	read/write	UNSIGNED16	0000h	0000 0000 0000 0000	0
9	Live Time Factor	100D	00	read/write	UNSIGNED8	00h	0000 0000	0
10	Store Parameters	1010	00	read only	UNSIGNED8	06h	0000 0110	6
11	Store Parameters - Save All Parameters To Non-volatile Memory	1010	01	read/write	UNSIGNED32	00000001h	0000 0000 0000 0000 0000 0000 0000 0001	1

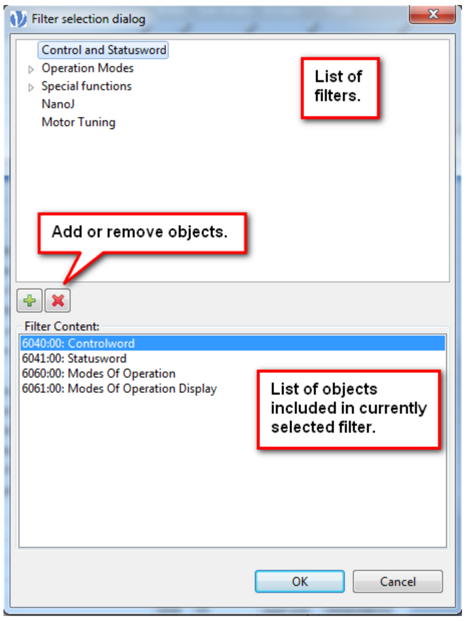
9.1 Filtering

You can filter the objects displayed in the table by clicking with the right mouse button anywhere in the table and selecting the menu entry **Select Predefined Filter**.



	Description	Index	Sub-Index	Access	Type	Value
1	Device Type	1000	00	read only	UNSIGNED32	00040192h
2	Error Register	1001	00	read only	UNSIGNED8	00h
3	Pre-defined Error Field				SIGNED8	00h
4	COB-ID Sync				SIGNED32	00000080h
5	Synchronous Window Lengt	1007	00	read/write	UNSIGNED32	00000000h
6	Manufacturer Device Name	1008	00	read only	VISIBLE_STRING	PD4-C5918M4204-E-08
7	Manufacturer Hardware Ver	1009	00	read only	VISIBLE_STRING	W005b
8	Manufacturer Software Vers	100A	00	read only	VISIBLE_STRING	FIR-v1544-B219678
9	Guard Time	100C	00	read/write	UNSIGNED16	0000h

In the upper part of the resulting window, select a filter that fits your needs. In the lower part, you will get a list of the entries that will be displayed.

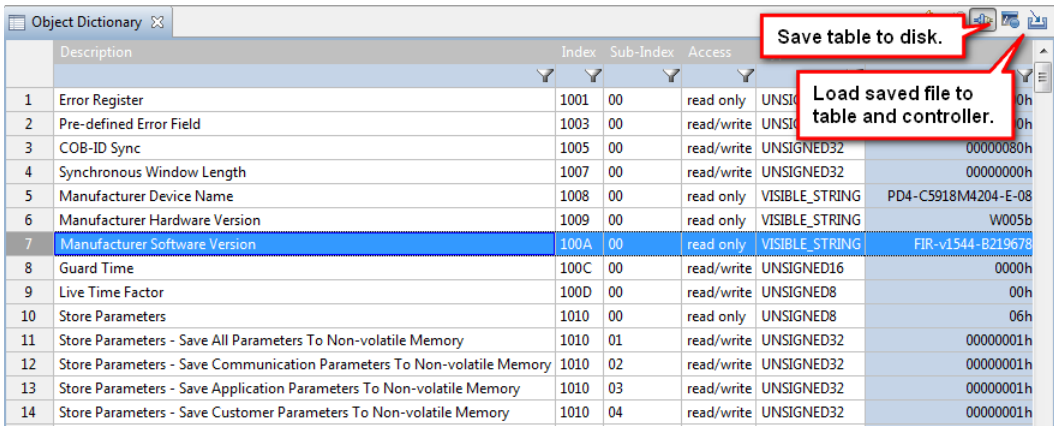


6040:00: Controlword
6041:00: Statusword
6060:00: Modes Of Operation
6061:00: Modes Of Operation Display

To remove the filter on the view, click with the right mouse button anywhere in the table and select **Clear Predefined Filter**.

9.2 Save and load

The whole table can be saved on the PC and loaded back again. The buttons for these actions can be seen in the image below.



	Description	Index	Sub-Index	Access		
1	Error Register	1001	00	read only	UNSI	0h
2	Pre-defined Error Field	1003	00	read/write	UNSI	0h
3	COB-ID Sync	1005	00	read/write	UNSIGNED32	00000080h
4	Synchronous Window Length	1007	00	read/write	UNSIGNED32	00000000h
5	Manufacturer Device Name	1008	00	read only	VISIBLE_STRING	PD4-C5918M4204-E-08
6	Manufacturer Hardware Version	1009	00	read only	VISIBLE_STRING	W005b
7	Manufacturer Software Version	100A	00	read only	VISIBLE_STRING	FIR-v1544-B219678
8	Guard Time	100C	00	read/write	UNSIGNED16	0000h
9	Live Time Factor	100D	00	read/write	UNSIGNED8	00h
10	Store Parameters	1010	00	read only	UNSIGNED8	06h
11	Store Parameters - Save All Parameters To Non-volatile Memory	1010	01	read/write	UNSIGNED32	00000001h
12	Store Parameters - Save Communication Parameters To Non-volatile Memory	1010	02	read/write	UNSIGNED32	00000001h
13	Store Parameters - Save Application Parameters To Non-volatile Memory	1010	03	read/write	UNSIGNED32	00000001h
14	Store Parameters - Save Customer Parameters To Non-volatile Memory	1010	04	read/write	UNSIGNED32	00000001h

The file uses the same format as the Object Dictionary Script Console. So as an alternative for loading the file, it is possible to copy the content - or parts of it - to the Object Dictionary Script Console.

10 Oscilloscope

The oscilloscope samples up to 12 objects of the object dictionary graphically over time.

10.1 Setting up the oscilloscope

You can add channels by clicking on the **Add** button in the lower section.

You can also load a preset, the presets are accessible via **Configurations** > **Load**.

You can also detach the tab "Oscilloscope" from the main window and make it full screen. This is particularly useful if you have a second monitor connected to the computer.

10.2 Starting and stopping the oscilloscope

With the buttons **Start** and **Stop** in the tab **Status & Control** the oscilloscope can be started or stopped.

Note
After hitting the **Start** button, the oscilloscope is waiting for the trigger condition **in the controller** to fire. Until this point, the oscilloscope is left blank.

10.3 Status of the oscilloscope

The status of the oscilloscope is displayed in the tab **Status & Control** and can reach the following states:

Icon	Explanation
Status: ■ Stopped	The oscilloscope is switched off.
Status: ⌛ Trigger armed	The oscilloscope is waiting for the trigger condition in the controller to fire.

Icon	Explanation
Status: ⇩ Fetching data	The oscilloscope is collecting the data from the controller.

10.4 Modifying the scales

Every channel can be scaled in three ways:

Automatically

In case, the check box in the column **Auto** is activated, the channel gets fitted in the graph automatically.

Manually

Deactivate the check box in the column **Auto** and use the numbers in the columns **min** and **max** to set the bounds for the channel.

Use the scaling of another channel

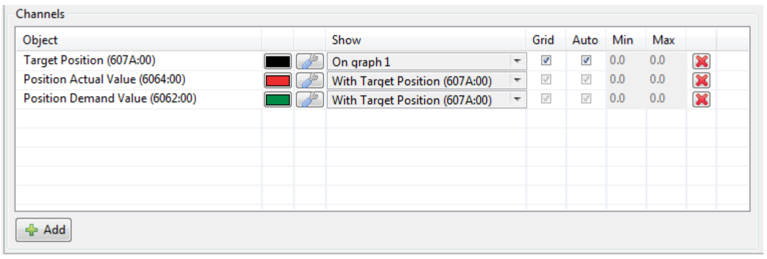
Use the option **With [channel name]** in the column **Show** to assign the scaling from another channel.

Example

The oscilloscope configuration contains three objects which value represents a position:

1. Target position
2. Position actual value
3. Position demand value

To get a better understanding of the correlation between the channels, the "Target position" sets the scale in graph 1 and both other channels will be drawn with the same scale.



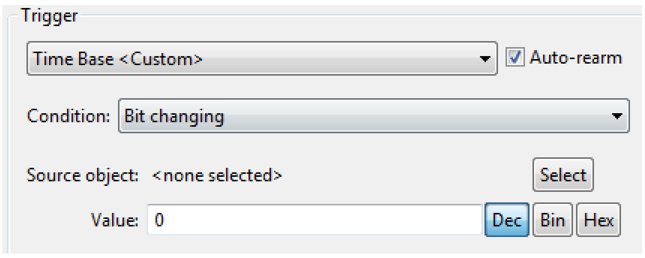
Channels

Object			Show	Grid	Auto	Min	Max	
Target Position (607A:00)	⬛		On graph 1	✓	✓	0.0	0.0	✖
Position Actual Value (6064:00)	🟥		With Target Position (607A:00)	✓	✓	0.0	0.0	✖
Position Demand Value (6062:00)	🟩		With Target Position (607A:00)	✓	✓	0.0	0.0	✖

10.5 Setting the trigger

The trigger defines when the data sampling in the controller starts. There is no data to display until this trigger fires.

You can modify the trigger in the lower right pane:



Trigger

Time Base <Custom>

Auto-rearm: ✓

Condition: Bit changing

Source object: <none selected>

Value: 0

The functions are:

Auto-rearm

If activated, the oscilloscope starts sampling the data every time the condition of the trigger is met. If deactivated, the oscilloscope will sample and display the data only once (often described as "single" mode).

Condition

The trigger condition itself. The "value" or the "bit" refers to the value or bit of the "source object". In case of **Immediate start**, sampling begins when "**Start**" is pressed.

Source object

Use the button **Select** to set the object that the trigger condition applies to.

Value

This is only valid if **Condition** is not set to **Immediate start**. The value will be used for the condition.

Example

To get a precise view of the start of the movement, the sampling is set up to start as soon the speed gets above zero in the "velocity mode":

1. Auto-Rearm: off
2. Condition: Object dictionary value bigger than threshold
3. Source object: "Vl Velocity actual value (6044)"
4. Value: 0

11 Object dictionary script console

11.1 General

The "Object Dictionary Script Console" is a simple command line to read or write values in the object dictionary.

You can open the console with a click an the button **More…** and selecting the entry "Console".

11.2 Read values from objects

To read the value of an object, type in the index and subindex of the objects of interest using the following syntax:

```
<INDEX>:<SUBINDEX>
```

and hit **Enter** key. Index and subindex are hexadecimal numbers (without the notation of "0x"), the index has four digits, the subindex two.

The answer will include a hexadecimal, decimal and binary representation of the number in the entry.

Example

Read out the value of 6040, type in: `6040:00` and hit **Enter**.

The answer may be

```
> 6040
 0x001f 31 0000 0000 0000 0000 (Controlword)
```

where

- `0x001f` is the number in hexadecimal notation
- `31` is the number in decimal notation
- `0000 0000 0001 1111` is the number in binary notation

11.3 Write values to objects

If you want to write in an object, simply write

```
<INDEX>:<SUBINDEX>=<VALUE>
```

and press **Enter** key. The command doesn't return anything in response.

The value can be a decimal number (e.g. "15") or hexadecimal (e.g. "0xF").

Example

To set the object at index 6040 to value "6", type in: `6040:00=6` and press **Enter** key.

11.4 Additional functions

Comments

You can insert comments with two slashes "`//`". The text after the slash is not executed.

Subindex 00

If no subindex is used in a read or write command, the subindex 00 is assumed. You can therefore read the value of the object at index 6040$_h$ like this: `6040`**Enter**. This also works for setting values to objects, so `6040:00=6`**Enter** is equivalent to `6040=6`**Enter**

History

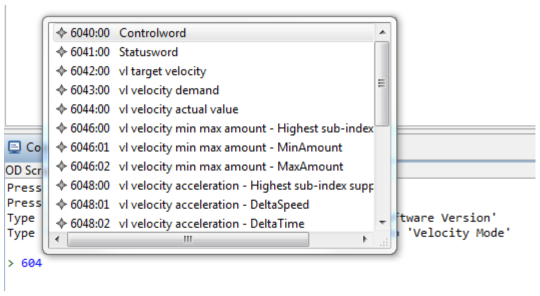
All commands entered are stored in a history. You can navigate through this history with the keys **Arrow up** and **Arrow down**.

Auto completion

The console helps choosing the correct index with pressing **Ctrl** and **Space**.

Example

Start typing in `604` and press **Ctrl** and **Space**. The console will now offer a list of objects starting with the number "604" and the names of the object. Navigate through the list with the keys **Arrow up** and **Arrow down** to select the object and hit enter to transfer the selection to the console.



```
6040:00  Controlword
6041:00  Statusword
6042:00  vl target velocity
6043:00  vl velocity demand
6044:00  vl velocity actual value
6046:00  vl velocity min max amount - Highest sub-index
6046:01  vl velocity min max amount - MinAmount
6046:02  vl velocity min max amount - MaxAmount
6048:00  vl velocity acceleration - Highest sub-index supp
6048:01  vl velocity acceleration - DeltaSpeed
6048:02  vl velocity acceleration - DeltaTime
```

```
OD Scr
Press
Press
Type                 ftware Version'
Type                  'Velocity Mode'

> 604
```

This also works if you need the index of an object where you know the name of an object. Start by typing in the name (for example "Statusword") and press **Ctrl** and **Space** and select the entry of interest as described in the example above.

12 Firmware update

12.1 Getting firmware files

The recommended firmware files are provided Plug & Drive Studio release package. Open the zip file and navigate to the folder Firmware/Firmware FIR-v1650-B434164/. The sub-folders are named after the product, where "x" is used as a placeholder. For instance, the sub-folder "N5-x-2" contains the firmware for the products "N5-1-2" and "N5-2-2".

If you need a different version please contact Nanotec at support@nanotec.de

12.2 Connect the controller

Before updating the controller's firmware, connect it as described in **Setting up the controller and the communication** and **Connecting to a controller**.

CAUTION

Ensure that the power supply for the controller is stable throughout the update process.

12.3 Updating the firmware

1. Click on the button **Setup** and open the tab **Firmware**.
2. Click on the button **Flash firmware…** and follow the instructions.
3. In the next window select the device to upgrade from the list and click on the button **Next >**.
4. In the following window select the firmware to update either from the file system or from the list of recently used firmware files. Then click on the button **Next >**.
5. The next window will show you an overview over the update process.

   CAUTION

   Check the controller model and the firmware file carefully. Using the wrong firmware will lead to a broken controller.

   After that click on the button **Next >**.

   Note

   IIn case of an Ethernet connection, administrator rights may be required and requested. Please confirm the corresponding Windows dialogs.