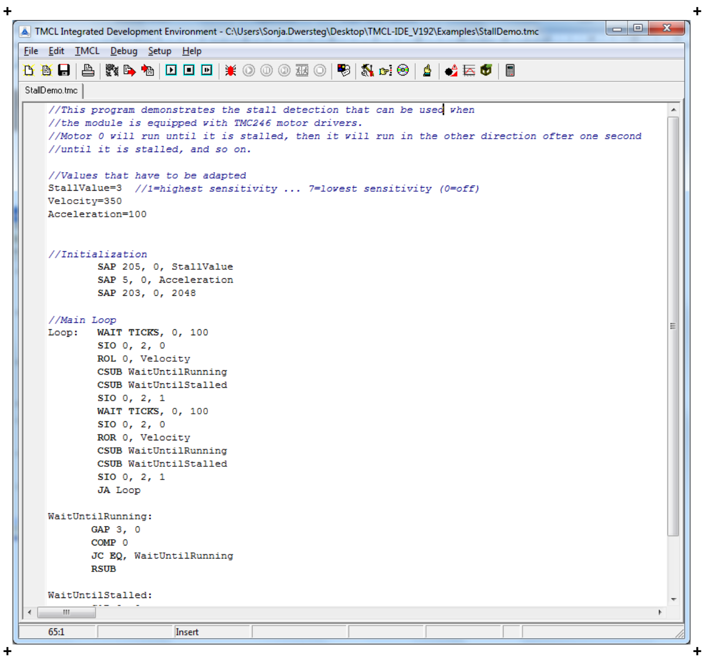
SOFTWARE

MODULES

V 2.01

TMCL-IDE USER MANUAL


TMCL Integrated Development Environment - C:\Users\Sonja.Dwersteg\Desktop\TMCL-IDE_V192\Examples\StallDemo.tmc
File Edit TMCL Debug Setup Help
StallDemo.tmc
65:1
Insert

```
//This program demonstrates the stall detection that can be used when
//the module is equipped with TMC246 motor drivers.
//Motor 0 will run until it is stalled, then it will run in the other direction ofter one second
//until it is stalled, and so on.

//Values that have to be adapted
StallValue=3  //1=highest sensitivity ... 7=lowest sensitivity (0=off)
Velocity=350
Acceleration=100


//Initialization
        SAP 205, 0, StallValue
        SAP 5, 0, Acceleration
        SAP 203, 0, 2048

//Main Loop
Loop:   WAIT TICKS, 0, 100
        SIO 0, 2, 0
        ROL 0, Velocity
        CSUB WaitUntilRunning
        CSUB WaitUntilStalled
        SIO 0, 2, 1
        WAIT TICKS, 0, 100
        SIO 0, 2, 0
        ROR 0, Velocity
        CSUB WaitUntilRunning
        CSUB WaitUntilStalled
        SIO 0, 2, 1
        JA Loop

WaitUntilRunning:
       GAP 3, 0
       COMP 0
       JC EQ, WaitUntilRunning
       RSUB

WaitUntilStalled:
```

TMCL-IDE

PC-Tools:

Integrated Development Environment for TMCL™ (Trinamic Motion Control Language)

TRINAMIC Motion Control GmbH & Co. KG
Hamburg, Germany

www.trinamic.com

TRINAMIC
MOTION CONTROL

Table of contents

Life support policy

TRINAMIC Motion Control GmbH & Co. KG does not authorize or warrant any of its products for use in life support systems, without the specific written consent of TRINAMIC Motion Control GmbH & Co. KG.

Life support systems are equipment intended to support or sustain life, and whose failure to perform, when properly used in accordance with instructions provided, can be reasonably expected to result in personal injury or death.

1 TMCL-IDE: Overview

The TMCL-IDE is an integrated development environment mainly for developing stand-alone TMCL™ applications, but it also includes a function for using the TMCL™ commands in direct mode. The TMCL-IDE is a PC application running under Windows 95/98/NT/2000/XP/Vista/Windows 7 (Windows 3.x is not supported) that includes

- a text editor for writing and modifying TMCL™ programs,
- a TMCL™ assembler for translating TMCL™ programs from mnemonic to binary format and downloading them into a module,
- a TMCL™ disassembler for getting a TMCL™ application out of a module and translating it back to mnemonic format,
- dialogues which allow setting the configuration of a module in an easy, interactive way,
- a dialogue for entering and executing TMCL™ commands in direct mode,
- a function for updating the firmware of a module,
- a dialogue for coolStep™ settings,
- a dialogue for configuring BLDC modules,
- and further powerful features.

Please be sure to always use the latest version of the TMCL-IDE as its functionality is being extended and improved constantly.

2 Installation / getting started

The TMCL-IDE is available on the TechLibCD and on www.trinamic.com.

1. To install the TMCL-IDE to your computer, just copy the file *TMCL.EXE* to your hard disk. Make sure the COM port you intend to use is not blocked by another program. After this double click the file to start the program.

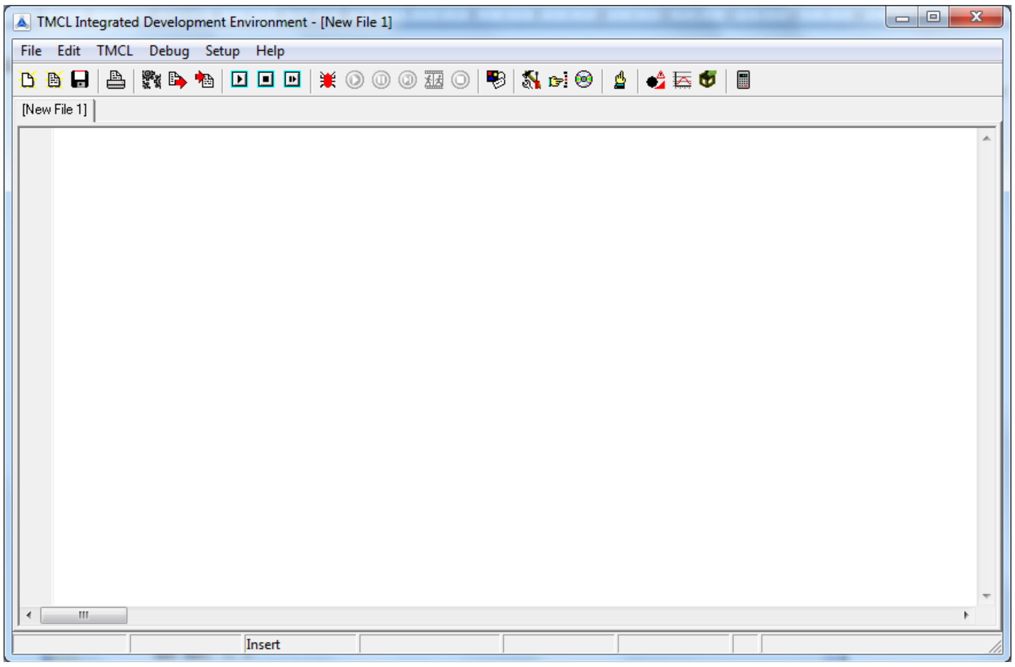

TMCL Integrated Development Environment - [New File 1]
File Edit TMCL Debug Setup Help
[New File 1]
Insert

Figure 2.1: TMCL-IDE main window

2. Choose *Setup* and *Options* and thereafter the *Connection tab*.

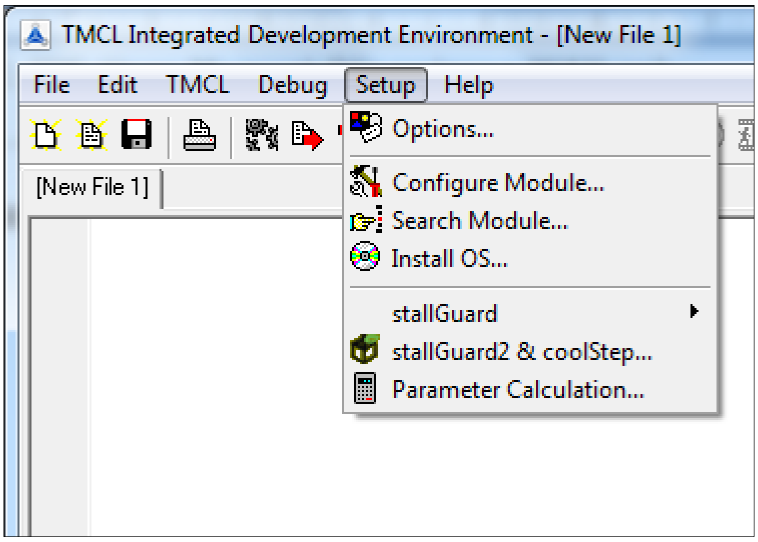
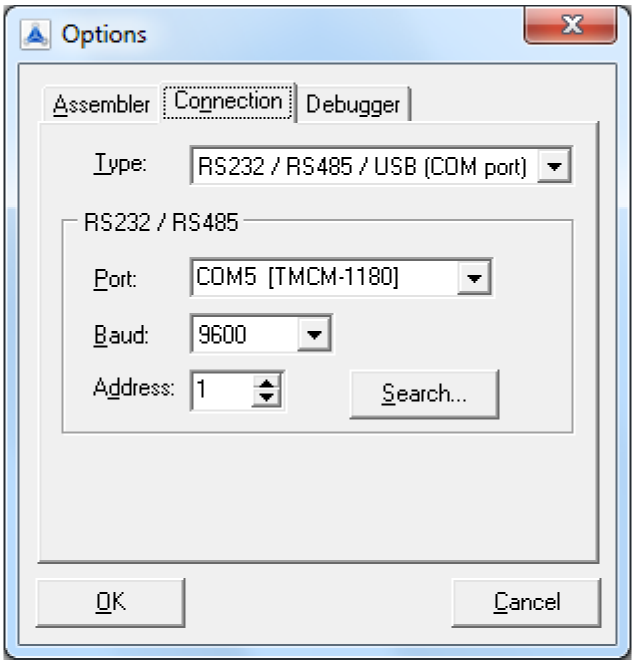

TMCL Integrated Development Environment - [New File 1]
File Edit TMCL Debug Setup Help
[New File 1]
Options...
Configure Module...
Search Module...
Install OS...
stallGuard
stallGuard2 & coolStep...
Parameter Calculation...

Figure 2.2: Setup

3. Choose *type* and *COM port* with the parameters shown below (baud rate usually 9600). Click *OK*. Make sure the COM port is not blocked by another application.

   You should not need to change any other settings.



Type: RS232 / RS485 / USB (COM port)

RS232 / RS485

Port: COM5 [TMCM-1180]

Baud: 9600

Address: 1

Figure 2.3: Connection tab

2.1 Getting started

First, try out the TMCL™ commands in direct mode.

1. Start TMCL™ *Direct Mode* with a click on the icon.

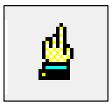


Figure 2.4: TMCL™ Direct Mode icon

2. If the communication is established the Module is automatically detected.

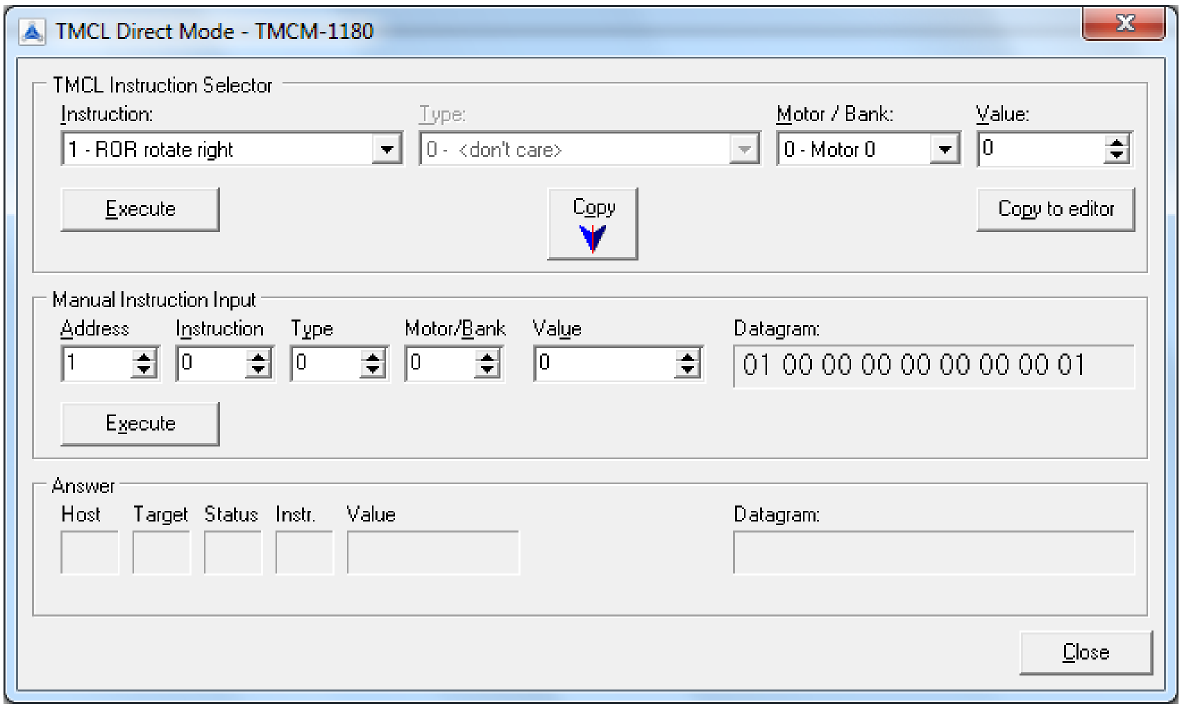
   If it is not detected, please check cables, interface, power supply, COM port, and baud rate.

3. Issue a command by choosing *Instruction*, *Type* (if necessary), *Motor*, and *Value* and click *Execute* to send it to the module. The motor will follow the instruction now.

MOTOR NUMBERS:

Motors	Motor numbers
one axis	0
two axes	0, 1
three axes	0, 1, 2
six axes	0, 1, 2, 3, 4, 5

The motor number is always 0, if only one motor is involved.



TMCL Direct Mode - TMCM-1180

TMCL Instruction Selector

Instruction: 1 - ROR rotate right
Type: 0 - <don't care>
Motor / Bank: 0 - Motor 0
Value: 0

Manual Instruction Input

Address: 1
Instruction: 0
Type: 0
Motor/Bank: 0
Value: 0
Datagram: 01 00 00 00 00 00 00 00 01

Answer

Host, Target, Status, Instr., Value, Datagram:

Figure 2.5: TMCL™ instruction selector

Examples:

- **ROR** rotate right, motor **0**, value **500** -> Click *Execute*. The motor is rotating now.
- **MST** motor stop, motor **0** -> Click *Execute*. The motor stops now.

2.2 Writing a simple TMCL™ program

After you have successfully tried out the direct mode, have a look at the TMCL™ sample programs supplied on the software CD. Load a program, assemble it, download it into a module and run it. Try to understand the TMCL™ sample programs. After that you are ready to write your own TMCL™ programs.

Example for a very simple TMCL™ program:

1. Type the following text in the open window.

```
      SAP 4, 0, 100            //Set the maximum speed

Loop: MVP ABS, 0, 150000       //Move to position 150000
      WAIT POS, 0, 0
      WAIT TICKS, 0, 200
      MVP ABS, 0, 0            //Move back to position 0
      WAIT POS, 0, 0
      WAIT TICKS, 0, 100

      JA Loop                  // Infinite Loop
```

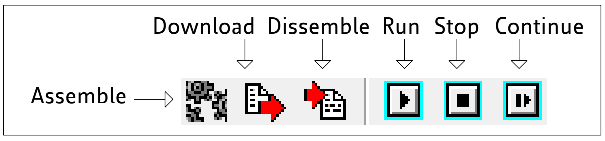


Figure 2.6: TMCL™ icons

2. Click the *Assemble* icon.
3. Thereafter click the *Download* icon to download the code to the module.
4. Click the *Run* icon. The downloaded program will now be executed.
5. Click *Stop* button to stop the program.

Note: The TMCL-IDE offers other examples, which can be assembled and downloaded for testing purposes. Open the files with the *Open* icon.

3 The File menu

The *File menu* provides functions to load and save files. Some functions can also be found in the tool bar below the menu bar.

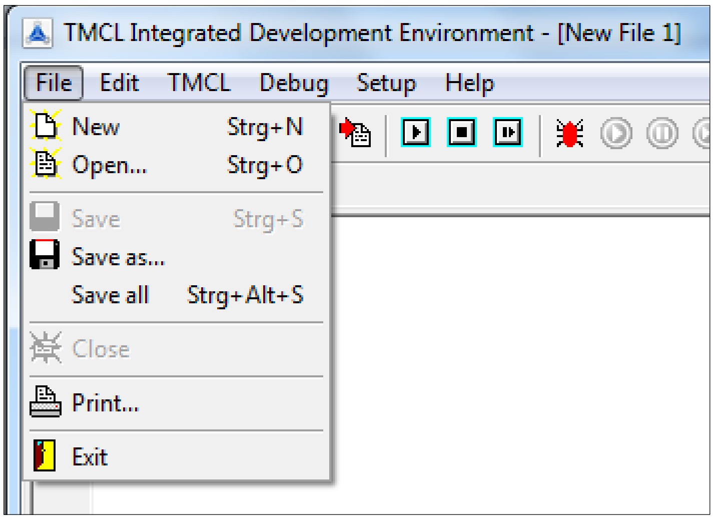


Figure 3.1: The File menu

3.1 New

This function opens a new editor page, so a new file can be created.

3.2 Open

After selecting the *Open* function a file selection dialog will be shown. Here you can select a file to be opened. Then, a new editor page opens and the selected file will be loaded into that editor page.

3.3 Save, Save as

Select one of these functions to save the contents of the currently selected editor page into a file. The *Save as* function allows saving a file using a new name.

3.4 Save all

Select this function to save all files that are currently loaded into the editor and have been changed.

3.5 Close

The *Close* function closes the actual editor page. This function can also be selected from the context menu of the editor (click the right mouse button in the editor window to open the context menu).

3.6 Exit

Use this function to close the TMCL-IDE. The same function can be achieved by closing the main window.

4 The integrated text editor

The text editor of the TMCL-IDE mainly provides the functionality of a standard Windows text editor. *An additional function to the Windows standard is Ctrl-Y to delete a line.* Some functions of the editor can be found in the *Edit menu*. It is also possible to edit multiple documents. They will be shown in a workbook style.

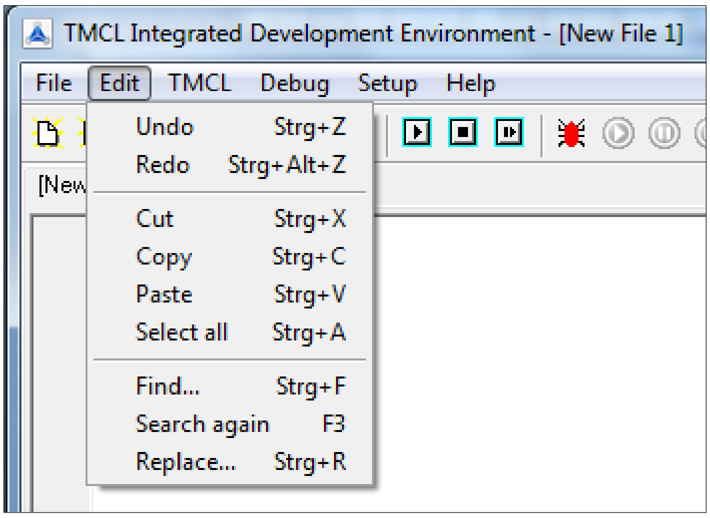


Figure 4.1: Edit menu

5 The TMCL™ menu

5.1 Basics

The *TMCL™ menu* contains all functions needed for assembling, downloading and disassembling TMCL™ programs. It also contains the functions to run and stop a TMCL™ program on a module and to use TMCL™ commands in direct mode. Assembling a TMCL™ program always takes place from the editor. So, before assembling a TMCL™ file it must be loaded into it.

All icons of the TMCL™ menu are on the toolbar, too.

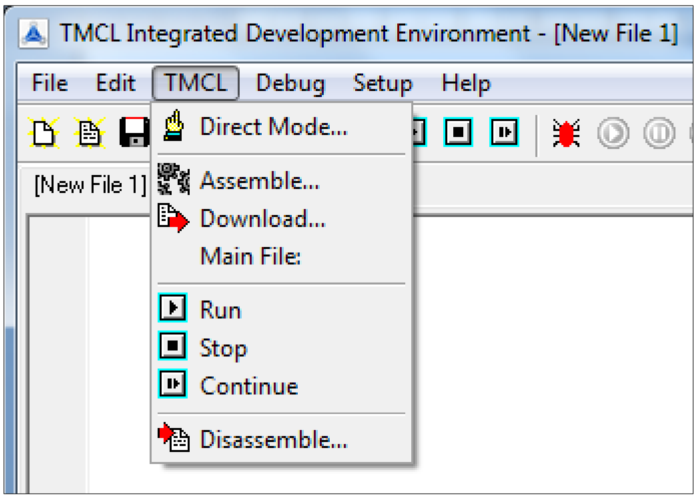


Figure 5.1: The TMCL™ menu

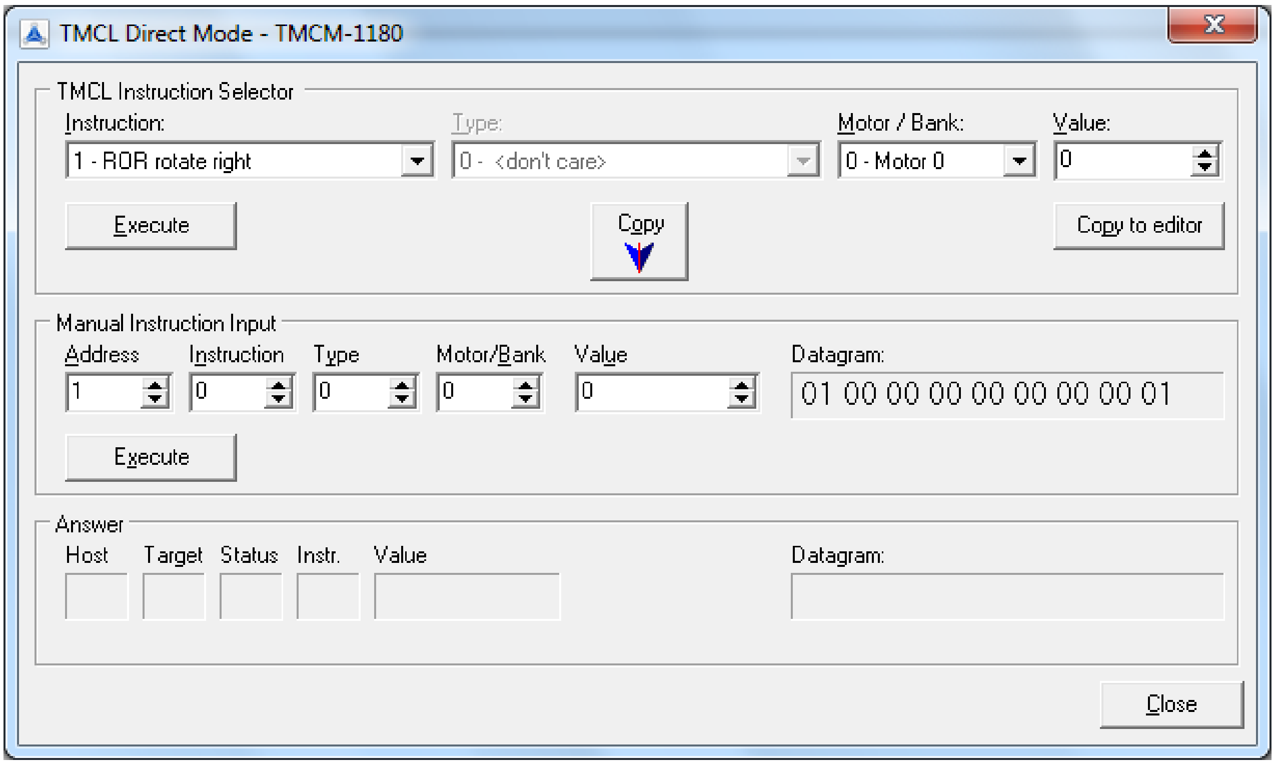
5.2 Direct Mode

Select the *Direct Mode* function in the TMCL™ menu to open the direct mode dialog. The TMCL-IDE then first checks the type of the module that is connected, as some menus in the direct mode dialogue vary because different modules have diverse attributes.

- *If the type of the module cannot be detected (e.g. because there is no module connected), a little dialogue pops up where you are prompted to select your module type. Do this by choosing the appropriate module type and click OK.*
- *If a module is not detected automatically the communication is not established. Please check cables, interface, COM port, and power supply again.*

By using the direct mode you can send commands to a module that are executed immediately. In the *TMCL™ Instruction Selector* area you can select a command and its parameters. Click the *Execute* button in this area to send the command to the module. The response will be displayed in the *Answer* section. By clicking the *Copy to editor* button the command mnemonic will be copied to the TMCL™ editor.

You can also enter the instruction numbers directly in the *Manual Instruction Input* area and execute them by clicking the appropriate *Execute* button (but this option is needed only very seldom). The *Copy* button in the TMCL™ Instruction Selector area copies the instruction bytes to the Manual Instruction Input area.


TMCL Direct Mode - TMCM-1180

TMCL Instruction Selector

Instruction: 1 - ROR rotate right
Type: 0 - <don't care>
Motor / Bank: 0 - Motor 0
Value: 0

Manual Instruction Input

Address: 1
Instruction: 0
Type: 0
Motor/Bank: 0
Value: 0
Datagram: 01 00 00 00 00 00 00 00 01

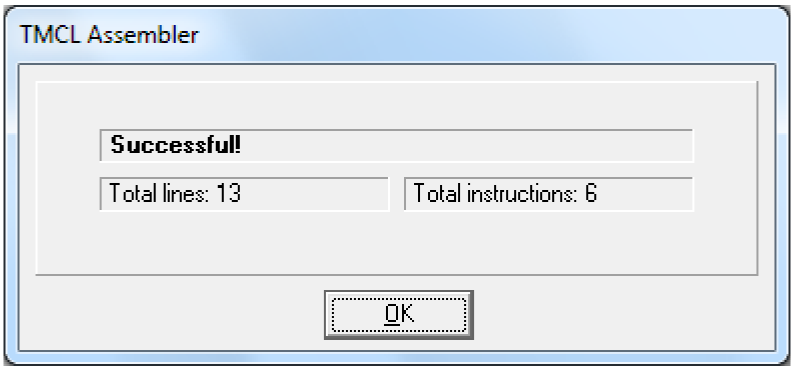
Answer

Host, Target, Status, Instr., Value, Datagram:

Figure 5.2: The direct mode dialog

5.2.1 Assemble a TMCL™ program

Selecting the *Assemble* function in the TMCL™ menu assembles a file. If more than one file is open and no main file has been defined the currently selected file will be assembled. If a main file has been selected (5.4) the main file will always be assembled, regardless of the currently selected editor file. The progress of the assembly is displayed.



Successful!

Total lines: 13

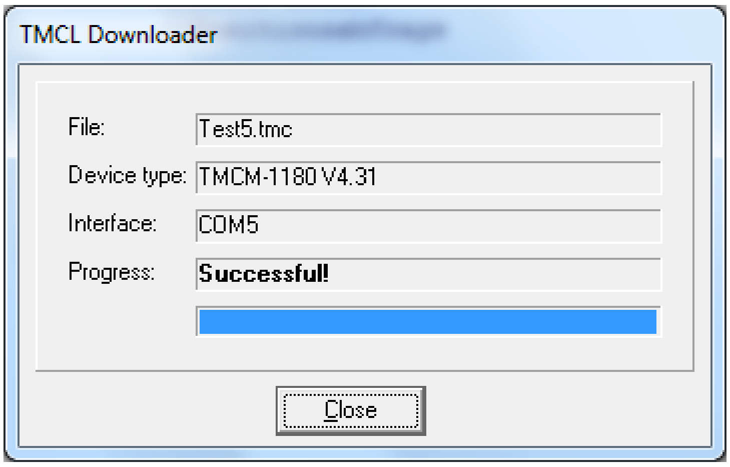
Total instructions: 6

Figure 5.3: Progress of the assembly

If an error occurs, the line containing the error will be highlighted and the assembly will be aborted. The error message will be displayed in the assembler progress dialog.

5.3 Download a TMCL™ program

After a TMCL™ program has been successfully assembled; it can be downloaded into the module. Make sure that the module is connected to a COM port and that the port is selected in the *Options* dialogue. By selecting the *Download* function the program will then be downloaded into the module. The download progress is shown in a special window. Downloading can be aborted by clicking the *Abort* button.


TMCL Downloader

File: Test5.tmc
Device type: TMCM-1180 V4.31
Interface: COM5
Progress: **Successful!**

Figure 5.4: Downloader

5.4 Select a main file

By using the *Main File* function you can select a main file. Now, this file will be always used for assembling, regardless of which file is currently selected in the editor.

You can click on the *Clear* button in the *Select Main TMCL™ File* dialog to switch off the function and to assemble always the currently selected editor file.

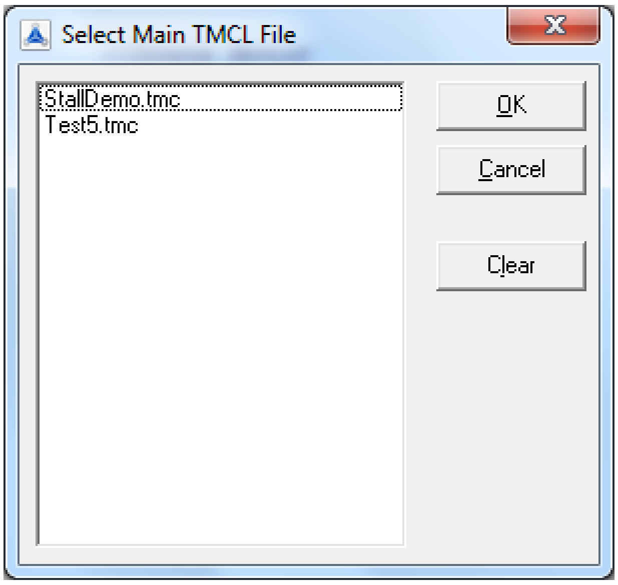


Figure 5.5: Main file dialog

5.5 Run a TMCL™ program

The ***Run*** function starts the TMCL™ program which is currently in the TMCL™ memory of the module just by sending a reset and a run command to the module. The module then starts executing the program form the first command on.

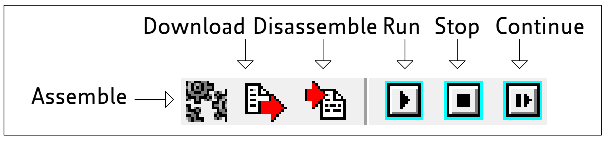


Figure 5.6: *Assemble*, *Download*, *Disassemble*, *Run*, *Stop*, and *Continue*

5.6 Stop a TMCL™ program

The *Stop* function sends a stop command to the module and stops the execution of a TMCL™ program.

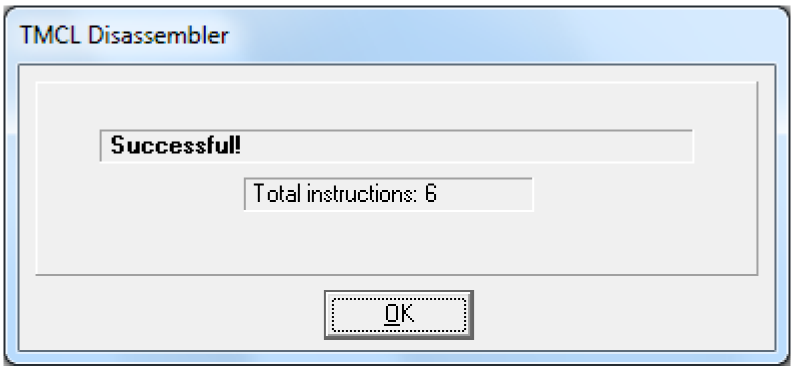
5.7 Continue a TMCL™ program

The *Continue* function allows continuing the run of a TMCL™ program that has previously been stopped. It sends a run command to the module (without sending a reset command before). The module continues executing the program from the next command on. *(For debugging functions like single steps, breakpoints etc. please see 0)*

5.8 Disassemble a TMCL™ program

The *Disassemble* function reads out the TMCL™ memory of a module and disassembles its contents. The result is written into a new editor page. This function allows checking the program which is currently in the TMCL™ program memory of a module.

The progress of downloading the program from the module and the disassembly is shown in special windows. It can be aborted by clicking the *Abort* button.



Successful!

Total instructions: 6

Figure 5.7: Disassembler

6 The Setup menu

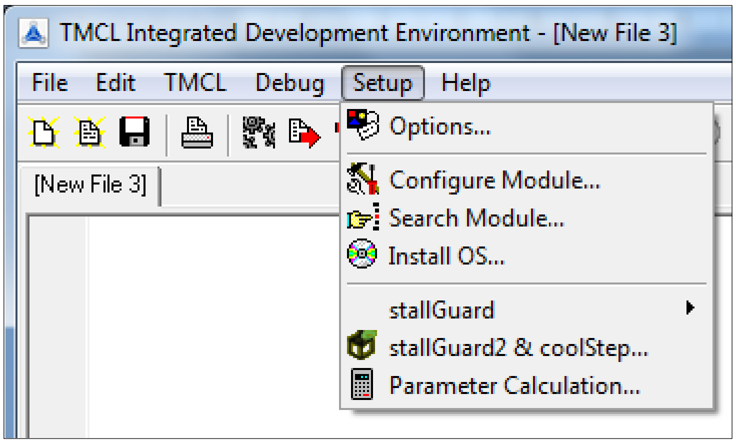


Figure 6.1: Setup menu

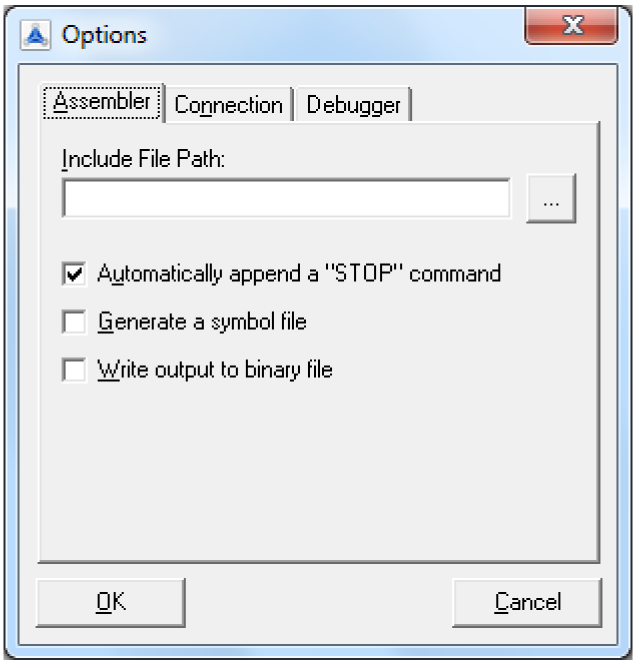
The Setup Menu provides tools for the basic configuration of your module:

- Configuration of interface and communication
- stallGuard™, stallGuard2™, and coolStep™ configuration tools
- Parameter calculation tool

6.1 Options dialogue

This dialogue with the tabs *Assembler*, *Connection*, and *Debugger* allows setting up all global options of the TMCL-IDE.

6.1.1 Assembler options


Options

Assembler | Connection | Debugger

Include File Path:

- [✓] Automatically append a "STOP" command
- [ ] Generate a symbol file
- [ ] Write output to binary file

Figure 6.2: Assembler options

The *Assembler* tab in the *Options* dialogue provides the following assembler options:

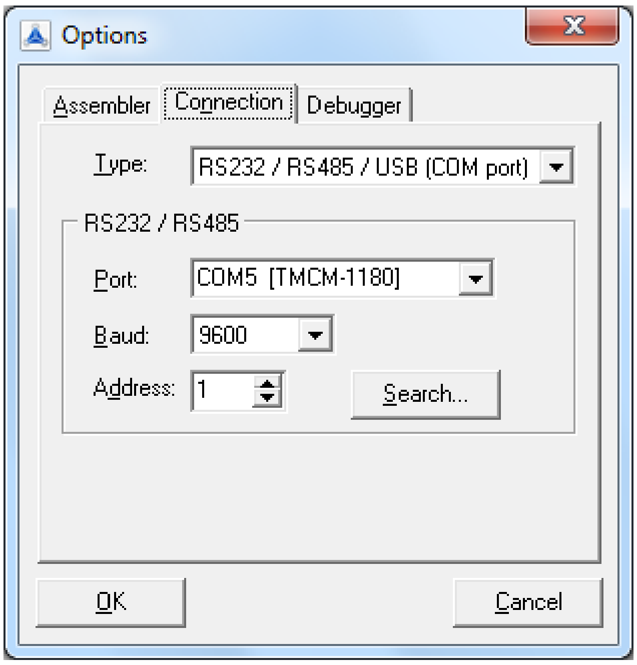
Include file path
The path where the assembler searches for include files included by the *#include* directive.

Automatically append a *STOP* command
If this option is selected, the assembler automatically appends a *Stop* command at the end of every TMCL™ program (if the last command in the program is not already a *Stop* command).

Generate a symbol file
The assembler generates a text file that contains the address of every label. This can be useful to start the program from other addresses than 0.

Write output to binary file
The assembler writes its output also to a binary file. For every command eight bytes are written (the command with checksum, but without a device address).

6.1.2 Connection options
Here the interface type and the port can be selected that are to be used to communicate with a module. First, select the connection type. In most cases this will be RS232/RS485. If you have a TRINAMIC CANnes card or USB-2-X module and you would like to use the CAN or I²C interface of a module just select the appropriate connection type. The connection parameters that are displayed below the connection type change according to the selected interface.


Options

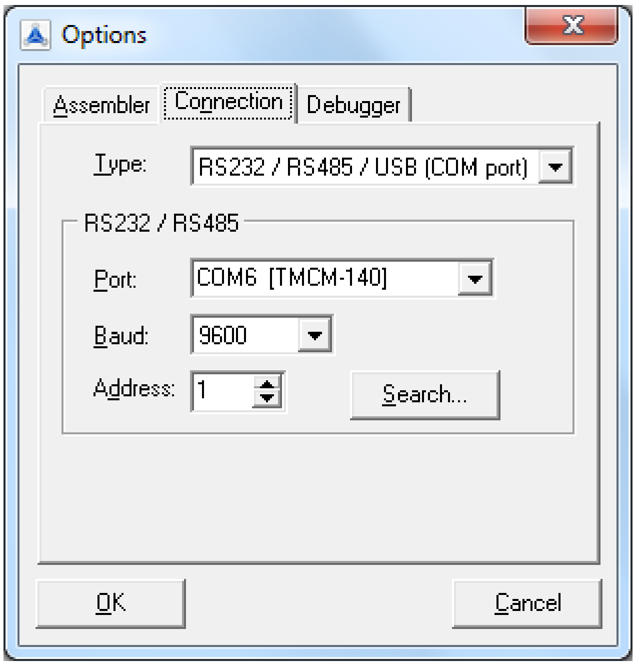
Type: RS232 / RS485 / USB (COM port)

RS232 / RS485

Port: COM5 [TMCM-1180]

Baud: 9600

Address: 1


Options

Type: RS232 / RS485 / USB (COM port)

RS232 / RS485

Port: COM6 [TMCM-140]

Baud: 9600

Address: 1

Figure 6.3: Examples: Connection options

Choose the fitting connection for your device:

ESD CAN module:	CAN (ESD)
TRINAMIC USB-2-485:	RS232/RS485 (COM port)
USB / TMCM-610/612:	USB (TMCM-610/612 device)
USB-2-X device:	I²C (USB-2-X)
USB-2-X device:	CAN (USB-2-X)
CANnes card:	CAN (CANnes card)
Other modules:	RS232/RS485 (COM port)

After you have selected *RS232/RS485* the *COM port* that is to be used, the *baud rate* and the *module address* can be set.

- The factory default baud rate on a TMCM module is 9600.
- The factory default module address is 1.

If you do not know the address of the connected module, just click the *Search* button. The *Search* function provides the address search dialogue. Here, just click the *Start* button. Now, every address (1… 255) will be tried out. You can abort the progress by clicking the *Stop* button.

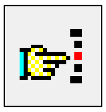
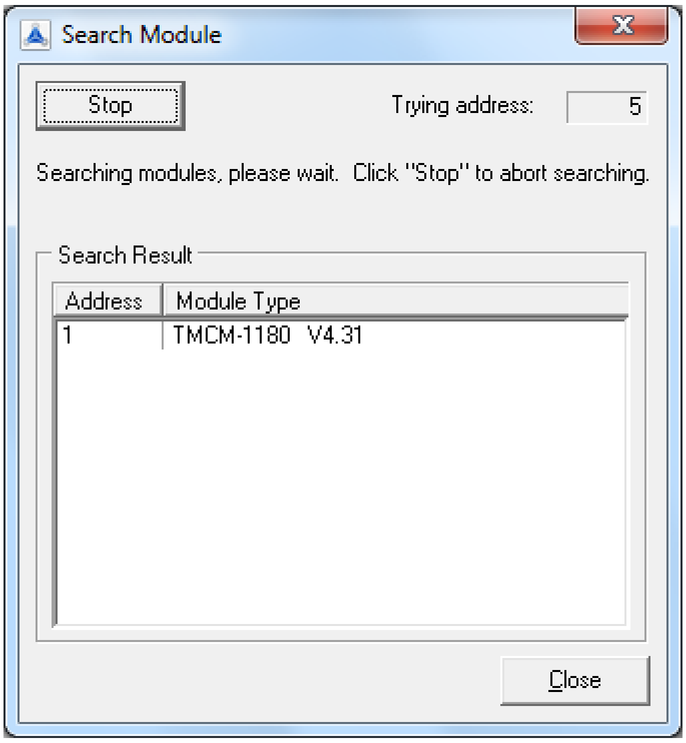


Figure 6.4: Search module icon



Trying address: 5

Searching modules, please wait. Click "Stop" to abort searching.

Search Result

Address	Module Type
1	TMCM-1180 V4.31

Figure 6.5: Search module dialogue

Please note that RS485 adapters where the data direction is controlled by the RTS line of the COM port do not work with Windows 95, Windows 98 or Windows ME (due to a bug in these operating systems). They can only be used with Windows NT4, Windows 2000 or Windows XP or Vista and also with these operating systems there may be the problem that switching back to receive takes place too late. It is recommended to use the TRINAMIC USB-2-485 instead.

Special information for the use of the CANnes card:

- *The factory default of the CAN bit rate on TMCM modules is normally 250kBit/s.*
- When the interface type *CANnes card* is selected, the CAN parameters are displayed and can be changed.
- *Device* selects the CANnes card you want to use (if you have more than one CANnes card installed).
- *Termination* turns the termination resistor on the CANnes card on or off.
- *Send ID* is the CAN ID to use when sending CAN data grams to the module.
- *Recv. ID* is the ID that the module uses to send back data to the PC.

Special information for the use of the USB-2-X:

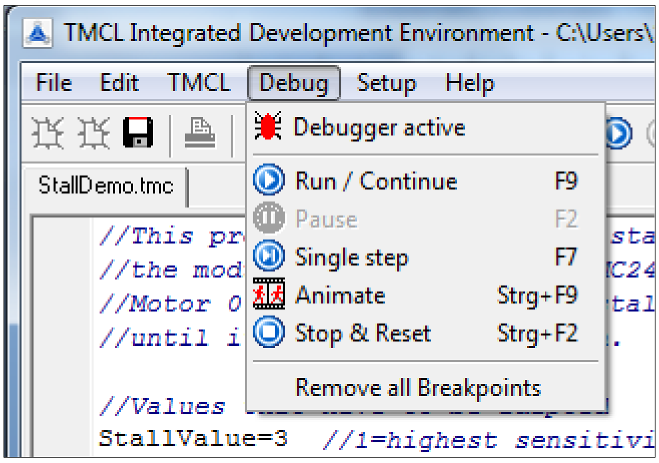
- Select *IIC* when you wish to use the IIC interface of a Trinamic USB-2-X device. Here, only the device name of the USB-2-X device and the IIC address of the module can be set. The factory default on a new module is normally 2.
- Select *CAN* when you wish to use the CAN interface of a Trinamic USB-2-X device. The device name of the USB-2-X device and the CAN send ID, receive ID and bit rate can be set. Please note that the USB-2-X device must have firmware version 1.01 or higher to make use of this feature!
- Select *RS485* when you wish to use the RS485 interface of a Trinamic USB-2-X device. The device name of the USB-2-X, the bit rate and the module address can be set. Please note that the USB-2-X device must have at least firmware version 2.05 to make use of this feature! This kind of RS485 interface is mainly useful for updating modules which only have the RS485 interface (e.g. TMCM-110/RS485).

Special information for the use of the TMCM-610 and TMCM-612:

- When a TMCM-610 module is connected via USB the option *USB direct* can be used. The TMCL-IDE can communicate with this module directly via USB. The device name of the TMCM-610 module can be selected here.

6.1.3 TMCL™ debugger options

The TMCL™ debugger makes source level debugging of TMCL™ programs possible. The TMCL™ program still runs on the module, so true in system debugging can be done.



```
//This pr
//the mod
//Motor 0
//until i

//Values
StallValue=3   //1=highest sensitivi
```

Figure 6.6: Debugger menu

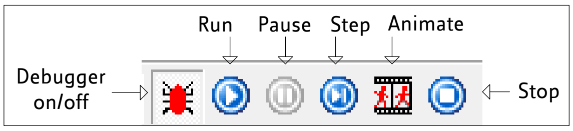


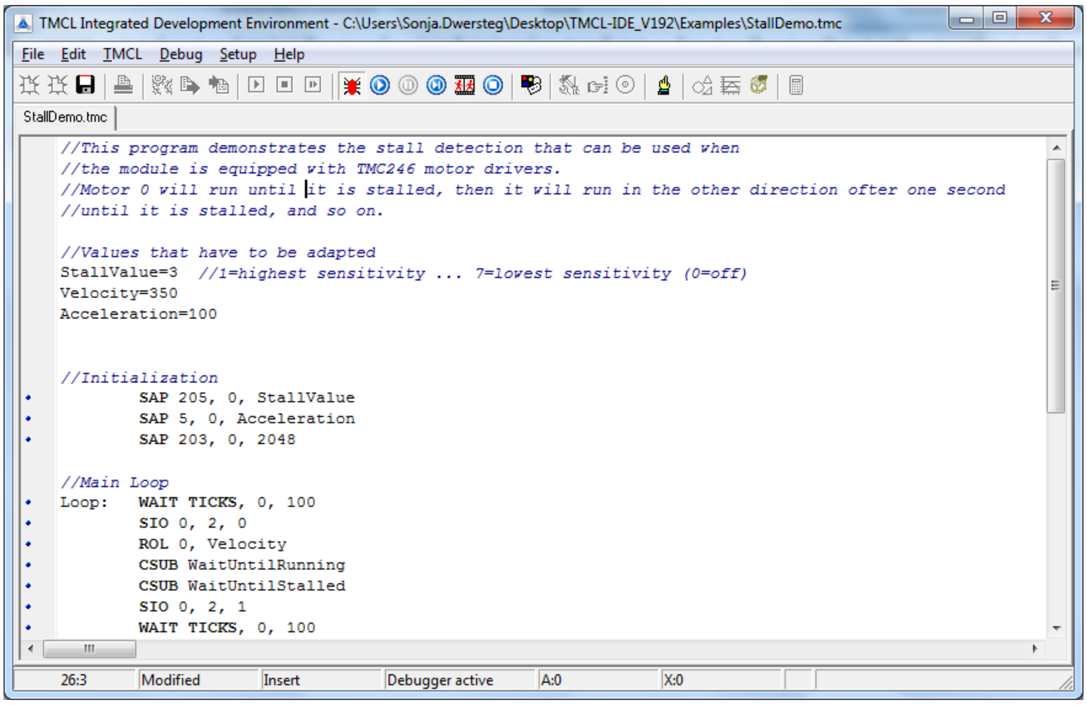
Figure 6.7: Debugger icons

6.1.3.1 Starting the debugger

Before starting the debugger you will first have to make sure that the module is connected to the PC and that the program in the editor of the TMCL-IDE is the same as the program on your module. This can be done either by assembling the program that currently is in the editor and afterwards downloading it to the module or by disassembling the program that is currently stored on the module.

After these two preconditions have been verified you can start the debugger either by selecting the function *Debugger active* in the Debug menu or by clicking the *Debugger* icon on the tool bar. After the debugger has been successfully started, the debugger functions will be enabled and most other functions of the TMCL-IDE will be disabled (now it is not possible to change the program in the editor).

You can exit the debugger by clicking *Debugger active* in the Debug menu or the *Debugger* icon on the tool bar once again. Then, all debugger functions will be disabled and all other functions of the TMCL-IDE will be enabled again.


TMCL Integrated Development Environment - C:\Users\Sonja.Dwersteg\Desktop\TMCL-IDE_V192\Examples\StallDemo.tmc
File Edit TMCL Debug Setup Help
StallDemo.tmc
26:3
Modified
Insert
Debugger active
A:0
X:0

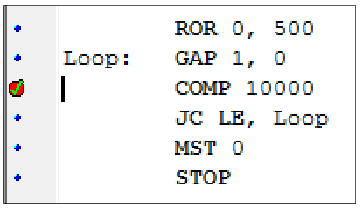
```
//This program demonstrates the stall detection that can be used when
//the module is equipped with TMC246 motor drivers.
//Motor 0 will run until it is stalled, then it will run in the other direction after one second
//until it is stalled, and so on.

//Values that have to be adapted
StallValue=3  //1=highest sensitivity ... 7=lowest sensitivity (0=off)
Velocity=350
Acceleration=100


//Initialization
        SAP 205, 0, StallValue
        SAP 5, 0, Acceleration
        SAP 203, 0, 2048

//Main Loop
Loop:   WAIT TICKS, 0, 100
        SIO 0, 2, 0
        ROL 0, Velocity
        CSUB WaitUntilRunning
        CSUB WaitUntilStalled
        SIO 0, 2, 1
        WAIT TICKS, 0, 100
```

Figure 6.8: The TMCL™ program with *debugger active*

6.1.3.2 Breakpoints

Breakpoints can be set or removed by clicking the appropriate line on the left breakpoint bar of the editor. A blue bullet is displayed in every line where a breakpoint can be set. When a breakpoint is set a *red bullet with a green tick* is displayed in that line.

When a program is run either by the *Run* or by the *Animate* function of the debugger it will be stopped when a breakpoint is reached. It can be continued by clicking *Run*, *Animate* or *Step* again. It can also be reset by clicking *Stop/Reset*, so that it can be restarted form the beginning again.



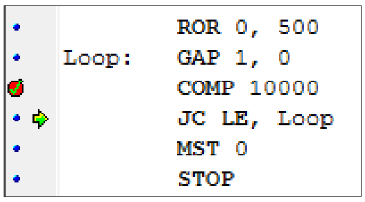
```
        ROR 0, 500
Loop:   GAP 1, 0
        COMP 10000
        JC LE, Loop
        MST 0
        STOP
```

Figure 6.9: Breakpoint marked

6.1.3.3 The *Run/Continue* function

Choose the *Run/Continue* function from the Debug menu (or click the *Run* icon or press *F9*) to start the program resp. continue its execution. The program will be stopped either when its end is reached, if a breakpoint is reached or when the *Pause* function in the Debug menu or the *Pause* icon is selected. In the latter two cases the actual position in the program will be shown in the editor by a *green arrow* on the left side and the program execution can be continued by using the *Run/Continue* function, the *Step* function or the *Animate* function. The contents of the accumulator and the X register are also shown on the status bar while the program is paused.

If you wish to restart the program from the very beginning, select the *Stop/Reset* function before clicking *Run/Continue*.



```
        ROR 0, 500
Loop:   GAP 1, 0
        COMP 10000
        JC LE, Loop
        MST 0
        STOP
```

Figure 6.10: Green arrow for actual position

6.1.3.4 The *Pause* function

When a program is running (started by the *Run/Continue* or by the *Animate* function), the program execution can be interrupted at any time by selecting the *Pause* function from the Debug menu or clicking the *Pause* icon on the tool bar or pressing *F2*. Program execution can be continued by clicking ***Run***, *Animate* or *Step* again. While a program is paused the actual position in the program is shown by a *green arrow* in the editor. The contents of the accumulator and the X register are shown on the status bar of the TMCL-IDE.

6.1.3.5 The *Step* function

Use this function for a step-by-step execution of the TMCL™ program. Every time the ***Step*** function is selected (by selecting ***Step*** in the ***Debug menu***, clicking the ***Step*** icon on the tool bar or pressing ***F7***) the next command in the TMCL™ program is executed. The actual position is shown by a ***green arrow*** in the editor. The contents of the accumulator and the X register are also shown on the status bar.

6.1.3.6 The *Animate* function

This function automatically executes the TMCL™ program step-by-step, so that the flow of the program can be seen. The actual position in the program is shown and updated after every command, and the contents of the accumulator and the X register are also shown on the status bar and updated after every command.

The program will be paused when running into a breakpoint or when the *Pause* function is selected. Program execution can be continued either by the *Run/Continue*, the *Animate* or the *Step* function.

6.1.3.7 The *Stop/Reset* function

Selecting *Stop/Reset* from the *Debug menu* (or clicking the *Stop/Reset* button or pressing *Ctrl+F2*) stops program execution immediately and resets the program. This means that starting the program again using *Run/Continue*, *Animate* or *Step* will start the program from the beginning.

This function can also be used when the program execution is paused (by a breakpoint or by the *Pause* function) to reset it and ensure that the program can be started again from the beginning.

6.1.3.8 The *Direct Mode* function in the debugger

In the debugger the *Direct Mode* can be used at any time when the program is not running. Now, it is possible to inspect or change parameters. Use this function with care, as changing parameters out of the normal program flow can lead to unexpected behavior of the TMCL™ program.

Please refer to paragraph 5.2 for further information about TMCL™ direct mode.

6.2 Configure

This function provides a dialogue for changing the configuration (global parameters) of a module.

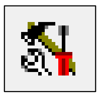
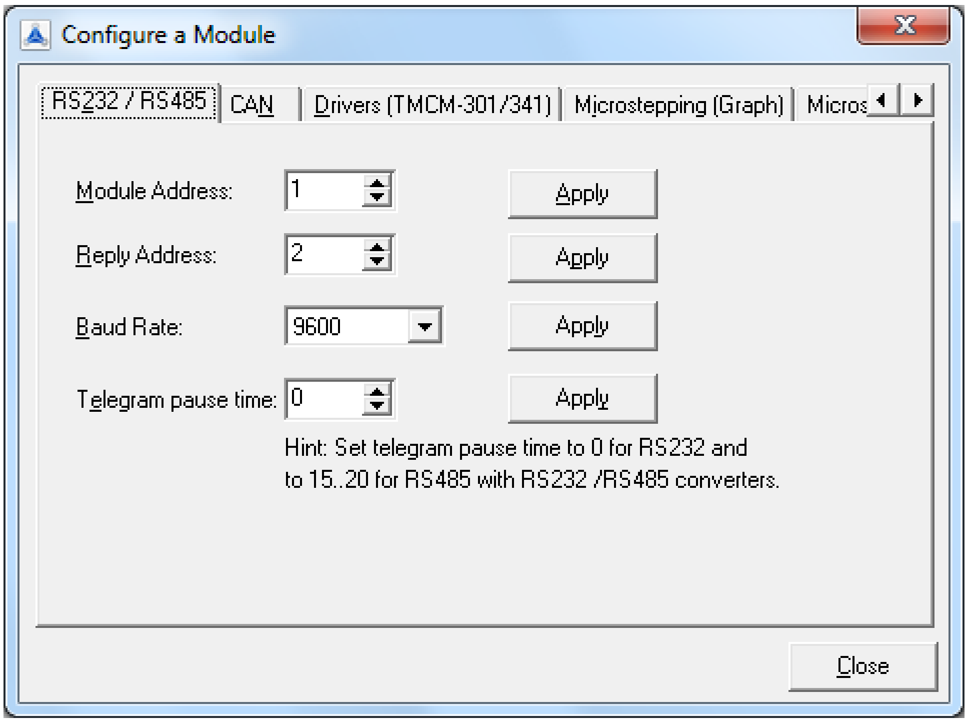


Figure 6.11: Configure module icon

6.2.1 RS232/RS485

Here, you can change the *send address*, the *receive addresses* and the *baud rate* of the serial interface of a module. Just set the desired value and then click on the appropriate *Apply* button to send the new setting to the module. Be careful, as most changes take effect immediately.



Module Address: 1

Reply Address: 2

Baud Rate: 9600

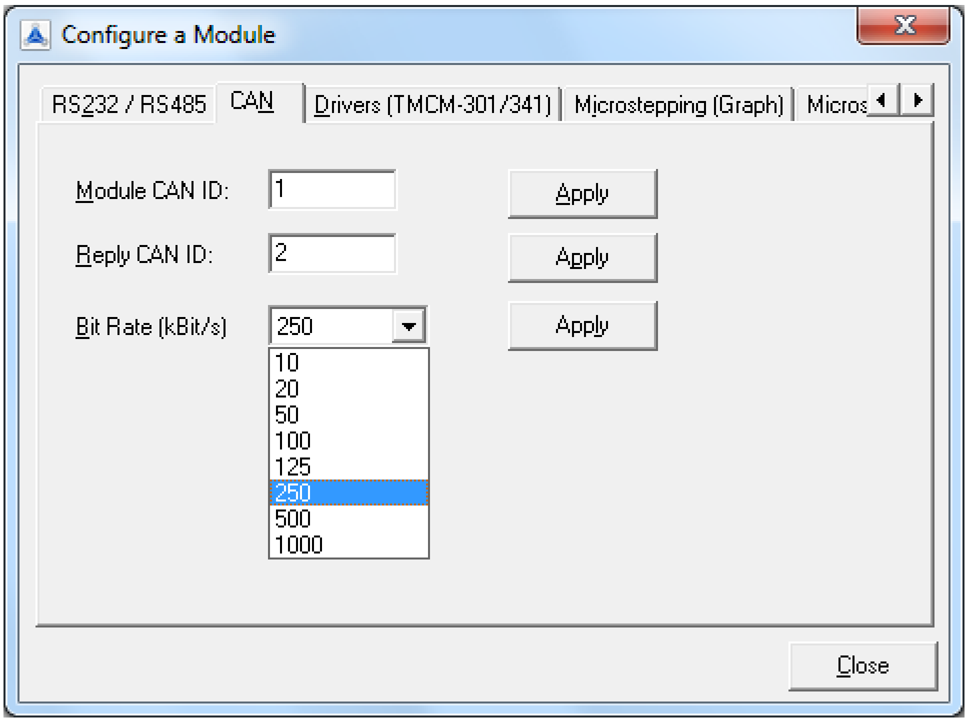
Telegram pause time: 0

Hint: Set telegram pause time to 0 for RS232 and to 15..20 for RS485 with RS232 /RS485 converters.

Figure 6.12: RS232/RS485 settings

6.2.2 CAN

This page provides the settings of the CAN interface of a module. The usage is the same as with the RS232/RS485 page.



Module CAN ID: 1

Reply CAN ID: 2

Bit Rate (kBit/s) 250

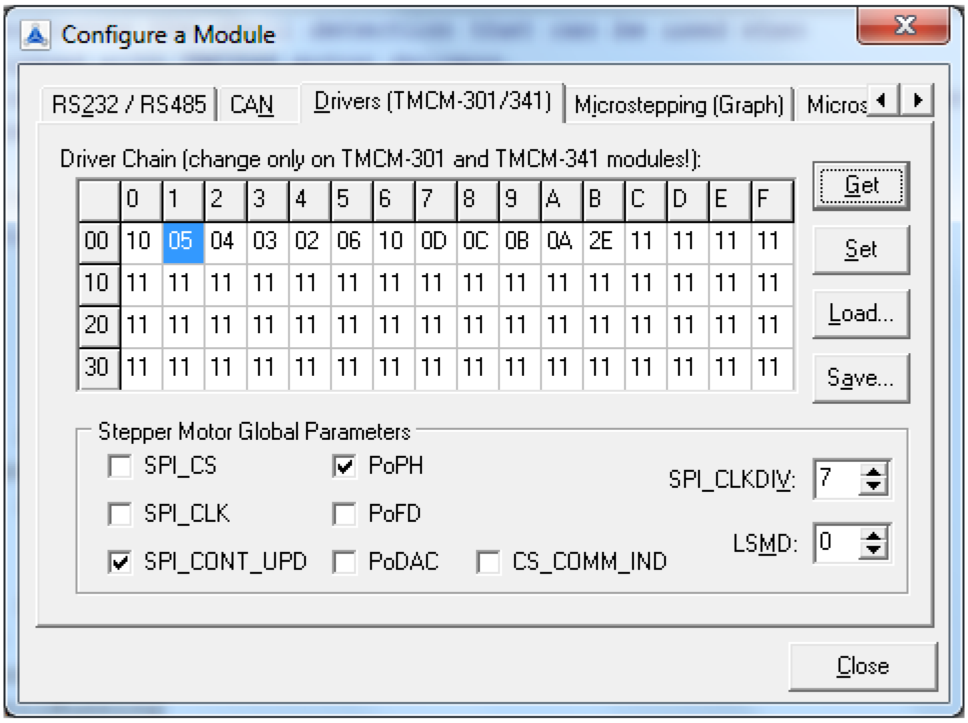
Figure 6.13: CAN settings

6.2.3 Drivers

This page allows changing the settings for the SPI driver chain. See the TMC428 / TMC429 datasheets for details. *Normally these settings must not be changed!*

It is only necessary to change the settings for the SPI driver chain on TMCM-301 and TMCM-341 modules if you prefer other motor drivers than TMC239, TMC249 or the TMCM-035 module.

When programming user specific driver configuration chain tables please disable the automatic mixed decay handling for TMC236 drivers, the automatic full step switching and the freewheeling function first (axis parameters 203, 204 and 211). This is needed because these functions modify the driver configuration table.



Driver Chain (change only on TMCM-301 and TMCM-341 modules!):

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	10	05	04	03	02	06	10	0D	0C	0B	0A	2E	11	11	11	11
10	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11
20	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11
30	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11

Stepper Motor Global Parameters

- ☐ SPI_CS
- ☐ SPI_CLK
- ☑ SPI_CONT_UPD
- ☑ PoPH
- ☐ PoFD
- ☐ PoDAC
- ☐ CS_COMM_IND

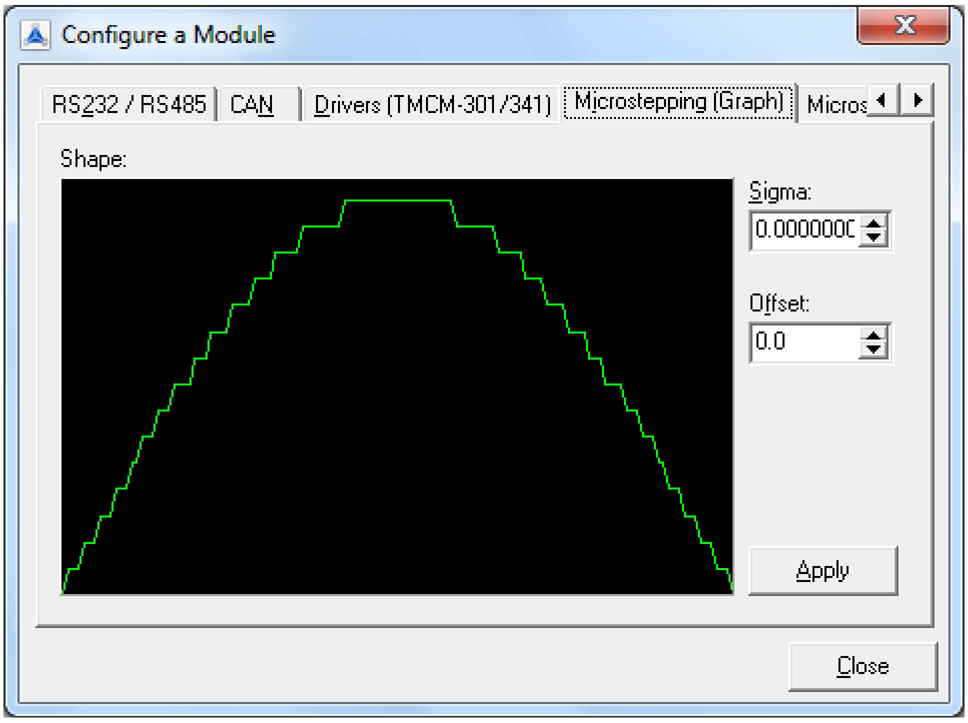
SPI_CLKDIV: 7

LSMD: 0

Figure 6.14: Driver chain configuration

6.2.4 Micro stepping (graphical view)

Here, the micro stepping wave of the module can be changed to enhance the micro stepping behavior of a stepper motor. The micro stepping wave function can be changed between a triangular wave, a sine wave or a trapezoid wave or something between that by changing the sigma value. Clicking the *Apply* button programs the new wave table into the module. Just let the motor run and try out the best value.



Sigma: 0.000000C

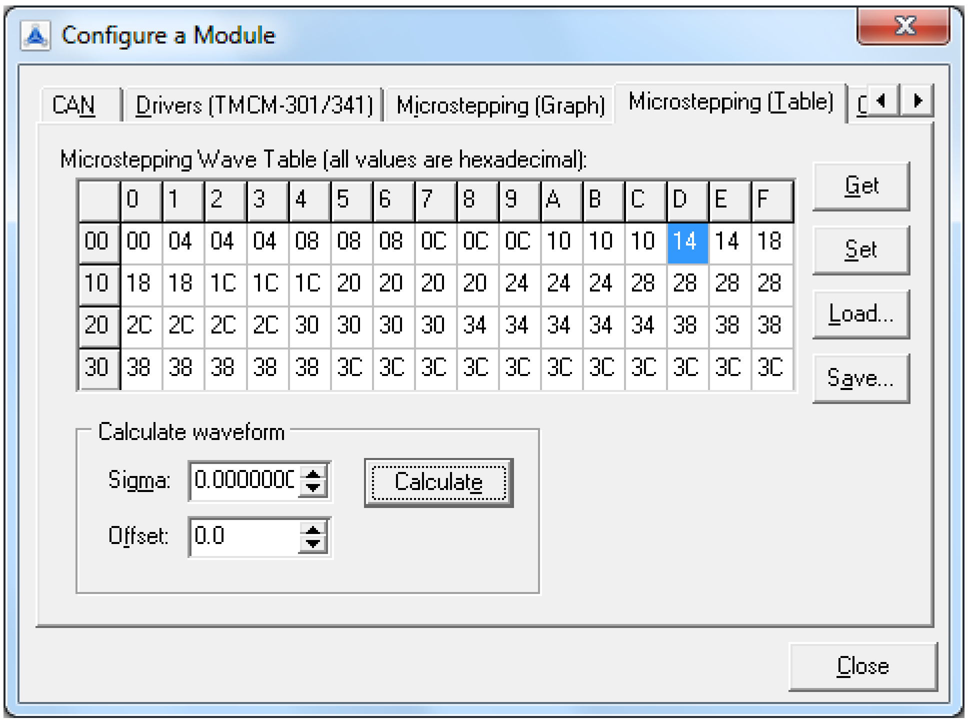
Offset: 0.0

Figure 6.15: Microstep wave form setting (graphical view)

6.2.5 Microstepping (table view)

The table view allows a finer adjusting of the microstep table. Clicking the *Get* button reads the actual microstep table from the module. You can then edit the values. All values are hexadecimal numbers between 0 and 3F. The *Set* button programs the table that can be seen on this screen into the module. Use the *Load* and *Save* buttons to load a table from a file or store a table into the file.

When clicking the *Calculate* button a waveform will be calculated in the same manner as in the graphical view of the microstepping function. This can be used as a basis of a table that can then be fine tuned manually.



Microstepping Wave Table (all values are hexadecimal):

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	00	04	04	04	08	08	08	0C	0C	0C	10	10	10	14	14	18
10	18	18	1C	1C	1C	20	20	20	20	24	24	24	28	28	28	28
20	2C	2C	2C	2C	30	30	30	30	34	34	34	34	34	38	38	38
30	38	38	38	38	38	3C	3C	3C	3C	3C	3C	3C	3C	3C	3C	3C

Calculate waveform

Sigma: 0.000000C

Offset: 0.0

Figure 6.16: Microstep wave form setting (table view)

6.2.6 Other

This page provides the *Firmware Revision*, the *Configuration EEPROM*, and the *TMCL™ auto start*.

1. Firmware revision:
   Click the *Get* button in the Firmware Revision section to read the firmware revision of a module. The result will be displayed beside the *Get* button.

2. Configuration EEPROM:
   Here you can lock or unlock the configuration EEPROM. When the configuration EEPROM is locked, STAP and STGP commands do not have any effect so that the settings stored in the configuration EEPROM cannot be changed by accident. The buttons in the *Configuration EEPROM section* provide the following functionality:

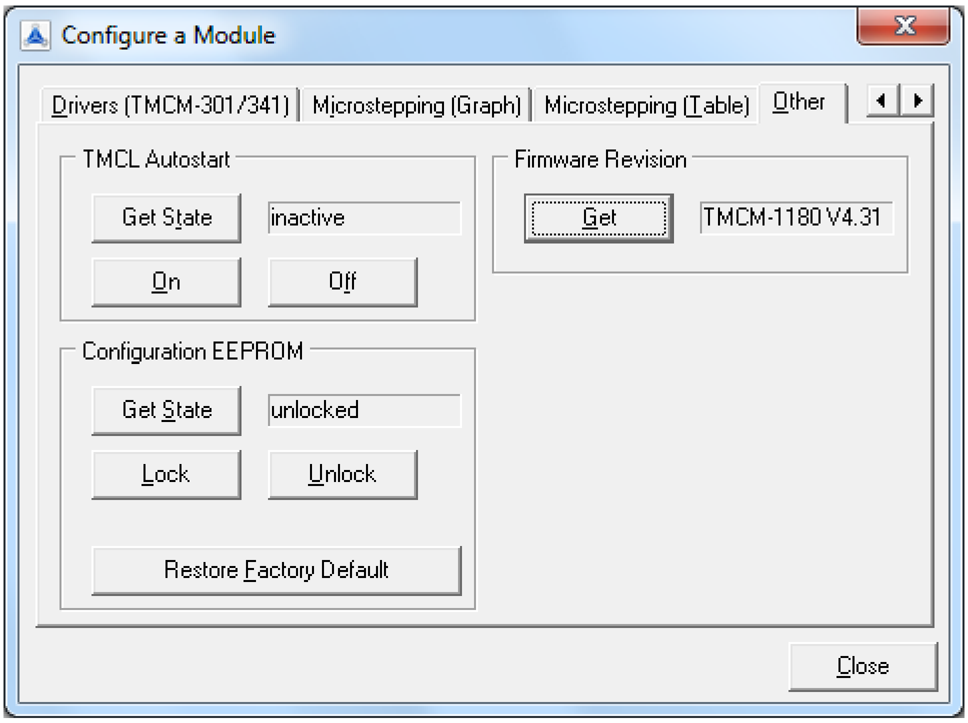
   Get State: Click this button to see if the configuration EEPROM is locked. The state will then be displayed beside the button.

   Lock: Click this button to lock the configuration EEPROM.

   Unlock: Click here to unlock the configuration EEPROM.

   Restore Factory Default: Clicking this button will restore all settings stored in the configuration EEPROM to their factory defaults and then reset the module. Please use this function with extreme care. If this function should not work, try to use the hardware reset option of the module.

3. TMCL auto start:
   The TMCL™ program auto start option can be turned on or off by clicking the appropriate button. Click the *Get State* button to see whether auto start is turned on or off on a module.



inactive

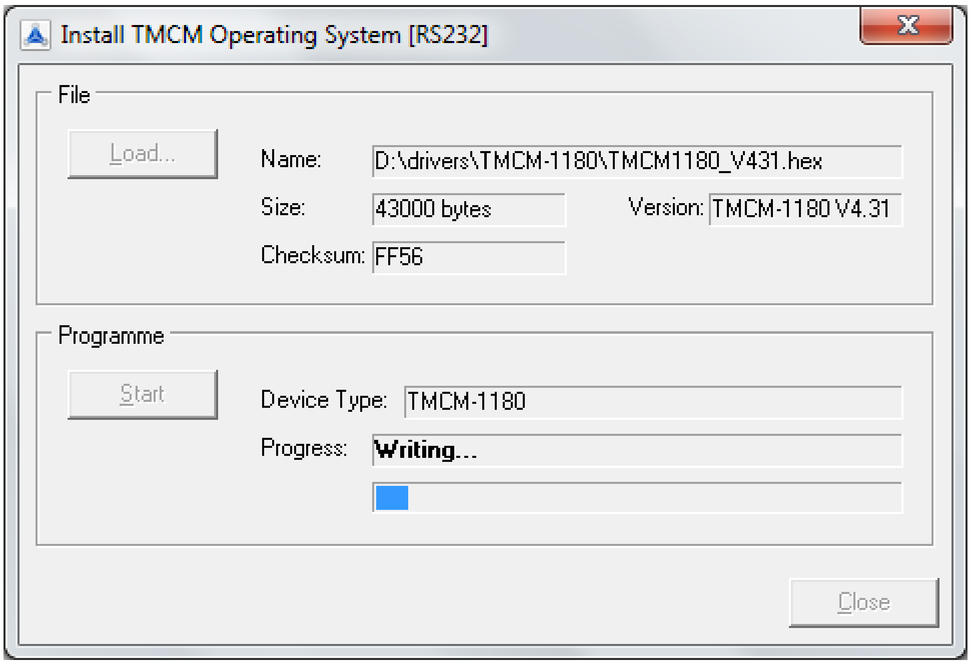
unlocked

TMCM-1180 V4.31

Figure 6.17: Other module settings

6.3 Install OS

By using this function you can update the firmware of a module. First, load the new firmware file (which must be in Intel-HEX format) by clicking the *Load* button. The file is then checked if it is a TMCL™ firmware file, and its device type and version number will be displayed. Then, click the *Start* button to program the new firmware into the module. Please make sure that there will be no power cut or cut of the serial connection during the programming process. The program checks if the device type in the firmware file and the device type of the module are identical. An error message will be displayed if this is not the case. If everything is okay, the new firmware will be programmed into the module and verified afterwards. The programming progress is shown by a status bar.



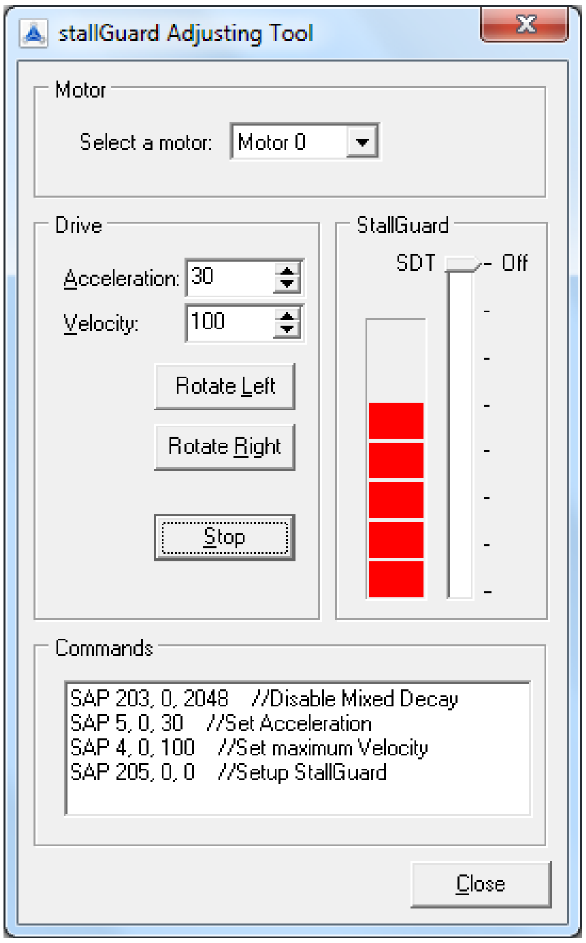
Name: D:\drivers\TMCM-1180\TMCM1180_V431.hex
Size: 43000 bytes
Version: TMCM-1180 V4.31
Checksum: FF56
Device Type: TMCM-1180
Progress: Writing...

Figure 6.18: Firmware update in progress

6.4 StallGuard™ adjusting tool

Figure 6.19: *stallGuard™ adjusting tool* icon

The stallGuard™ adjusting tool helps to find the necessary motor parameters when stallGuard™ is to be used. This function can only be used when a module is connected that features stallGuard™. This will be automatically checked when the stallGuard™ adjusting tool is selected in the *Setup menu*. If the test is carried out successfully the stallGuard™ adjusting tool will be displayed.



```
SAP 203, 0, 2048   //Disable Mixed Decay
SAP 5, 0, 30   //Set Acceleration
SAP 4, 0, 100   //Set maximum Velocity
SAP 205, 0, 0   //Setup StallGuard
```

Figure 6.20: stallGuard adjusting tool

<u>Using the stallGuard™ adjusting tool:</u>

1. First, select the axis that is to be used in the *Motor area*.
2. Now you can enter a *Velocity* and an *Acceleration value* in the *Drive area* and then click *Rotate Left* or *Rotate Right*. Clicking one of these buttons will send the necessary commands to the module so that the motor starts running.
3. The red bar in the *stallGuard™ area* on the right side of the windows displays the actual load value. Use the *slider* to set the stallGuard™ threshold value. If the load value reaches this value the motor stops.
4. Clicking the *Stop* button also stops the motor.

The commands which are necessary for setting the chosen values (entered in this dialogue) are displayed in the *Commands area* at the bottom of the window. They can be selected, copied and pasted into the *TMCL™ editor*.

6.5 stallGuard™ profiler

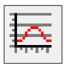
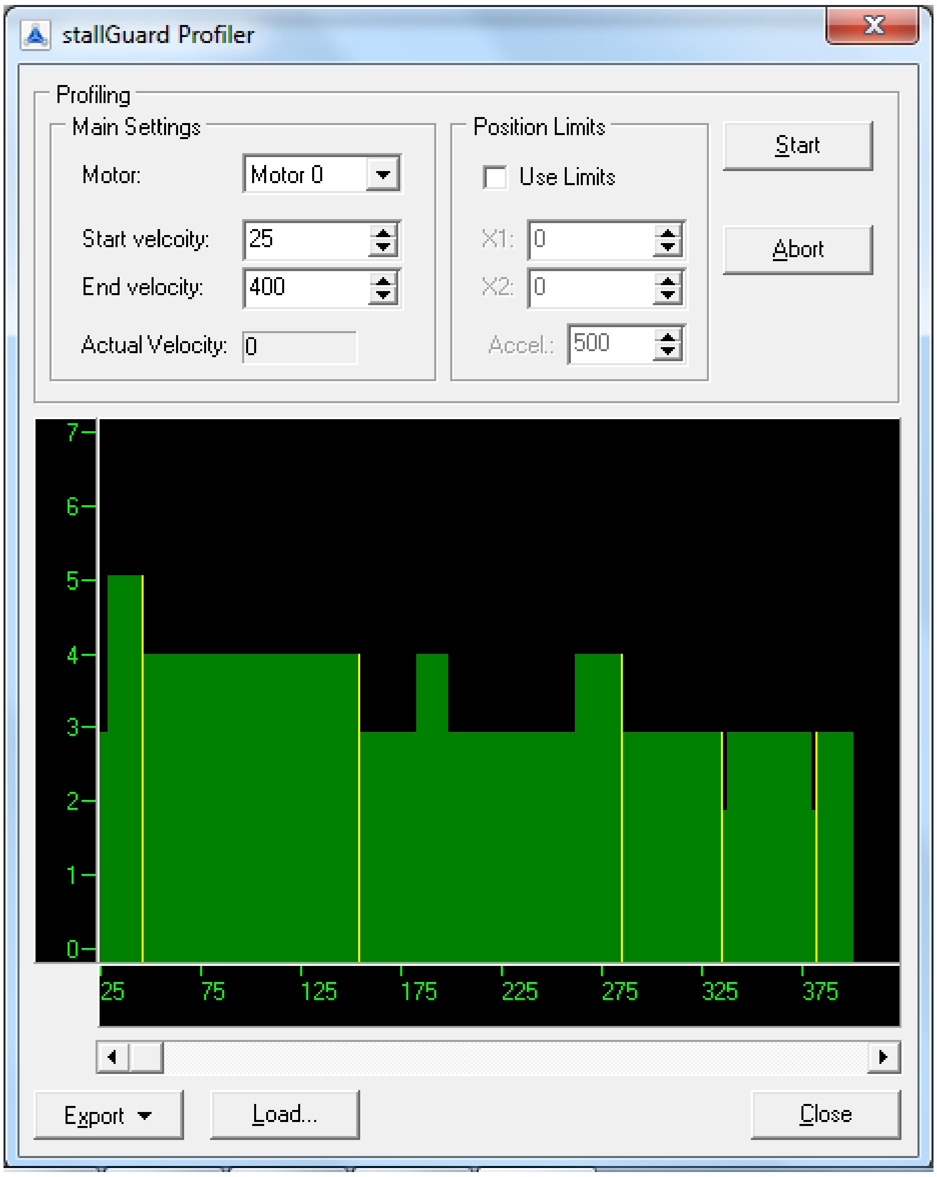


Figure 6.21: stallGuard™ profiler icon

The stallGuard™ profiler is a utility that helps you to find the best parameters for using stall detection. It scans through given velocities and shows which velocities are the best ones. Similar to the stallGuard™ adjusting tool it can only be used together with a module that supports stallGuard™. This is checked right after the *stallGuard™ profiler* has been selected in the *Setup menu*. After this has been successfully checked the stallGuard™ profiler window will be shown.



stallGuard Profiler

Profiling

Main Settings

Motor: Motor 0

Start velcoity: 25

End velocity: 400

Actual Velocity: 0

Position Limits

☐ Use Limits

X1: 0

X2: 0

Accel.: 500

Figure 6.22: The stallGuard™ profiler

Using the stallGuard™ profiler:

1. First, select the axis that is to be used.
2. Then, enter the *Start velocity* and the *End velocity*. The start velocity is used at the beginning at the profile recording. The recording ends when the end velocity has been reached. Start velocity and end velocity must not be equal.
3. After you have entered these parameters, click the *Start* button to start the stallGuard™ profile recording. Depending on the range between start and end velocity this can take several minutes, as the load value for every velocity value is measured ten times.
4. The *Actual velocity* value shows the velocity that is currently being tested and so tells you the progress of the profile recording.
5. You can also abort a profile recording by clicking the *Abort* button.
6. The result can also be exported to Excel or to a text file by using the *Export* button.

6.5.1 The result of the stallGuard™ profiler

The result is shown as a graphic in the stallGuard™ profiler window. After the profile recording has finished you can scroll through the profile graphic using the scroll bar below it. The scale on the vertical axis shows the load value: A higher value means a higher load. The scale on the horizontal axis is the velocity scale. The color of each line shows the standard deviation of the ten load values that have been measured for the velocity at that point. This is an indicator for the vibration of the motor at the given velocity.

There are three colors used:

🟩	The standard deviation is very low or zero. This means that there is effectively no vibration at this velocity.
🟥	The red color means that there is high vibration at that velocity.
🟨	Yellow: This color means that there might be some low vibration at this velocity.

6.5.2 Interpreting the result

In order to make effective use of the stallGuard™ feature you should choose a velocity where the load value is as low as possible and where the color is green. The very best velocity values are those where the load value is zero (areas that do not show any green, yellow or red line).

Velocities shown in yellow can also be used, but with care as they might cause problems (maybe the motor stops even if it is not stalled).

Velocities shown in red should not be chosen. Because of vibration the load value is often unpredictable and so not usable to achieve good results when using stall detection.

As it is very seldom that exactly the same result is produced when recording a profile with the same parameters a second time, always two or more profiles should be recorded and compared against each other.

6.6 stallGuard2™ & coolStep™ adjusting tool

Figure 6.23: *stallGuard2™ & coolStep™ adjusting tool* icon

The *stallGuard2™ & coolStep™ Adjusting Tool* helps to find necessary motor parameter adjustments when stallGuard2™ and coolStep™ are to be used. These functions are only provided when a module is connected that features stallGuard2™ and coolStep™ (modules equipped e.g. with TMC260, TMC261, TMC262, and TMC389). This will be automatically checked when the *stallGuard2™ & coolStep™ Adjusting Tool* is selected in the *Setup menu*. If the test is carried out successfully the tool will be displayed.

<u>There are four tabs which concern different aspects of the features:</u>
- motor
- stallGuard2™
- coolStep™
- TMCL™ code

The *stallGuard2™ & coolStep™ Adjusting Tool* of the TMCL-IDE provides an interactive way for adjusting the module. All settings which have been tried out with the help of the *stallGuard2™ tab* and the *coolStep™ tab* are immediately represented as TMCL™ mnemonics on the *TMCL™ code tab*.

stallGuard2™ and coolStep™ parameters have to be adjusted with a special command of the TMCL-IDE, the *SAP* (set axis parameter) command. With this command most of the motion control parameters of the module can be specified. The settings will be stored in SRAM and therefore are volatile. That is, information will be lost after power off. For storing any setting permanently, please use command STAP (store axis parameter). If you want to read out your parameter calibrations, use the GAP command. Please refer to the special TMCL™ Firmware Manual for your module for further information about writing TMCL™ commands.

<u>The SAP command has the following mnemonic:</u>

SAP <parameter number>, <motor number>, <value>

6.6.1 Motor tab

The motor tab concerns four commands of the TMCL-IDE:

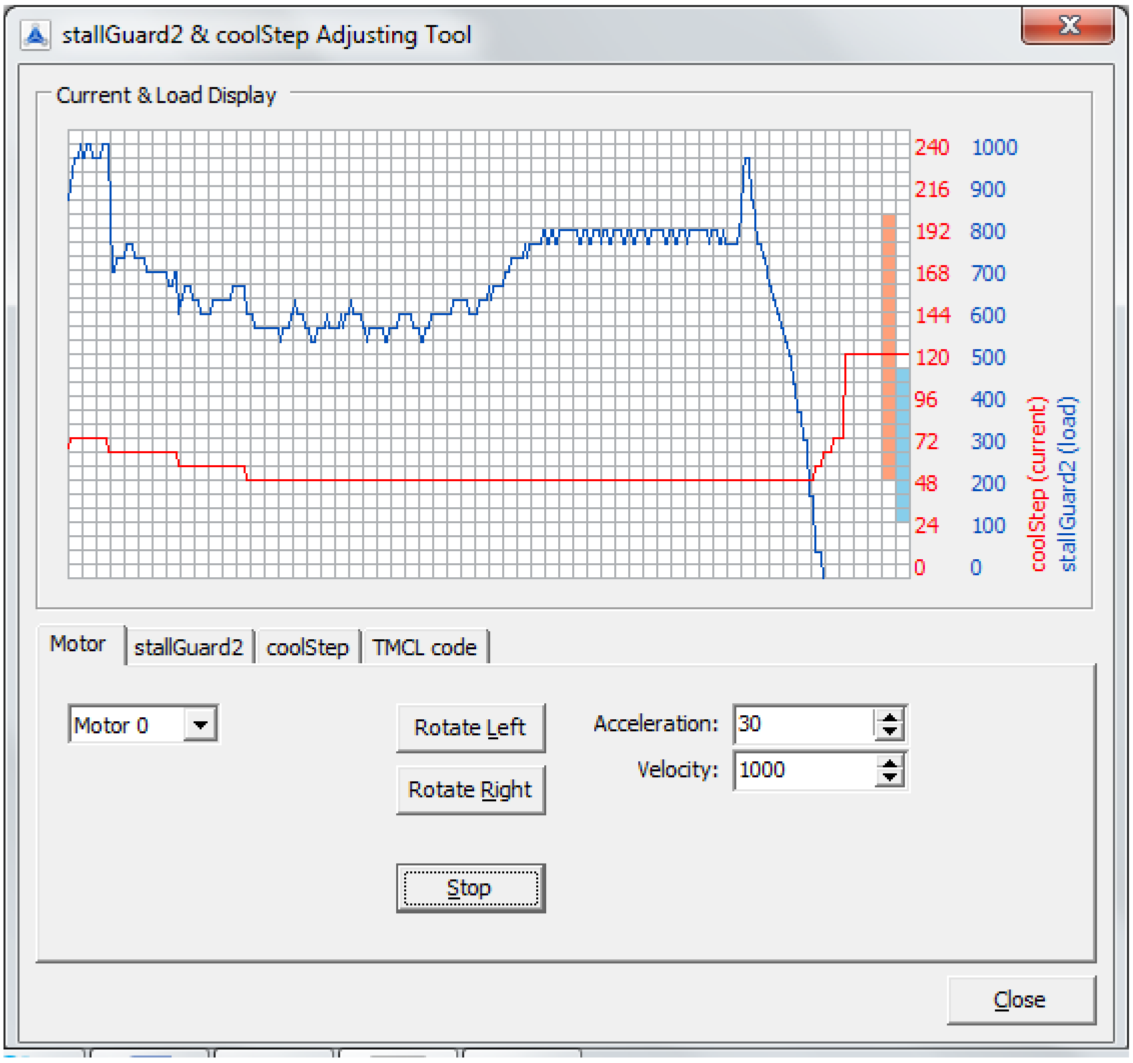
ROR (rotate right): With this command the motor will be instructed to rotate with a specified velocity in *right* direction (increasing the position counter). Please specify the velocity in the *Velocity field*.

ROL (rotate left): With this command the motor will be instructed to rotate with a specified velocity (opposite direction compared to ROR, decreasing the position counter). Please specify the velocity in the *Velocity field*.

MST (motor stop): With this command the motor will be instructed to stop. (Depending on the motor speed a hard stop might lead to step losses.)

SAP (set axis parameter): With this command most of the motion control parameters of the module can be specified. On the *Motor tab*, the acceleration (axis parameter 5) is set. Just fill in the *Acceleration field*.

All settings will be represented as mnemonics on the TMCL™ code tab immediately.



Acceleration: 30
Velocity: 1000

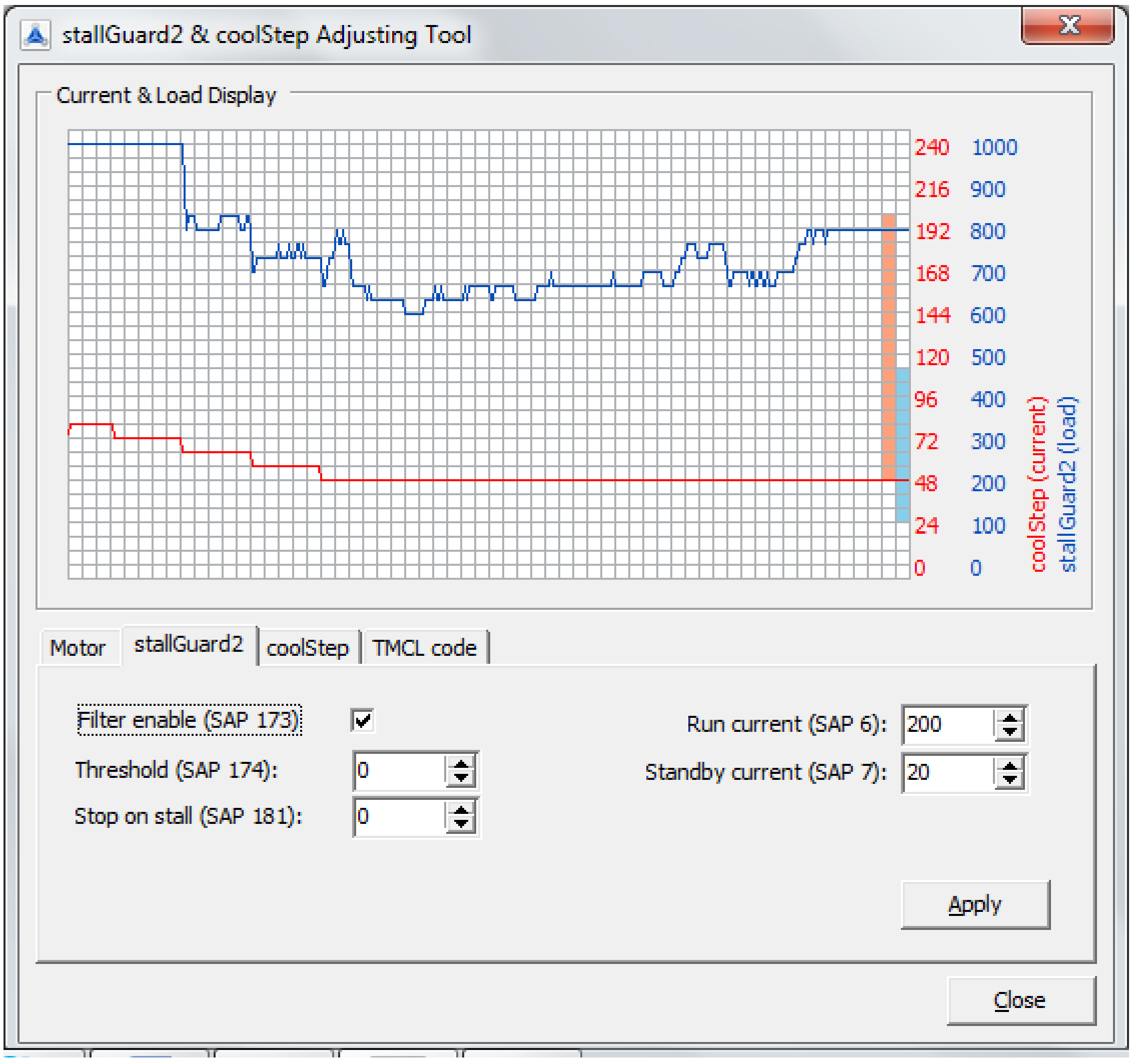
Figure 6.24: stallGuard2™ and coolStep™ adjusting tool

Using the motor tab:

- Select the *axis* that is to be used in the *Motor field*.
- Select *acceleration* and *velocity*.
- The buttons *Rotate Left*, *Rotate Rig**ht***, and S*top* are for testing purposes. Start and stop your drive as you like.
- Now, adjust the parameters for stallGuard2™ and coolStep™ with the other tabs of the *stallGuard2™ and coolStep™ Adjusting Tool*. Thereafter run your application with the help of the *Motor tab* to find the best fitting values.

6.6.2 stallGuard2™ tab

The *stallGuard2™ tab* offers the possibility to set stallGuard2™ adjustments on the fly. This tab of the *stallGuard2™ & coolStep™ Adjusting Tool* enables the user to check several adjustments and to try out which is the best fitting value of each axis parameter for the intended use of the drive. The *Current & Load Display* shows the applied changes of parameter values immediately. Certainly, every change of parameters will be documented as written TMCL™ commands on the *TMCL™ code tab*.



Filter enable (SAP 173): ✓
Threshold (SAP 174): 0
Stop on stall (SAP 181): 0
Run current (SAP 6): 200
Standby current (SAP 7): 20

Using the stallGuard2™ tab:

Realize that every change of parameters bases on the *SAP* (Set Axis Parameter) command of the TMCL-IDE. Please refer to the TMCL™ Firmware Manual of your module and read more about this command. A short information is given in the beginning of this chapter (6.6 stallGuard2™ & coolStep™ adjusting tool).

All chosen settings will be shown immediately as graphical view on the *Current and Load Display* and as mnemonics on the *TMCL™ code tab*.

For changing parameter values write your chosen value in the specific *value field* und click the ***Apply*** button afterwards. It is possible to change several parameter values before clicking the ***Apply*** button, but we recommend trying out new adjustments step by step for a better screening of their effects.

The following axis parameters are used for calibrating stallGuard2™:

Filter enable (SAP 173)

Setting this parameter enables the stallGuard2™ filter for more precision of the measurement. If set, the measurement frequency will be reduced to one measurement per four fullsteps. In most cases it is expedient to set the filtered mode before using coolStep2™.
Value range: 0/1

stallGuard2™ threshold (SAP 174)

This signed value controls the stallGuard2™ threshold level for stall output and sets the optimum measurement range for readout. A lower value gives a higher sensitivity. Zero is the starting value. A higher value makes stallGuard2™ less sensitive and requires more torque to indicate a stall.

0	Indifferent value
1… 63	less sensitivity
-1… -64	higher sensitivity

Stop on stall (SAP 181)

Below this speed the motor will not be stopped. Above this speed the motor will stop in case the stallGuard2™ load value reaches zero.
Value range: the range is module-specific and conforms to other speed parameters of your module. Please refer to the Firmware Manual of your module.

Run current / current scale (SAP 6)

This is the most important motor setting, since too high values might cause motor damage.
Value range: 0… 255, can be adjusted in 32 steps. The maximum value is 255 (this value means 100% of the maximum current of the module).

0… 7	79…87	160… 167	240… 247
8… 15	88… 95	168… 175	248… 255
16… 23	96… 103	176… 183	
24… 31	104… 111	184… 191	
32… 39	112… 119	192… 199	
40… 47	120… 127	200… 207	
48… 55	128… 135	208… 215	
56… 63	136… 143	216… 223	
64… 71	144… 151	224… 231	
72… 79	152… 159	232… 239	

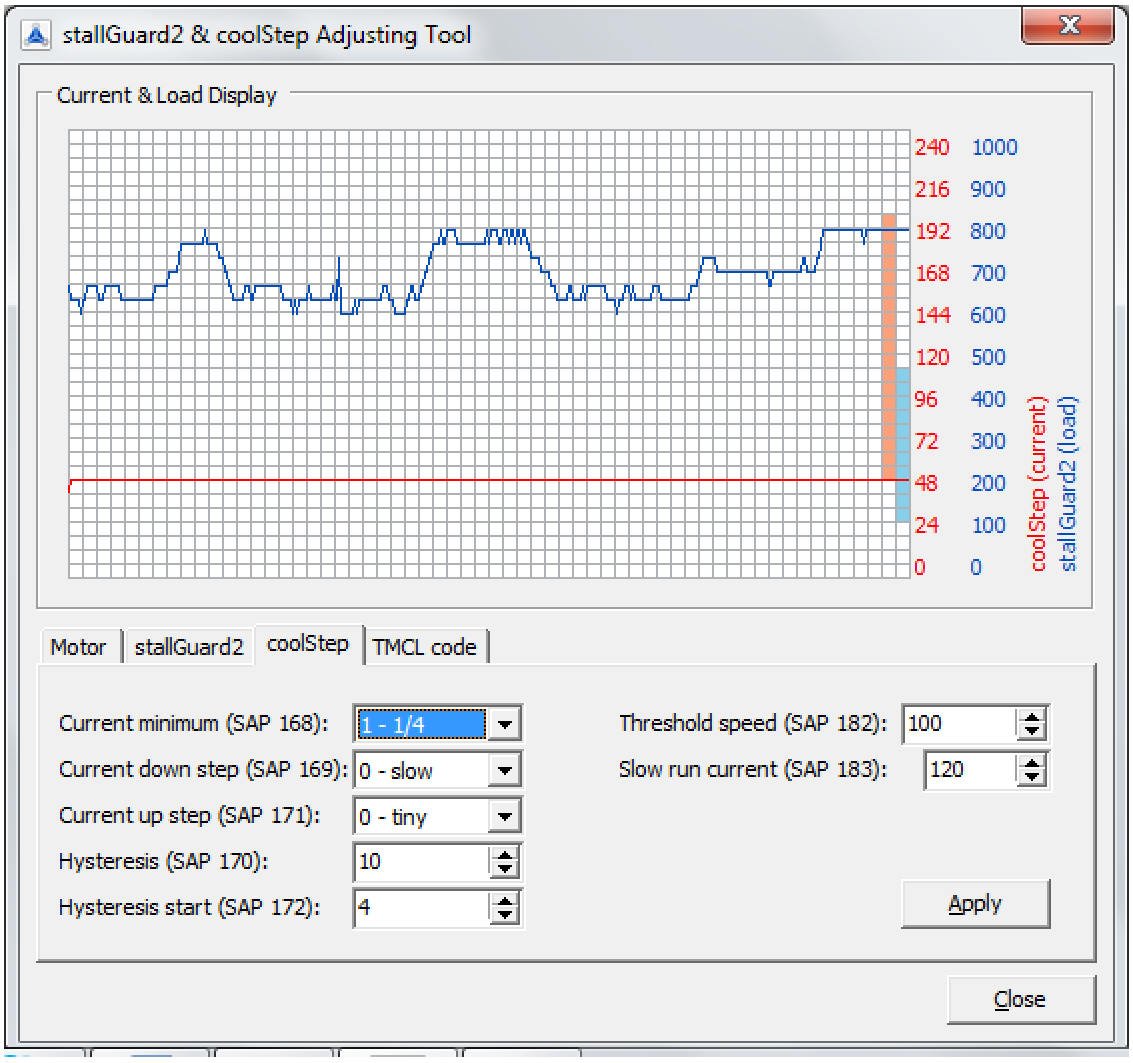
Standby current (SAP 7)

The current limit two seconds after the motor has stopped
Value range: parallel to axis parameter 6, see above.

6.6.3 coolStep™ tab

The *coolStep™ tab* offers the possibility to set coolStep™ adjustments on the fly. This tab of the *stallGuard2™ & coolStep™ Adjusting Tool* enables the user to check several adjustments and to try out which is the best fitting value of each axis parameter for the intended use of the drive.

The *Current & Load Display* shows the applied changes of parameter values immediately. Certainly, every change of parameters will be documented as written TMCL™ commands on the *TMCL™ code tab*.



Current minimum (SAP 168): 1 - 1/4
Current down step (SAP 169): 0 - slow
Current up step (SAP 171): 0 - tiny
Hysteresis (SAP 170): 10
Hysteresis start (SAP 172): 4
Threshold speed (SAP 182): 100
Slow run current (SAP 183): 120

Using the coolStep™ tab:

Realize that every change of parameters bases on the *SAP* (Set Axis Parameter) command of the TMCL-IDE. Please refer to the TMCL™ Firmware Manual of your module and read more about this command. A short information is given in the beginning of this chapter (6.6 stallGuard2™ & coolStep™ adjusting tool).

All chosen settings will be shown immediately as graphical view on the *Current and Load Display* and as mnemonics on the *TMCL™ code tab*.

For changing parameter values write your chosen value in the specific *value field* und click the ***Apply*** button afterwards. It is possible to change several parameter values before clicking the ***Apply*** button, but we recommend trying out new adjustments step by step for a better screening of their effects.

The following axis parameters are used for calibrating coolStep™:

Current minimum (SAP 168)

This parameter sets the lower motor current limit for coolStep™ operation by scaling the CS value (current scale, see axis parameter 6).

Value	Minimum motor current
0	1/2 of CS
1	1/4 of CS

Current down step (SAP 169)

This parameter sets the number of stallGuard2™ readings above the upper threshold necessary for each current decrement of the motor current.

Number of stallGuard2™ measurements per decrement:

Value	stallGuard2™ readings
0	32 (slow decrement)
1	8
2	2
3	1 (fast decrement)

Current up step (SAP 171):

This parameter sets the current increment step. The current becomes incremented for each measured stallGuard2™ value below the lower threshold (see also below: *hysteresis start*; SAP 172).

Current increment step size:

Value	Reaction to rising load in steps
0	1 (slow increment)
1	2
2	4
3	8 (fast increment)

Hysteresis (SAP 170):

Sets the distance between the lower and the upper threshold for stallGuard2™ reading. Above the upper threshold the motor current becomes decreased.
Value range: 0… 15

Hysteresis:
(smartEnergy hysteresis value + 1) * 32

Upper stallGuard2™ threshold:
(smartEnergy hysteresis start + smartEnergy hysteresis + 1) * 32

Hysteresis start (SAP 172):

This parameter sets the lower threshold for the stallGuard2™ value (see *Current up step*; SAP 171).
Value range: 0…15

Threshold speed (SAP 182):

Above this speed coolStep™ becomes enabled.
Value range: 0… 2047

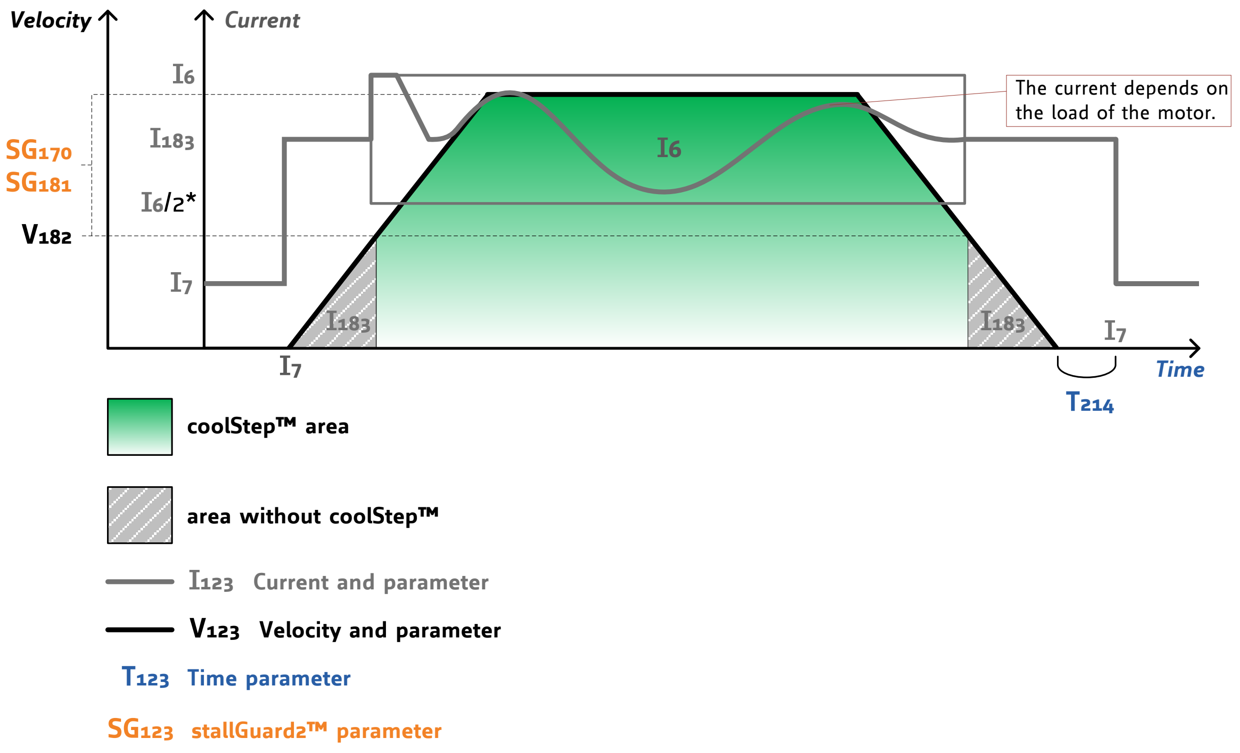
Slow run current (SAP 183):
This parameter sets the motor current which is used below the threshold speed.
Value range: 0… 255 (parallel to axis parameter 6 and 7; can be adjusted in 32 steps)

0… 7	79…87	160… 167	240… 247
8… 15	88… 95	168… 175	248… 255
16… 23	96… 103	176… 183	
24… 31	104… 111	184… 191	
32… 39	112… 119	192… 199	
40…47	120… 127	200… 207	
48… 55	128… 135	208… 215	
56… 63	136… 143	216… 223	
64… 71	144… 151	224… 231	
72… 79	152… 159	232… 239	

6.6.3.1 coolStep™ and stallGuard2™ adjustment / example

It is necessary to identify and configure the thresholds for current (axis parameters 6, 7, and 183) and velocity (axis parameter 182). Furthermore the stallGuard2™ feature has to be adjusted and enabled (axis parameters 170 and 181). The reduction or increasing of current in the coolStep™ area of the figure below (depending on the load) has to be configured with parameters 169 and 171. Parameter 168 sets the lower motor current limit for coolStep™ operation by scaling the CS value. Please adjust all parameters with the ***stallGuard2™ and coolStep™ Adjusting Tool***.

coolStep™ adjustment points and thresholds


Velocity
Current
I6
I183
I6/2*
I7
SG170
SG181
V182
I6
The current depends on the load of the motor.
I183
I183
I7
I7
Time
T214
coolStep™ area
area without coolStep™
I123 Current and parameter
V123 Velocity and parameter
T123 Time parameter
SG123 stallGuard2™ parameter

* The lower threshold of the coolStep™ current can be adjusted up to I6/4. Refer to parameter 168.

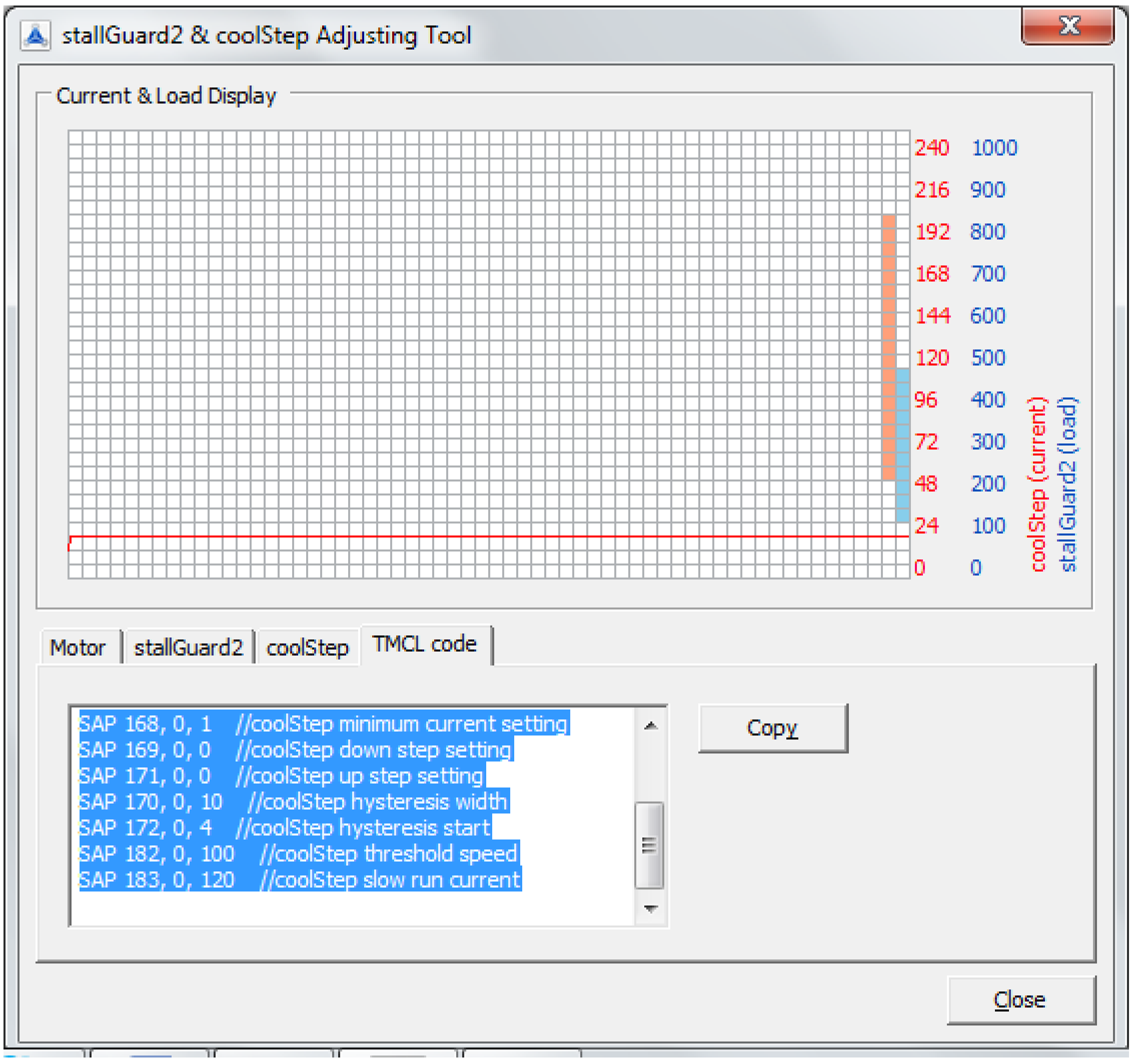
Refer to the specific TMCL™ Firmware Manual for your TMCM module, too. Here you will find detailed information about e.g. the *Set Axis Parameter* command, special parameters and their ranges for your TMCM module. Here, only basic parameters for coolStep™ and stallGuard2™ are mentioned.

6.6.4 TMCL™ code tab

The *TMCL™ code tab* shows the adjustments which have been made on the other tabs of the *stallGuard2™ & coolStep™ Adjusting Tool*. In the output text box all settings are documented as written TMCL™ commands.

Use the commands in the text box for drawing up your TMCL™ program in the editor:

Mark the interesting lines, click the ***Copy*** button, close the window, and paste the chosen commands in the editor. Add all commands you need.



```
SAP 168, 0, 1    //coolStep minimum current setting
SAP 169, 0, 0    //coolStep down step setting
SAP 171, 0, 0    //coolStep up step setting
SAP 170, 0, 10    //coolStep hysteresis width
SAP 172, 0, 4    //coolStep hysteresis start
SAP 182, 0, 100    //coolStep threshold speed
SAP 183, 0, 120    //coolStep slow run current
```

6.7 Parameter calculation tool

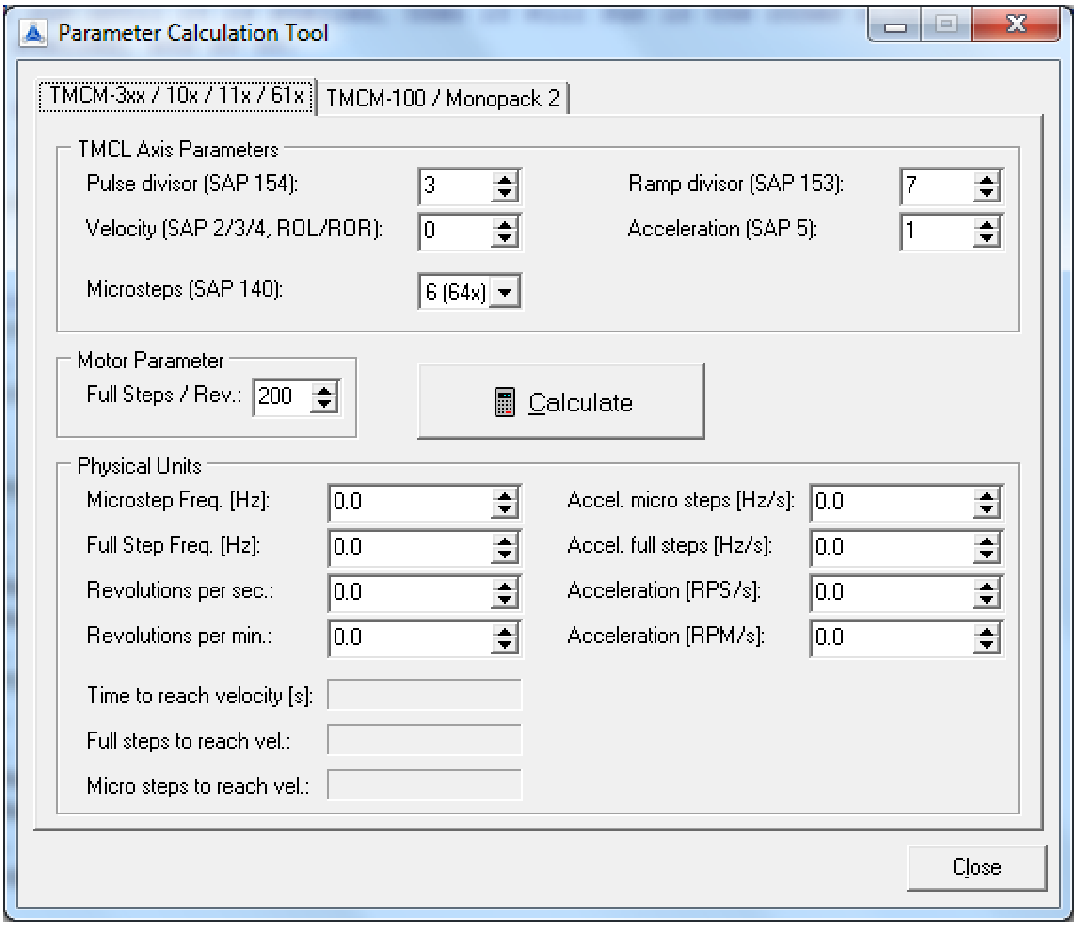
The parameter calculation tool helps you to calculate the velocity and acceleration parameters in TMCL™. Parameters can be converted from physical units like rpm or rps into the internal units of TMCL™ and vice versa.

TMCL-IDE offers actually two parameter calculation tools:

- One parameter calculation tool for all TMC428 / TMC429 based modules.
- Another parameter calculation tool for the TMCM-100 module and the MONOpack 2, which are TMC453 based.

You can choose the tool by selecting the appropriate tab page in the parameter calculation dialogue. *Always be sure to use the right one. In case of doubt please refer to your Hardware Manual.*

To use the calculation tool, just fill in the values that are known and then click the *Calculate* button. After changing any parameter always click it again. When a parameter in the TMCL™ section has been changed the physical units will be recalculated with the next click on the *Calculate* button. When a parameter in the physical units section has been changed the TMCL™ parameters will be recalculated with the next click on the *Calculate* button also.



TMCL Axis Parameters

Pulse divisor (SAP 154): 3
Ramp divisor (SAP 153): 7
Velocity (SAP 2/3/4, ROL/ROR): 0
Acceleration (SAP 5): 1
Microsteps (SAP 140): 6 (64x)

Motor Parameter

Full Steps / Rev.: 200

Physical Units

Microstep Freq. [Hz]: 0.0
Accel. micro steps [Hz/s]: 0.0
Full Step Freq. [Hz]: 0.0
Accel. full steps [Hz/s]: 0.0
Revolutions per sec.: 0.0
Acceleration [RPS/s]: 0.0
Revolutions per min.: 0.0
Acceleration [RPM/s]: 0.0
Time to reach velocity [s]:
Full steps to reach vel.:
Micro steps to reach vel.:

Figure 6.25: Parameter calculation tool

Information for TMC457 based modules:
Please note, that there is actually no TMCL-IDE calculation tool for TMC457 based modules. For calculating parameters please refer to our TechLibCD or www.trinamic.com and enter the file tmc457_calculations.xls. You will find it next to the TMC457 datasheet. Furthermore, physical units (pos) can directly be used on TMC457 based modules (e.g. TMCM-142, PD-146).

7 Syntax of TMCL™ in the assembler

Here, the syntax of the TMCL™ commands used by the TMCL™ assembler is given. Please refer to the specific Firmware Manual of your module for getting explanations concerning the functionality of the TMCL™ commands. The command mnemonics given there are used in the TMCL™ assembler. Please see also the sample program files of the TMCL-IDE and the information about TMCL™ programming in the Firmware Manual.

7.1 Assembler directives

An assembler directive starts with the **#** sign, and the only directive is *#include* to include a file. The name of that file must be given after the *#include* directive. If that file is already in the editor, it will be taken from there. Otherwise it will be loaded from file, using the include file path that can be set in the *Options dialogue*.

Example:
#include test.tmc

7.2 Symbolic constants

Symbolic constants are defined using the following syntax:

```
<Name>=<Value>
```

A *Name* must always start with a letter or the sign _ and may then contain any combination of letters, numbers and the sign _.

A *Value* must always be a decimal, hexadecimal or binary number or a constant expression. Hexadecimal numbers start with a $ sign, binary numbers start with a % sign.

Examples:
Speed=1000
Speed2=Speed/2
Mask=$FF
Binary Value=%1010101

7.3 Constant expressions

Wherever a numerical value is needed, it can also be calculated during assembly. For this purpose constant expressions can be used. A constant expression is just a formula that evaluates to a constant value. The syntax is very similar to BASIC or other programming languages.

The calculating will be done during the assembly.

Internally, the assembler uses floating point arithmetic to evaluate a constant expression, but as TMCL™ commands only take integer values, the result of a constant expression will always be rounded to an integer value when used as an argument to a TMCL™ command.

Functions, which can be used in constant expressions:

Name	Function
SIN	Sinus
COS	Cosinus
TAN	Tangens
ASIN	Arcus Sinus
ACOS	Arcus Cosinus
ATAN	Arcus Tangens
LOG	Logarithm Base 10
LN	Logarithm Base e
EXP	Power to Base e
SQRT	Square root
ABS	Absolute value
INT	Integer (truncate)
ROUND	Integer (Round)
SIGN	Returns -1 if argument<1, 0 if argument=0 1 if argument>0
DEG	Converts from radiant to degrees
RAD	Converts from degrees to radiant

Operators, which can be used in constant expressions:

Symbol	Meaning
()	Parenthesis
^	Power
*	Multiplication
/	Division
+	Addition
-	Subtraction

Symbolic constants, floating point numbers, integer numbers, hexadecimal numbers and binary numbers can also be used in constant expressions.

Here are some examples of constant expressions used wherever constant values can be placed:

```
ROL 0, 7+9*8
Speed2=Speed*SIN(0.5)
MVP ABS, 0, 3*1000
Sin90=Sin(Rad(90))
```

7.4 Labels

Labels have the form following form:

```
<Name>:
```

The same rules are valid for label names as for symbolic constants.

Example (the label has the name ***Loop***):

```
Loop:   MVP ABS, 0, 1000
        WAIT POS, 0, 0
        MVP ABS, 0, 0
        WAIT POS, 0, 0
        JA Loop
```

7.5 Comments

Comments always start with // (like in C++). The rest of the line is then ignored.

7.6 TMCL™ commands

Here is a short list of all command mnemonics that are recognized by the assembler. Please refer to the specific firmware manual of your module for a detailed explanation of every command.

```
ROR <n1>, <n2>
ROL <n1>, <n2>
MST <n1>
MVP <mvp_opt>, <n1>, <n2>
SAP <n1>, <n2>, <n3>
GAP <n1>, <n2>
STAP <n1>, <n2>
RSAP <n1>, <n2>
SGP <n1>, <n2>, <n3>
GGP <n1>, <n2>
STGP <n1>, <n2>
RSGP <n1>, <n2>
RFS <rfs_opt>, <n1>
SIO <n1>, <n2>, <n3>
GIO <n1>, <n2>
CALC <op1>, <n2>, <n3>
CALCX <op2>, <n2>, <n3>
COMP <n1>
JC <cc>, <Label>
JA <Label>
CSUB <Label>
RSUB
WAIT <Event>, <n1>, <n2>
STOP
SAC <n1>, <n2>, <n3>
SCO <n1>, <n2>, <n3>
GCO <n1>, <n2>
CCO <n1>, <n2>
ACO <n1>, <n2>
AAP <n1>, <n2>
AGP <n1>, <n2>
CLE <Flag>
VECT <Interrupt number>, <Label>
EI <Interrupt number>
DI <Interrupt number>
RETI
```

With:

<n1>, <n2>, <n3>: Any numerical value, constant expression or symbolic constant
<mvp_opt>: An option for MVP: ABS, REL or COORD.
<rfs_opt>: An option for RFS: START, STOP or STATUS.
<cc>: A condition code: ZE, NZ, EQ, NE, GT, GE, LT, LE, ETO, EAL, EDV, EPO.
<Event>: A wait event. This can be TICKS, POS, LIMSW, REFSW or RFS.
<op1>: An operator for the CALC command: ADD,SUB,MUL,DIV,MOD,AND,OR,NOT,LOAD
<op2>: An operator for the CALCX command: ADD,SUB,MUL,DIV,MOD,AND,OR,NOT,LOAD,SWAP
<Label>: A label defined somewhere else in the program.
<Flag>: An error flag code: ALL, ETO, EAL, EDV or EPO.
<Interrupt number>:

Number	Interrupt type
0	Timer 0
1	Timer 1
2	Timer 2
3	Target position reached
15	stallGuard™
21	Deviation
27	Left stop switch
28	Right stop switch
39	Input change 0
40	Input change 1

8 TMCL™ programming techniques and structure

8.1 Initialization

The first task in a TMCL™ program (like in other programs also) is to initialize all parameters where different values than the default values are necessary. For this purpose, SAP and SGP commands are used.

8.2 Main loop

Embedded systems normally use a main loop that runs infinitely. This is also the case in a TMCL™ application that is running stand alone. Normally the auto start mode of the module should be turned on. After power up, the module then starts the TMCL™ program, which first does all necessary initializations and then enters the main loop, which does all necessary tasks end never ends (only when the module is powered off or reset).

There are exceptions to this, e.g. when TMCL™ routines are called from a host in direct mode.

<u>So most (but not all) stand alone TMCL™ programs look like this:</u>

```
//Initialization
      SAP 4, 0, 500  //define max. positioning speed
      SAP 5, 0, 100  //define max. acceleration

MainLoop:
      //do something, in this example just running between two positions
      MVP ABS, 0, 5000
      WAIT POS, 0, 0
      MVP ABS, 0, 0
      WAIT POS, 0, 0
      JA MainLoop       //end of the main loop => run infinitely
```

8.3 Using symbolic constants

To make your program better readable and understandable, symbolic constants should be taken for all important numerical values that are used in the program. The TMCL-IDE provides an include file with symbolic names for all important axis parameters and global parameters.

Example:

```
//Define some constants
#include TMCLParam.tmc
MaxSpeed = 500
MaxAcc = 100
Position0 = 0
Position1 = 5000

//Initialization
      SAP APMaxPositioningSpeed, Motor0, MaxSpeed
      SAP APMaxAcceleration, Motor0, MaxAcc

MainLoop:
      MVP ABS, Motor0, Position1
      WAIT POS, Motor0, 0
      MVP ABS, Motor0, Position0
      WAIT POS, Motor0, 0
      JA MainLoop
```

Just have a look at the file TMCLParam.tmc provided with the TMCL-IDE. It contains symbolic constants that define all important parameter numbers.

Using constants for other values makes it easier to change them when they are used more than once in a program. You can change the definition of the constant and do not have to change all occurrences of it in your program.

8.4 Using variables

The *User Variables* can be used if variables are needed in your program. They can store temporary values. The commands SGP, GGP and AGP are used to work with user variables:

SGP is used to set a variable to a constant value (e.g. during initialization phase).

GGP is used to read the contents of a user variable and to copy it to the accumulator register for further usage.

AGP can be used to copy the contents of the accumulator register to a user variable, e.g. to store the result of a calculation.

Example:

```
MyVariable = 42
      //Use a symbolic name for the user variable
      //(This makes the program better readable and understandable.)

SGP MyVariable, 2, 1234   //Initialize the variable with the value 1234
...
...
GGP MyVariable, 2         //Copy the contents of the variable to the
accumulator register
CALC MUL, 2                //Multiply accumulator register with two
AAP MyVariable, 2         //Store contents of the accumulator register to the
variable
...
...
```

Furthermore, these variables can provide a powerful way of communication between a TMCL™ program running on a module and a host. The host can change a variable by issuing a direct mode SGP command (remember that while a TMCL™ program is running direct mode commands can still be executed, without interfering with the running program). If the TMCL™ program polls this variable regularly it can react on such changes of its contents.

The host can also poll a variable using GGP in direct mode and see if it has been changed by the TMCL™ program.

8.5 Using subroutines

The *CSUB* and *RSUB* commands provide a mechanism for using subroutines. The *CSUB* command branches to the given label. When an *RSUB* command is executed the control goes back to the command that follows the *CSUB* command that called the subroutine.

This mechanism can also be nested. From a subroutine called by a *CSUB* command other subroutines can be called. In the current version of TMCL™ eight levels of nested subroutine calls are allowed.

8.6 Mixing direct mode and stand alone mode

Direct mode and stand alone mode can also be mixed. When a TMCL™ program is being executed in standalone mode, direct mode commands are also processed (and they do not disturb the flow of the program running in standalone mode). So, it is also possible to query e.g. the actual position of the motor in direct mode while a TMCL™ program is running.

Communication between a program running in standalone mode and a host can be done using the TMCL™ user variables. The host can then change the value of a user variable (using a direct mode SGP command) which is regularly polled by the TMCL™ program (e.g. in its main loop) and so the TMCL™ program can react on such changes. Vice versa, a TMCL™ program can change a user variable that is polled by the host (using a direct mode GGP command).

A TMCL™ program can be started by the host using the run command in direct mode. This way, also a set of TMCL™ routines can be defined that are called by a host. In this case it is recommended to place JA commands at the beginning of the TMCL™ program that jump to the specific routines. This assures that the entry addresses of the routines will not change even when the TMCL™ routines are changed (so when changing the TMCL™ routines the host program does not have to be changed).

Example:

```
//Jump commands to the TMCL™ routines
Func1:      JA Func1Start
Func2:      JA Func2Start
Func3:      JA Func3Start

Func1Start: MVP ABS, 0, 1000
            WAIT POS, 0, 0
            MVP ABS, 0, 0
            WAIT POS, 0, 0
            STOP

Func2Start: ROL 0, 500
            WAIT TICKS, 0, 100
            MST 0
            STOP

Func3Start:
            ROR 0, 1000
            WAIT TICKS, 0, 700
            MST 0
            STOP
```

This example provides three very simple TMCL™ routines. They can be called from a host by issuing a run command with address 0 to call the first function, or a run command with address 1 to call the second function, or a run command with address 2 to call the third function.

You can see the addresses of the TMCL™ labels (that are needed for the run commands) by using the *Generate symbol file* (chapter 6.1.1) function of the TMCL-IDE.

9 Revision history

9.1 Document revision

Version	Date	Author	Description
1.00		OK	Initial Release as a chapter of the TMCL™ reference and programming manual
2.00	2009-JUL-30	SD	New version as self-contained TMCL™ User Manual
2.01	2011-SEP-14	SD	Chapter 6.6 (stallGuard2™ and coolStep™) added, minor changes

Table 9.1: Document revision