TG Motion

version 4

Control Observer II

operation manual

TG Drives, s.r.o.
Olomoucká 1290/79
CZ - 627 00 Brno

Tel.: +420 548 141 811
Fax: +420 548 141 890
E-mail: info@tgdrives.cz

Revision History

date	version	revision
31 July 2017	1.0	Initial release

TG Drives, s.r.o.
Olomoucká 1290/79
CZ - 627 00 Brno

Tel.: +420 548 141 811
Fax: +420 548 141 890
E-mail: info@tgdrives.cz

Contents

TG Drives, s.r.o.
Olomoucká 1290/79
CZ - 627 00 Brno
Tel.: +420 548 141 811
Fax: +420 548 141 890
E-mail: info@tgdrives.cz

Control Observer II

Control Observer description

Control Observer is a set of utilities, which was developed to diagnose the **TG Motion** system and to debug the PLC and user Windows applications. **Control Observer** comprises tools for servo drive direct testing and control, PLC code retrieval and System Timer parameter display. Another group of utilities is intended to display, track and change selected shared memory registers. Yet another section enables to save and download the **Control Observer** projects (displaying and parameters of each utility, window deployment and settings, etc.) and parts of the shared memory.

Control Observer is an independently executable program **Control_Observer_II.exe**, which is being supplied with three libraries:
TGM_Comm_Int_2.dll – is used to communicate with TG Motion on the same computer
TGM_Mini.dll – enables connecting to a TG Motion running on a TGMmini
TGM_Remote.dll – connects with a TG Motion on another computer via a network.

Control Observer operating modes

After **Control Observer** is started, an icon appears in the message area of the Windows task bar, whose color indicates the Control Observer operating mode and/or the **TG Motion** status. The same icon may be found in the headings of each utility window.

online – green icon
Control Observer is connected with **TG Motion**, the shared memories are defined by the running **TG Motion**, **Control Observer** can communicate with **TG Motion** as well as with Windows applications. This mode is used to debug the entire system of **TG Motion**, PLC, user Windows applications, and to test the servo drives and I/O modules.

offline – red icon
Control Observer is not connected to **TG Motion**, the shared memories were created by the **Control Observer.** The latter can only communicate with Windows applications. This mode is particularly useful when debugging the Windows applications.

version conflict – yellow icon
Control Observer has detected the **TG Motion**, the shared memories have already been defined by the running **TG Motion**. **Control Observer** can communicate with Windows applications but, owing to incompatibility, cannot communicate with **TG Motion**. Either **Control Observer** or **TG Motion** has to be updated to the same version.

Fig. Location of the Control Observer icon in the window caption and in the notification area

Control Observer Utilities

General Utilities

Servo Tester

This utility is intended for testing and controlling of individual servo drives.

PLC Loader

It is intended to download and start PLC.

System Timer

It displays the current CPU load by the different **TG Motion** processes.

Tools

Oscilloscope

It provides the graphical representation of the time dependence of selected register values.

Graphic Viewer

It is used to provide the graphical representation of a continuous series of selected registers.

Select Registers

It is an auxiliary utility, which is designed to select the registers to monitor, modify or use in other utilities.

Watch List

It is used to found a new WatchList or to select an already existing one.

Auxiliary Utilities

Open/Save

It makes it possible to open or save a project or save the shared memories.

Connection Info

It provides a clear information on the system connection and functionality.

Exit

It closes the **Control Observer** including all of its utilities. **TG Motion** keeps running.

TG Drives, s.r.o.
Olomoucká 1290/79
CZ - 627 00 Brno

Tel.: +420 548 141 811
Fax: +420 548 141 890
E-mail: info@tgdrives.cz

Servo Tester

Servo Tester Utility description

The **Servo Tester** utility is used to diagnose the connection and proper functioning of the different servo drives and their interactions with each other. **Control Observer** allows to launch one instance of the **Servo Tester**.

It makes it possible to diagnose the different servo drives, to control them directly, to simulate the gear box and to set the reference position. **Servo.Command** interface structure is used to position and control the servo drives**.** The user has an option to designate the different servo drives for the purpose of improving the lucidity.

Servo Tester offers two push buttons, namely, **Reset All** and **Stop All** in all tabs. They are always located at the bottom of the window and are immediately accessible at any time.

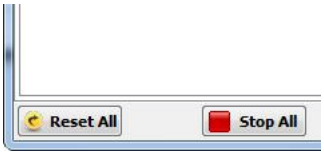


Fig. Reset All and Stop All push buttons

Reset All – resets the failure of all servo drives. The servo movement, if any, will not be cancelled.
Stop All – stops all servo drives according to the current deceleration ramp, sets all servo drives to Mode=0 and leaves all servo drives under torque.

Config

Config is the first tab to show the different servo drives in the form of a table. At the most, 256 servo drives can be connected (0–255).

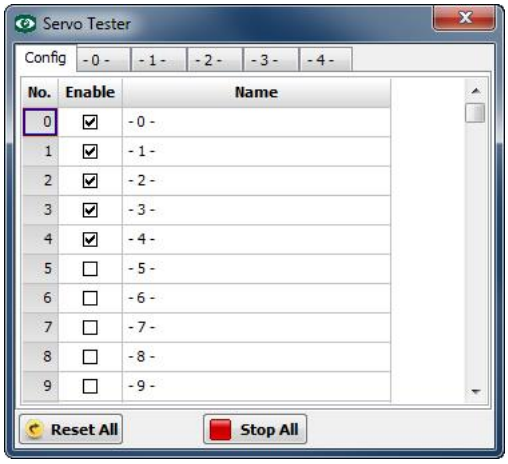


Fig. Servo Tester – Config tab

No. – defines the index number of a virtual servo drives. Physical servo drives are connected to the virtual logical servo drives according to **Tgmotion4xx.ini** file.

Enable – allows the user to display a tab with control parameters of the respective servo drives. The servo drive number is defaulted to 4.

Name – contains the servo drive number (see No. column). However, the user can change the names of the different virtual servo drives arbitrarily in the context of the **Servo Tester.** This improves the clarity and orientation, particularly in the case of a high number of servo drives.

*The servo drive name is only valid in the context of the **Servo Tester**.*

Servo drive tabs

When a particular servo tab is chosen a dialog box is open with information on the basic values of selected registers and tabs to control the servos and set the values of some registers.

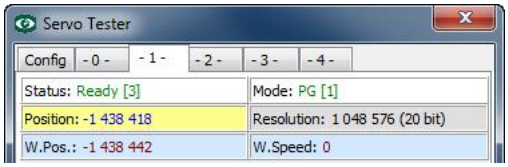


Fig. Servo drive status and important registers

Values of following registers of the selected servo drives are shown in respective boxes:
Status – displays the values of the register Servo_xxx.Status (Ready, >> OFFLINE <<, etc.).
Servo Mode – shows the value of the register Servo_xxx.Mode (PG[1], Gear + PG[4], >> OFFLINE <<, etc.).
Position – current absolute position as obtained by reading from the servo [inc].
Resolution – resolution of one turn of the servo [inc].
W.Pos – required absolute position [inc].
W.Speed – required speed, calculated from W.Pos [inc/s].

These parameters are always displayed, regardless of the selected subtab of controlling the particular servo.

Servo drive subtabs

Five subtabs of each tab allow to control directly the servo speed as well as position in various modes or to set the values of some registers.

Jog subtab

Is used to control the servo speed in the Jog style using a non-arrest push button switch – the servo drive moves in the selected direction as long as one of the push buttons (*Positive*, *Negative*) is pressed. When the push button is released, the servo drive will stop.

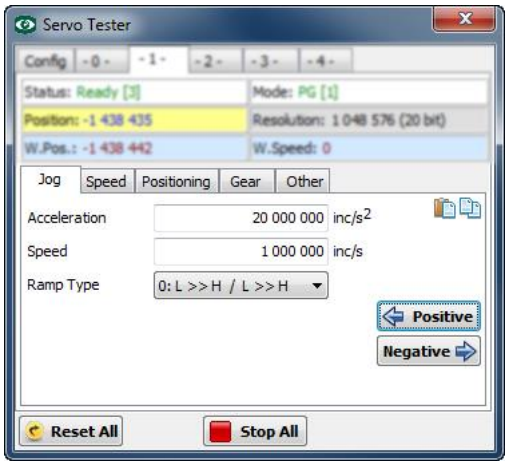


Fig. Jog subtab

Description of controls

Acceleration – The required value of the acceleration is to be typed in the box in [inc/s²].
Speed – the required speed is to be typed in the box in [inc/s].
Ramp Type – selection of one from four run-up, run-down waveform types. L (linear), H (harmonic).

Fig. Ramp types

Positive, Negative – forward, backward motion push buttons.

Description of Operation

When the ***Positive*** push button is pressed and held pressed, the servo starts moving in positive direction (Position value is growing up) with an acceleration given by the value of the ***Acceleration*** parameter. The shape of the run-up curve depends on the form of the run-up ***ramp***. As soon as the servo reaches the speed which is determined by the ***Speed*** parameter value, it keeps moving at this constant speed.

After the ***Positive*** push button is released, the servo starts slowing down according to the ***Acceleration*** parameter value. The shape of the slow-down curve depends on the form of the run-down ***ramp***. When the servo stops, it stays under the torque, waiting for next instructions.

If the push button is released during the speed-up phase (i.e. before the ***Speed*** was reached), the servo control will pass smoothly into the slow-down phase according to ***Acceleration*** and ***Ramp Type*** parameters.

Speed subtab

It is used to control the servo speed – the servo drive starts moving in the selected direction after one of the push buttons (***Positive***, ***Negative***) is clicked on. To stop the servo motion, click on ***Stop*** push button. It follows that none of the push buttons is to be held pressed to keep the servo moving so that the user may go in for other activities (observing the register values in WatchList, utility Oscilloscope, etc.).

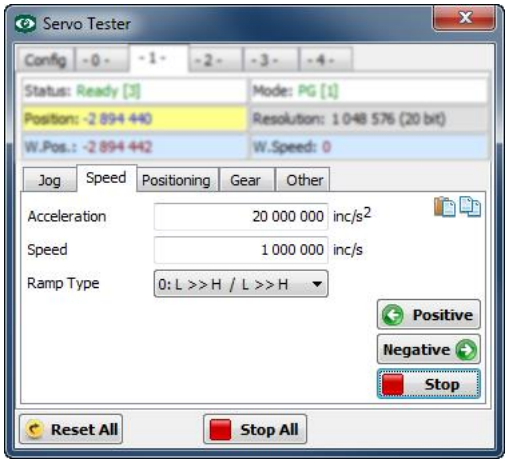


Fig. Speed subtab

Description of controls

Acceleration – the required value of the acceleration is to be typed in the box in [inc/s²].
Speed – the required speed is to be typed in the box in [inc/s].
Ramp Type – selection of one from four run-up, run-down waveform types. L (linear), H (harmonic).

Fig. Ramp types

Positive, Negative – forward, backward motion push buttons.
Stop – this push button stops the movement.

Description of Operation

When the ***Positive*** push button is clicked on, the servo starts moving in positive direction (Position value is growing up) with an acceleration given by the value of the ***Acceleration*** parameter. The shape of the run-up curve depends on the form of the run-up ***ramp***. As soon as the servo reaches the speed, which is determined by the ***Speed*** parameter value, it will keep moving at this constant speed.

After the ***Stop*** push button is clicked on, the servo starts slowing down according to the ***Acceleration*** parameter value. The shape of the slow-down curve depends on the form of the run-down ***ramp***. After stopping, the servo will stay under torque.

If the user changes any parameter during any phase, the system will pass smoothly from the current status to the newly required status.

*If the **Stop** push button is pressed during the speed-up phase (i.e., before the **Speed** was reached), the servo control will pass smoothly from the current speed into the slow-down phase according to **Acceleration** and **Ramp Type** parameters.*

*If an opposite direction push button (in this particular case **Negative**) is pressed during the speed-up phase or after the required speed is reached, the system will pass smoothly through the slow-down phase to zero speed according to the run-down **Ramp** and then to the acceleration phase according to the respective parameters in opposite direction (in this case to lower **Position** values).*

*If, during the movement or during the acceleration phase, new values of **Acceleration**, **Speed**, or **Ramp Type** parameters are entered, the system will accept the new values. After the **Positive** or **Negative** push button is pressed, the system will pass smoothly to speed control of the motion according to the new parameters.*

Positioning subtab

It serves to the position control of the servo, i.e., setting the servo at the required position. The positioning may be either relative or absolute.

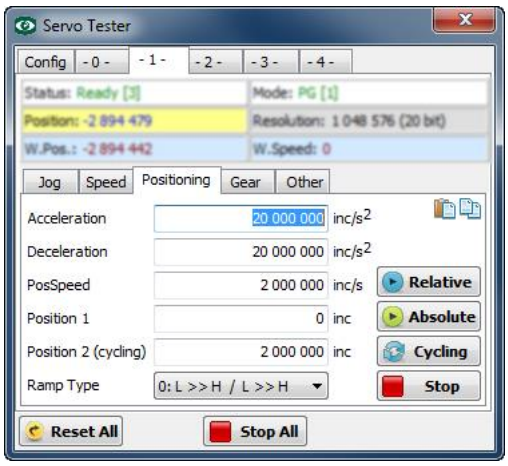


Fig. Positioning subtab

Description of controls

Acceleration – the acceleration in the run-up phase [inc/s²].
Deceleration – the deceleration when reaching the new position [inc/s²].
PosSpeed – the speed to be reached after the speed-up phase [inc/s].
Position 1 – the target position in increments, in the case of a cyclical movement, one of the positions between which the cyclical movement takes places [inc].
Position 2 – the second position in increments, it is only used to the cyclical movement [inc].
Ramp Type – selection of one from four speed ramp types. L (linear), H (harmonic).

Fig. Ramp types

Relative – enables relative positioning.
Absolute – enables absolute positioning.
Cycling – enables cyclical movement between two absolute positions.
Stop – this push button stops the movement.

Relative

When ***Relative*** push button is pressed, the servo starts moving in the required direction by the number of increments resulting from the ***Position 1*** parameter value. The movement will begin with an acceleration resulting from the ***Acceleration*** parameter, the speed following the run-up ***Ramp.*** Arrival to the target position will take place with a deceleration resulting from the ***Deceleration*** parameter and the run-down ***Ramp*** shape. After stopping, the servo will stay under torque.

Absolute

When ***Relative*** push button is pressed, the servo starts moving in the required direction to the position resulting from the ***Position 1*** parameter value. The movement will begin with an acceleration resulting from the ***Acceleration*** parameter, the speed following the run-up ***Ramp.*** Arrival to the target position will take place with a deceleration resulting from the ***Deceleration*** parameter and the run-down ***Ramp*** shape. After stopping, the servo will stay under torque.

Cycling

When ***Cycling*** push button is pressed, the servo starts moving in the required direction to the position resulting from the ***Position 1*** parameter value. The movement will begin with an acceleration resulting from the ***Acceleration*** parameter, the speed following the run-up ***Ramp.*** Arrival to ***Position 1*** will take place with a deceleration resulting from the ***Deceleration*** parameter and the run-down ***Ramp*** shape. After stopping, accelerated movement to ***Position 2*** according to the respective parameters will follow. When this position is reached, the entire movement repeats cyclically.

If the input parameters change during the cyclical movement, the system will complete the current movement phase. At the stop point it will update the parameters and start the cyclical movement with new parameters.

If the mode changes during the movement (e.g., ***Absolute*** push button is pressed during the cyclical movement), the system will pass smoothly to the new control mode.

Stop

When this push button is pressed, the movement of the servo drive will stop and the servo drive will be set under torque.

TG Drives, s.r.o.
Olomoucká 1290/79
CZ - 627 00 Brno
Tel.: +420 548 141 811
Fax: +420 548 141 890
E-mail: info@tgdrives.cz

Gear subtab

It is intended to simulate a controlled motion of two or more servo drives. A linear transmission gear with a fixed transmission ratio is simulated here.

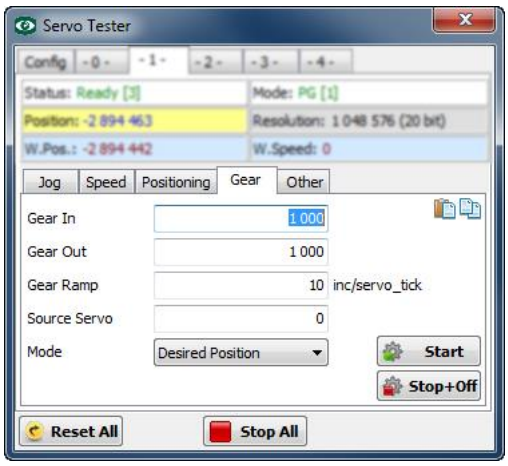


Fig. Gear subtab

Description of controls

Gear In – denominator in the transmission ratio (number of teeth of the driving wheel).
Gear Out – numerator in the transmission ratio (number of teeth of the driven wheel).
Gear Ramp – (*Gear.IncIn)* it is used to enable the transmission gear smoothly.
Source Servo – servo which acts as a master (driving gear).
Mode – method of connecting the servo to Source Servo.
Desired Position – required position – *WritePosition.*
Actual Position – current position – *Position.*
Time – time base of TG Motion in servo ticks.
External Position – current position from the position sensor of the Source Servo.
Start – activation of Gear.
Stop+Off – the servo is stopped and disconnected from the Source Servo (deactivation of Gear).

Description of functions

Gear In and ***Gear Out*** are used to set up the transmission ratio. Example: ***Gear In*** = 1000 and ***Gear Out*** = 500 will set the transmission ratio 1:2. Therefore, one turn of the ***Source Servo*** will cause two turns of the controlled servo.

Gear In *accepts negative values. In this case, the direction of rotation of the driven servo changes.*

Gear Ramp – as the activation of the transmission ratio may start even during the Source Servo operation, a smooth onset of the transmission must be guaranteed (the clutch principle). ***Gear Ramp*** determines how quickly the required transmission gear parameters are reached. ***Gear Ramp*** value is added to / is subtracted from each tick of the Source Servo until the required transmission ratio is reached.

Source Servo – determines the index number of the servo drive, which acts as a master (Source) for the current servo.

Mode – defines the method of connecting to the Source Servo.

Desired Position – the required position, *WritePosition,* makes the source value for the synchronization.

Actual Position – the current position *Position* makes the source for synchronization. In fact, it is the value obtained by reading from the *Source Servo encoder*.

Time – time base of TG Motion in ticks – Cycle_Time. The setting of the Source Servo parameter does not influence the result of the synchronization.

External Position – the current position value as read out from the external position sensor of the relevant servo makes the source.

Start – enables the Gear transmission with updated parameters.

Stop+Off – stops the current (driven) servo and disconnects it from the ***Source Servo***, disables the current Gear transmission ratio. The disconnection does not affect the operation of the Source Servo in any way.

Other subtab

Is used to set the reference position employing the value of the Offset register.

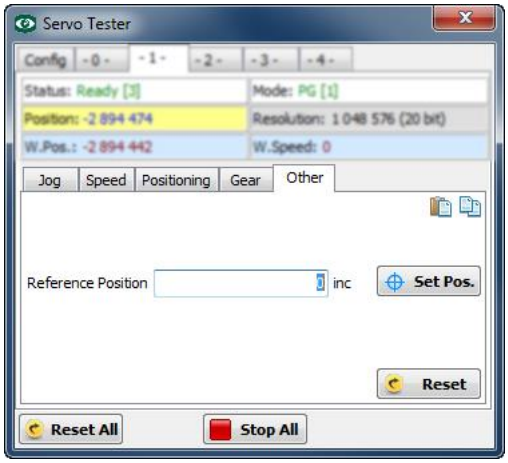


Fig. Other subtab

Reference Position – this box is intended for entering the reference position value.
Set Pos. – assigns the value entered into the *Reference Position* box to the current absolute position.
Reset – resets failures of the actual servo and sets them under torque.

Given an absolute position of 10 000 000, if a Reference Position value = 50 000 is entered and the Set Pos. push button is pressed, following values will be in the corresponding registers:
APos = 10 000 000
Position = 10 000 000
Offset = 9 950 000
RefPosition = 50 000

Value copying and entering

In the right upper corner of each subtab of the **Servo** tab there are two icons, namely, for parameter value copying and entering.

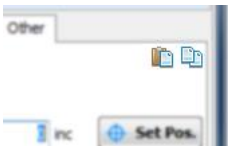


Fig. Copying and entering icons

Only data of equal tab types can be copied between different servo drives. For example, the values of the **Gear** tab parameters can be copied into the **Gea**r parameters of another servo. Example: It follows from the logic of the matter that data of the **Positioning** tab cannot be copied into the **Jog** tab.

When copying values of the parameters, the Windows clipboard can be used (Ctrl+C, Ctrl+V).

PLC Loader

PLC Loader is a simple utility featuring an intuitive interface. It is used to general handling of the PLC code (readout, pause, etc.). **Control Observer** allows to run one instance of **PLC Loader**.

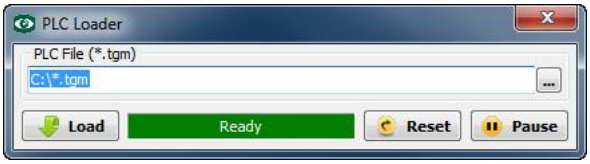


Fig. PLC Loader

PLC

PLC is a file comprising a user code for controlling servo drives and I/O units. PLC programming is carried out in programming languages which allow creation of native dll (C++, Delphi, etc.). It contains instructions, which are used in the operation of **TG Motion**. The extension of PLC files is *.tgm.

Selection of a PLC file

To select a file, just to type the path into the text box. Click on the icon ... next to the text box to open a classical Windows dialog box for selecting the directory and the file, by means of which the PLC file and its location can be chosen.

Load

Press the **Load** button to read and automatically run the selected PLC. To avoid undefined states (particularly in servo drives), it is advisable to use Load of a new PLC, when the servo drives are not in motion and do not carry out important operations.

Reset

It stops the PLC and restarts it from the beginning.

*It is advisable to carry out **Reset** when the servo drives are in standstill.*

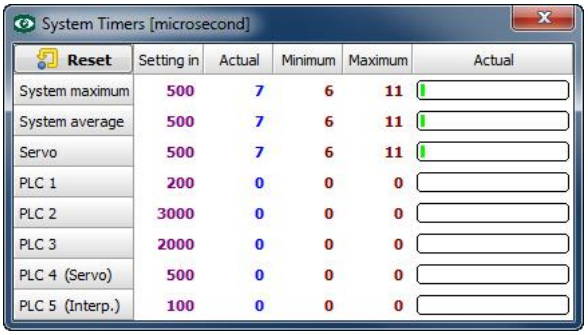
Pause

It stops temporarily the running PLC. Pressing it again makes the PLC continue operating. When **Pause** is active, the push button is framed in red.

*The PLC Loader box displays the **TG Motion** status at the bottom: Online (green), Offline (red) and Version Conflict (yellow).*

System Timer

System Timer is a utility, which is used to diagnose and debug PLC from the viewpoint of using the time-based facilities of the system. **Control Observer** allows to launch one instance of the **System Timer**.



Reset	Setting in	Actual	Minimum	Maximum	Actual
System maximum	500	7	6	11	
System average	500	7	6	11	
Servo	500	7	6	11	
PLC 1	200	0	0	0	
PLC 2	3000	0	0	0	
PLC 3	2000	0	0	0	
PLC 4 (Servo)	500	0	0	0	
PLC 5 (Interp.)	100	0	0	0	

Fig. System Timer

The table shows the maximum and the average values for the CPU load of system, servo drives and for Program_01 through Program_05 of the currently running PLC. The item values (set value, current value, minimum and maximum values attained) are displayed in microseconds. To the right, there is a progress bar showing the current value of each item.

Oscilloscope

Oscilloscope utility description

This utility is intended for debugging PLC or other service programs running under Windows. **Control Observer** allows to launch one instance of this utility.

Oscilloscope is a sophisticated tool for graphical representation of values and their time dependence. It detects and saves (in the **TGM_Oscilloscope** shared memory) data – values of selected registers within a certain time interval to display them in the form of a chart. Time is plotted on the horizontal axis and register values ate plotted on the vertical axis. Subsequently, the data can be analyzed and used in calculations and measurements can be carried out.

The values displayed and the settings of **Oscilloscope** can be saved in file ***.Osc_CO2** and retrieved again.

The user interface consists of the main menu, a graph and a service function section at the bottom of the box.

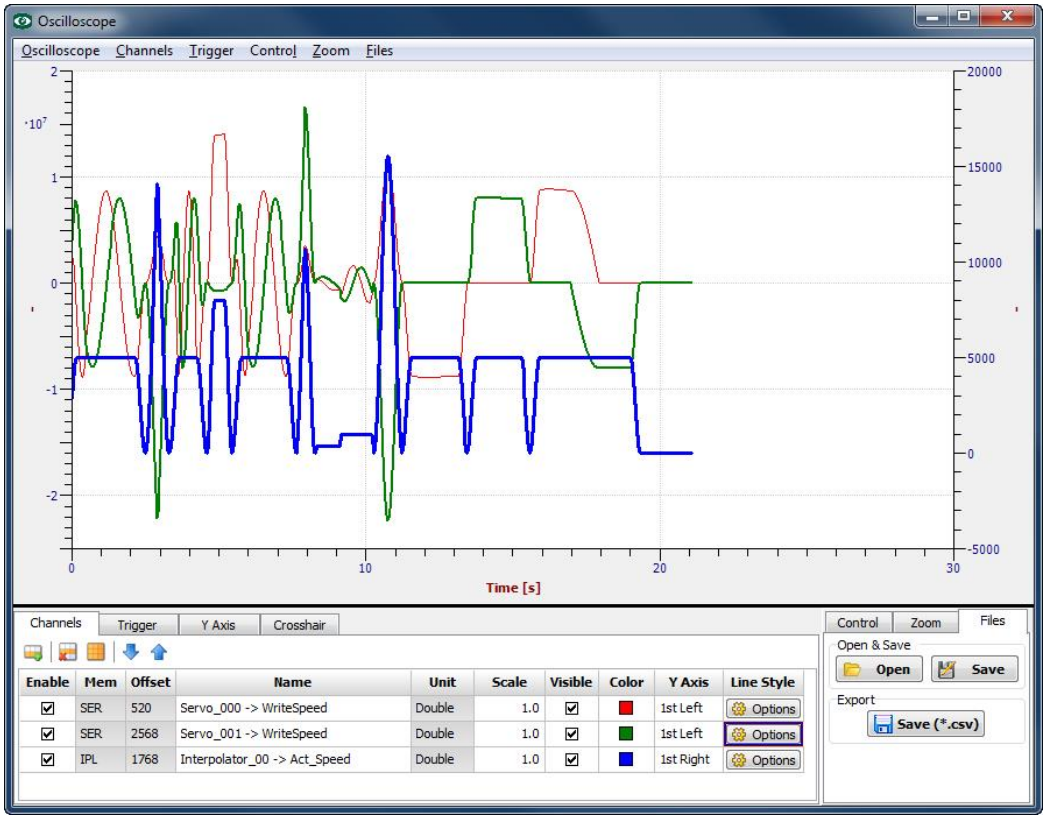


Fig. User interface of Oscilloscope utility

The size proportion between the whole chart and its bottom part can be changed by dragging the black dividing line between the two areas.

Oscilloscope menu

The menu of **Oscilloscope** utility contains functions and settings, which are available in the form of charts and tabs at the bottom of **Oscilloscope** box. For their detailed description, see the next paragraphs.

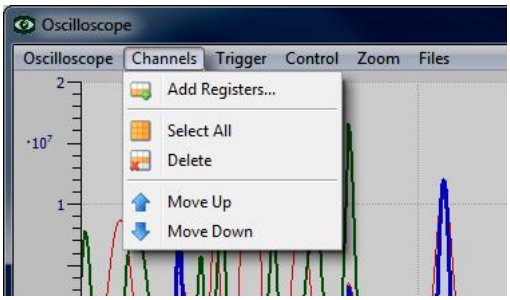


Fig. Oscilloscope utility menu

Oscilloscope – it contains a command to close Oscilloscope. As the data remain in the shared memory after the Oscilloscope utility is closed, they are permanently available after it is open again.
Channels – it contains function for operations and handling the items of the Channels table.
Trigger – it contains function for operations and handling the items of the Trigger table.
Control – it allows to start and stop the register value recording.
Zoom – it contains commands for navigation in the Zoom history and for resetting the history.
Files – it allows to save the displayed data and settings of the Oscilloscope into ***.Osc_CO2** file and retrieve them again. To share data with other applications, export to *.csv file can be used.

Graphic area

The graphical area displays the values of selected registers versus time. Time, which is common for all registers, is plotted on the horizontal axis. Values of the represented registers are plotted on the vertical axes. Four vertical axes can be shown simultaneously, for four different registers (see Y Axis paragraph).

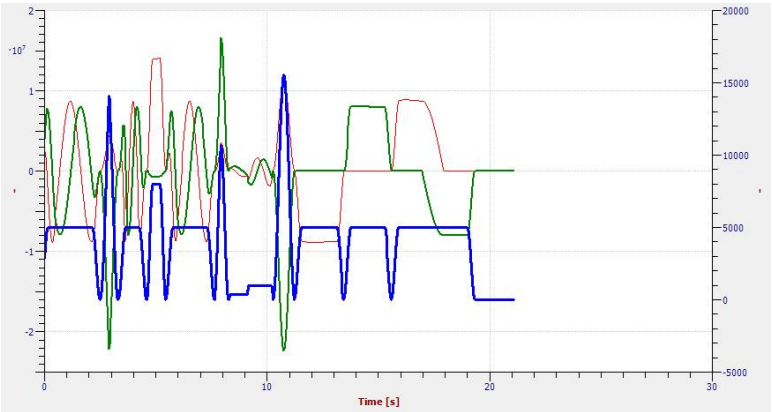


Fig. Graphical area of Oscilloscope utility

The charts may contain crosshairs (cursors), by means of which values can be read out, measurements and calculations can be carried out.

Zoom – chart cropping

To enlarge a certain part of the chart, i.e., to show its cropped part, select a rectangle in the graphics area by the mouse using the Drag&Drop method.

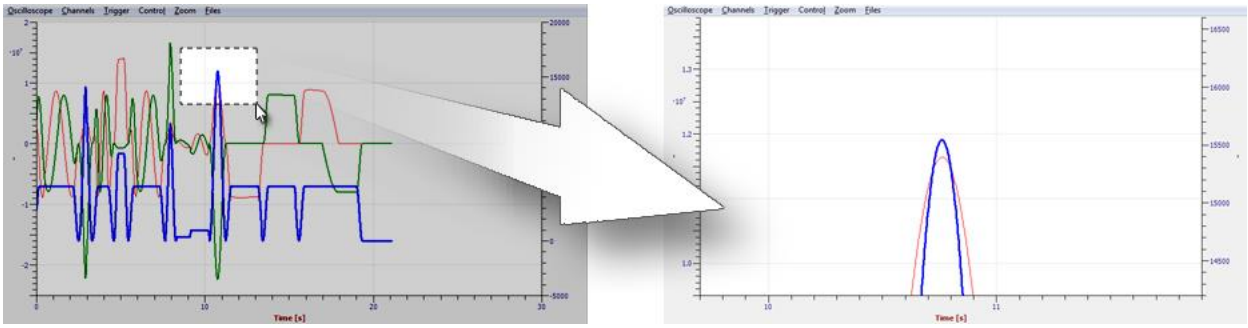


Fig. Enlarging a part of the chart using the mouse

Parameter area

At the bottom left part of the **Oscilloscope** box there are four tabs to be used for displayed registers, triggering registers, y axis setting and for cursors.

Channels – it makes it possible to select the registers and set their representation.
Trigger – it offers a choice of triggering registers and the setting of recording conditional start.
Y Axis – it is used to set up the y axis of the different plots.
Crosshair – cursor setting.

Channels

This tab is used to select the registers whose values are to be displayed and to set the format of their representation.

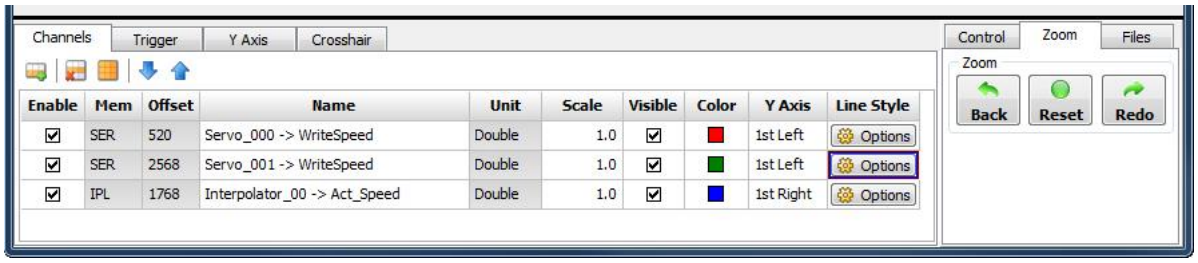


Fig. Oscilloscope utility – Channels tab

At the upper left part of the tab there are five icons intended for selecting registers and altering their sequence in the table.

Add Register

After **Add Register** tab is selected, **Select Registers** dialog box is open to select and add registers, whose values are to be displayed in **Oscilloscope.** For a detailed description of **Select Registers** utility, see Select Registers chapter.

The registers which were added are shown in a table, in which some of their parameters are displayed and allows to set the parameters of their graphic representation.

*The **Oscilloscope/Channels** tab can contain up to 64 registers. The user is informed when the maximum number is exceeded. The table is completed to contain the maximum register number.*

The maximum number of simultaneously represented registers is 32 (see Enable column below).

The Channels tab may contain several identical registers.

Delete

It removes the selected register or registers from the table.

To select a register in the table just click on any cell, which pertains to it. To select multiple registers or table cells use the Drag&Drop method over the cell area pertaining to the required registers.

Channels | Trigger | Y Axis | Crosshair

Enable	Mem	Offset	Name	Unit	Scale	Visible	Color	Y Axis	Line Style
☑	SER	0	Servo[0].Number	I32-Dec	1.0	☑		1st Left	Options
☑	SER	4	Servo[0].Node	I32-Dec	1.0	☑		1st Left	Options
☑	SER	8	Servo[0].Axe	I32-Dec	1.0	☑		1st Left	Options
☑	SER	12	Servo[0].Type	I32-Dec	1.0	☑		1st Left	Options
☑	SER	16	Servo[0].Resolution	uI32-Dec	1.0	☑		1st Left	Options
☑	SER	20	Servo[0].SerialNumber	I32-Dec	1.0	☑		1st Left	Options
☑	**SER**	**24**	**Servo[0].Control**	**I32-Dec**	**1.0**	☑		**1st Left**	Options
☑	**SER**	**28**	**Servo[0].Status**	**I32-Dec**	**1.0**	☑		**1st Left**	Options
☑	SER	32	Servo[0].Mode	I32-Dec	1.0	☑		1st Left	Options

Fig. Selection of registers in the table

Select All

It selects all items of the table.

Move Down, Move Up

It moves the selected row of the table (registers) by one position upwards or downwards.

Represented register table – items

Enable – allows to read register data from a shared memory. Values of a maximum of 32 registers can be displayed at a time.
Mem – type of the shared memory, in which the register is situated.
Offset – pointer offset to the memory where the register is situated.
Name – name of the register. The user can change this name. Any new name is only valid for one register within the table of the Channels tab.
Unit – type of the register variable.
Scale – coefficient of representation in y axis. It is defaulted to 1.0.

*If the **Autorange** parameter of the **Y Axis** tab is active, the change of the **Scale** coefficient has no effect.*

Visible – the register value chart is displayed.
Color – selection of the register chart color.
Y Axis – location of y axis for the register in question.

*If the axes of multiple registers are located at the same place and the **Autohide** parameter of **Y Axis** tab is active, only the y axis of the uppermost-located register in the table will be shown.*

Line Style – it opens the dialog box to set up the color and style of the chart as well as the representation of the chart points of the register in question.

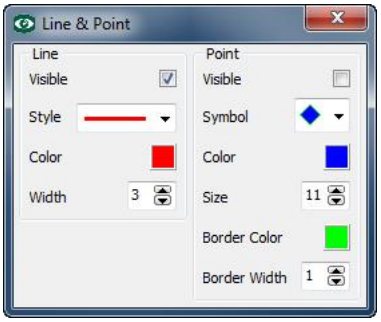


Fig. Setting the graphic parameters of the representation

Line
Visible – enables the representation of a continuous chart.
Style – offers the style of displaying a line in a chart (continuous, dashed, dash-dotted, etc.).
Color – sets up the line color it is the same parameter as Color in Channels table).
Width – sets up the width of the line.

Point
Visible – enables displaying the measurement readings in the form of points.
Symbol – offers a choice of shapes of the displayed points.
Color – sets up the color of the points.
Color – sets up the size of the points.
Border Color – sets up the color of the border lines.
Border Width – sets up the width of the border lines.

Both representation types can be combined arbitrarily. At least one representation must be active.

Adjust the column width by dragging the division line between the headings.

Fig. Setting the width of table columns

Trigger

The tab is intended for selecting the register whose values are used to conditional automatic start of **Oscilloscope** recording. When a certain value of the register is reached or passed through, the recording is started automatically.

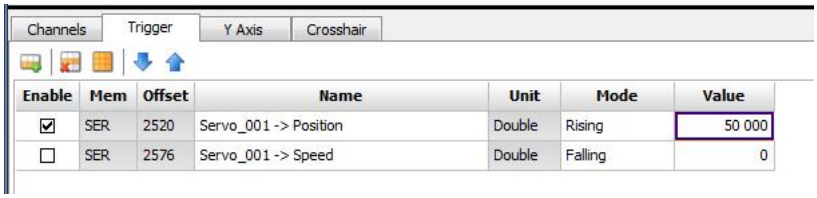


Fig. Triggering register tab

Five push buttons in upper left corner, i.e., ***Add Register…***, ***Delete***, ***Select All***, ***Move Down*** a ***Move Up*** have the same function as in Channels tab.

Add Register

After **Add Register** icon is clicked, **Select Registers** dialog box is opened to select and add registers, whose values determine the record triggering. For a detailed description of the select register utility, see **Select Register** chapter.

The registers which were added are shown in a table, in which some of their parameters are displayed, and allows to set up the parameters of their graphic representation.

*The **Oscilloscope/Trigger** tab can contain up to 4 registers. The user will be informed, if he exceeds the maximum permissible number. The table is completed to contain the maximum register number.*

Only one triggering register may be active (Enable) – see the Enable column below.

The Trigger tab may contain several identical registers.

Delete

It removes the selected register or registers from the table.

To select a register in the table just click on any cell, which pertains to it. To select multiple registers or table cells use the Drag&Drop method over the cell area pertaining to the required registers.

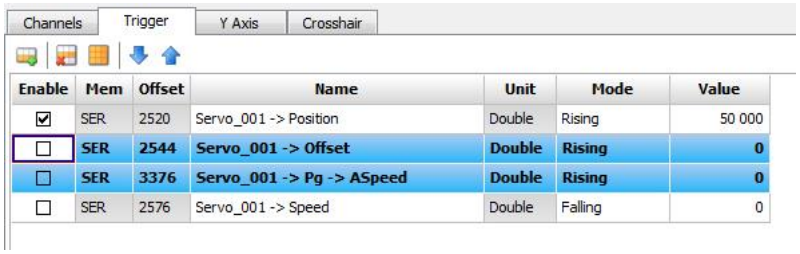


Fig. Selection of registers in the table

Select All

It selects all items of the table.

Move Down, Move Up

It moves the selected rows of the table (registers) by one position upwards or downwards.

Triggering register table – items

Enable – selects the register whose value is tracked by the record start algorithm.
Mem – type of the shared memory, in which the register is situated.
Offset – byte offset to the memory place where the register is situated.
Name – name of the register. The name can be changed. Any new name is only valid for one register within the table of the Channels tab.
Unit – type of the register variable.
Mode – selecting the method to detect the register value.
Rising – incremental (register value growth is supposed).
Falling – decremental (register value decrease is supposed).
Value – value at which the Oscilloscope recording starts.

*The triggering register values need not be exactly required values. **Oscilloscope** is testing the triggering register value by means of Mode parameter. As soon as it is reached or passed through the **Oscilloscope** record is launched.*

Example: Value=20, Mode=Rising
The values of the register in question are growing up with an increment of 3, i.e., 12, 15, 18, 21, 24, 27, 30… After values of 18 and subsequently 21 are detected, the triggering algorithm decides that the required value of 20 has already been reached and launches the Oscilloscope record.

Adjust the column width by dragging the division line between the headings.

Fig. Changing the column width in a table

Setting up the triggering

Set up the registers in **Channels** tab, set up and enable the triggering register and press **Record** push button. Waiting to Trigger status will appear in a yellow box. **Oscilloscope** is ready to record and waits for the trigger condition to be met. As soon as it is met, values of the selected registers will start to be recorded automatically. The user is informed of the recording progress at the bottom of the tab by Recording tag in a yellow box together with the progress bar informing of the **TGM_Oscilloscope** memory status. Press **Stop** button to end recording.

After the recording is completed, the recorded values will be shown. The time axis will be set to match the recording duration and the y axis, according to the values recorded. Both cursors are disabled at the same time.

*If the number of recorded values reaches the size of the **TGM_Oscilloscope** memory, the recording will be stopped.*

Y Axis

This tab is used to set up and display the y axes in the **Oscilloscope** graphic area.

Channels | Trigger | Y Axis | Crosshair

Y Axis	Caption	Autorange	Min	Max	Type	Reverse	Autohide
1st Left	-	☑	0.0	1.0	Linear	☐	☑
1st Right	-	☑	0.0	1.0	Linear	☐	☑
2nd Left	-	☑	0.0	1.0	Linear	☐	☑
2nd Right	-	☑	0.0	1.0	Linear	☐	☑

Fig. Y Axis tab

Four y axes are available to depict the range of the first four registers in Channels tab. Two y axes can be shown at the left hand side of the chart (1st Left, 2nd Left). Similarly, the remaining two y axes can be shown at the right hand side of the chart (1st Right, 2nd Right).

Y axis parameters

Y Axis – axis location at the chart side.
Caption – axis name (description). It will also appear at the corresponding axis in the graphic area.

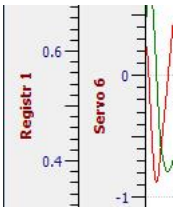


Fig. Y axis representation in the chart

Autorange – on enabling, the range of the y axis is set up automatically to be consistent with the register values being read. If the **Autorange** parameter is disabled, the y axis range is determined by two following parameters.
Min – y axis range minimum value.
Max – y axis range maximum value.
Type – it determines the y axis plot type – linear (Linear), logarithmic (Log) and exponential (10EE).
Reverse – the representation of the register values in y axis is inverted.
Autohide – when enabled, the priority representation pertains to the axis of a register located higher in the table. When disabled, both axes are displayed at the corresponding side of the chart. This parameter does not influence the graphic representation of the register values.

Crosshairs

This tab comprises the setting for two cursors, which are used to read values and carry out measurements. These cursors are shown in the form of a cross in the graphics area of the **Oscilloscope**.

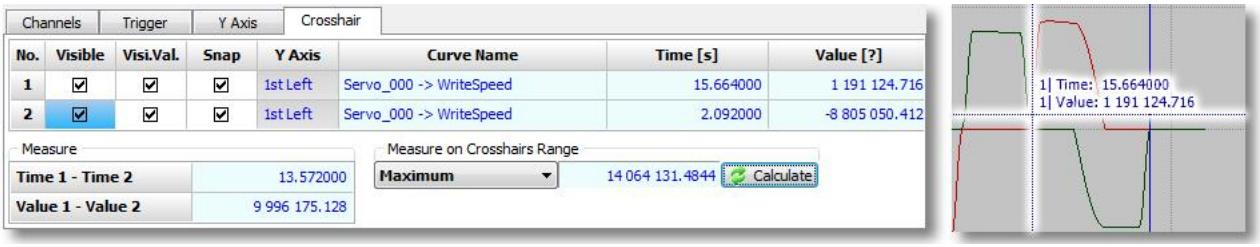


Fig. Cursor table and cursor representation in the chart

The cursor setting table comprises following parameters

Visible – enables and disables the cursor (the cursor is disables automatically after the record is completed).
Visi.Val. – allows to display *Time* and *Value* at the current position of the graphic part.
Snap – makes it possible to snap the cursor to the chart.
Y Axis – sets the y axis which value the cursor is reading. The y axis function is not available if the Snap parameter is enabled.

The user will be alerted if he chooses a y axis which is not displayed (Y Axis tab, Autohide parameter). This fact does not affect the value reading.

Curve Name – **Snap** parameter being enabled, **Curve Name** shows the name of the register, to which the cursor is snapped and whose value it is reading.
Time [s] – time of the cursor current position.
Value [?] – value of the cursor current position (it depends on y axis or on snapping by means of Snap function).

If the Snap function is enabled, the cursor snaps to the nearest curve of the chart.

Measure

Time 1 – Time 2 – the time interval during which the measurement is carried out is displayed.
Value 1 – Value 2 – the difference between the register values at the cursor positions is displayed.

Measure is only enabled if both cursors are active (parameter Visible).

Measure on Crosshairs Range

It is used to calculate the values within the time interval defined by the two cursors.
drop-down bar – it is used to select the calculation of maximum, minimum, average or root-mean-square value.
value box – it displays the calculation result.
Calculate – this push button is used to carry out the calculation.

The Calculate push button (execution of calculation) is only enabled of both cursors are snapped to the same curve, hence if both cursors are displaying values of the same register.

Auxiliary functions

At the bottom right corner of **Oscilloscope** window there are three tabs intended to record, zoom control and data saving and retrieving.

Control – offers record control and supporting function.
Zoom – allows navigation in zoom history and its resetting.
Files – is intended to save data and Oscilloscope setting into *.Osc_CO2 file, their retrieval and export to *.csv.

Control

It starts recording the register values in the shared memory of **TGM_Oscilloscope** and provides information on the record basic parameters.

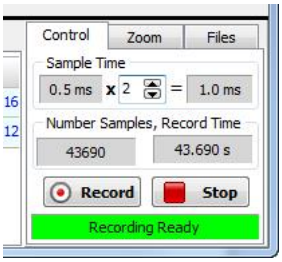


Fig. Auxiliary functions – Control

Record – starts the register record or enables the triggering algorithm.
Stop – stops the recording and plots the recorded register values in the chart at the same time, it sets the Zoom and all Axis parameters to default values (Fit to Window). Simultaneously, the zoom history is deleted and both cursors are disabled.
Sample Time – it shows the Oscilloscope sampling time. It is a multiple of Cycle_Time. A coefficient ranging from 1 to 64 is used to set it. More in Oscilloscope chapter.

Given Sample Time = 1 ms, the Oscilloscope will read 1000 register values in one second. If Sample time = 50 ms, the Oscilloscope will read 20 samples in one second.

Number Samples – the highest possible number of samples. It depends on the number of registers recorded and the type of their variables (number of bytes required for recording).
Record Time – maximum record time. It depends on the Number Samples values.

The shared memory of **TGM_Oscilloscope** has a size which is determined by the **TGM_System.HEADER.Mem_Size_Osc** register and is common for data of all recorded channels/registers. If a single channel values are recorded the entire memory is reserved to the record. If multiple channel data are recorded, the memory is divided uniformly among the channels. The distribution of the recorded data in the **TGM_Oscilloscope** memory and their offsets are determined by **TG Motion**; which saves these parameters into corresponding registers.

*The uniformity of the **TGM_Oscilloscope** memory distribution relates to the number bytes required to record the values of individual registers.*
*For example, in the case of two registers, the first being of Long Integer type (4 bytes) and the other of Double type (8 bytes), **TG Motion** will divide the shared memory of **TGM_Oscilloscope** in the ratio of 1:2, so that the same number of samples can be recorded in both channels.*

TG Motion and **Oscilloscope** status is displayed at the bottom of the tab.
Recording Ready, Waiting to Trigger, Offline and Version Conflict in due colors.

Recording procedure

Set up the registers in Channels tab (if need be, disable the triggering, too) and press the **Record** button. The record of selected registers will start. The user is informed of the recording progress at the bottom of the tab by Recording tag in a yellow box together with the progress bar informing of the **TGM_Oscilloscope** memory status. Press **Stop** button to stop recording.

The recording process will terminate if the size of the recorded values reaches the size of the **TGM_Oscilloscope** memory reserved for records.

After the recording is completed, the recorded values will be shown. The time axis will be set to match the recording duration and the y axis, according to the values recorded. Both cursors are disabled at the same time.

Zoom

Every change in magnification is saved in the memory. To navigate in the zoom history, functions of Zoom tab are used. To display the entire record, reset the zoom.

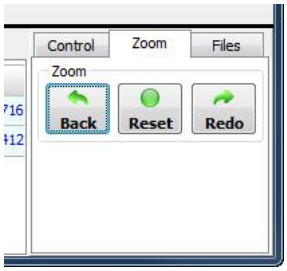


Fig. Auxiliary functions – Zoom

Back – a step backwards in the zoom history.
Reset – the entire chart is displayed (so called Fit to Window) and, at the same time, the zoom history is deleted.
Redo – a step forward in the zoom history.

To enlarge a certain part of the chart, i.e., to show its cropped part, select a rectangle in the graphics area by the mouse.

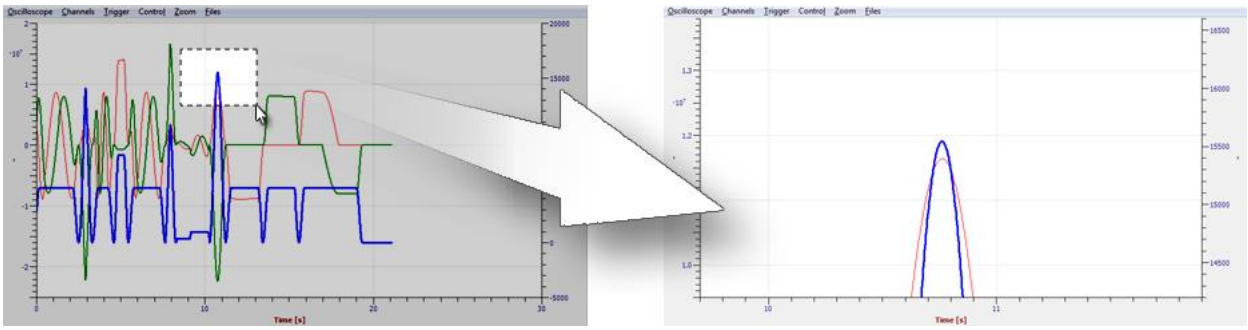


Fig. Enlarging a part of the chart using the mouse

Files

This tab is used to save and retrieve values and **Oscilloscope** setting and export of retrieved register values into the *.csv format.

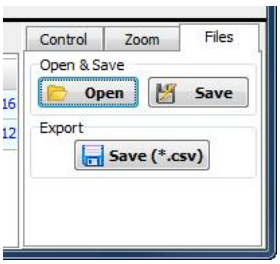


Fig. Auxiliary functions – Files

Open/Save

The tab comprises a couple of push buttons, namely, for opening and saving in *.Osc_CO2 format.

Open – opens a standard dialog box for file opening. After a file is selected, **Oscilloscope** will be set according to the parameters of the file being opened. Next, data is read into the **TGM_Oscilloscope** shared memory. Progress bar provides information on the loading progress. After the loading is completed, charts will be displayed automatically.

*If **Oscilloscope** contains already some data, the user will be alerted prior to opening a new file that these data will be overwritten by the new ones.*

Save – opens a standard dialog box for file saving. After the file name and location are chosen **Oscilloscope** saves its current settings and the **TGM_Oscilloscope** shared memory data to file. Progress bar provides information on the saving progress.

*The zoom history is not saved to *.Osc_CO2 files.*

Export

Allows to export the **TGM_Oscilloscope** memory data to *.csv formatted file. Subsequently, data may be processed by other programs.

Save (*.csv) – opens a standard dialog box for file saving. After the file name and location are chosen **Oscilloscope** will export the **GM_Oscilloscope** shared memory data to *.csv formatted file. Progress bar provides information on the export progress.

*The *.csv file contains the date and time of its creation, the names of the registers whose values have been recorded and samples as independent rows in T; v1; v2; …; vX format, where*
T = time
v1, v2, …, vX = values of registers 1, 2, …, X, where X is the number of simultaneously recorded registers.
Illustration of a *.csv file format:

```
3.12.2015 13:23:28

Time [s];Servo_000 -> Position;Servo_000 -> Speed
-5E-200;-2;1000
0.001;1;3000
0.002;5;4000
0.003;3;-2000
0.004;-1;-4000
0.005;-4;-3000
0.006;-2;2000
0.007;1;3000
0.008;2;1000
0.009;-5E-200;-2000
0.01;1;1000
0.011;-5E-200;-1000
```

Graphic Viewer

Graphic Viewer utility description

This utility is intended for debugging PLC or other service programs running under Windows. **Control Observer** allows to launch one instance of this utility.

Graphic Viewer is a tool allowing graphical representation of values of a series of consecutive register of a single shared memory. The offset is plotted on the horizontal axis, and the register values, on the vertical axis. Data may be updated permanently. It is also possible to turn off the updating process and examine in detail the particular status, carry out calculations and measurements.

The displayed values can be saved in *.GV_Data file. The data can be retrieved and displayed in **Graphic Viewer.**

The user interface consists of the main menu, a chart and a service function section at the bottom of the box.

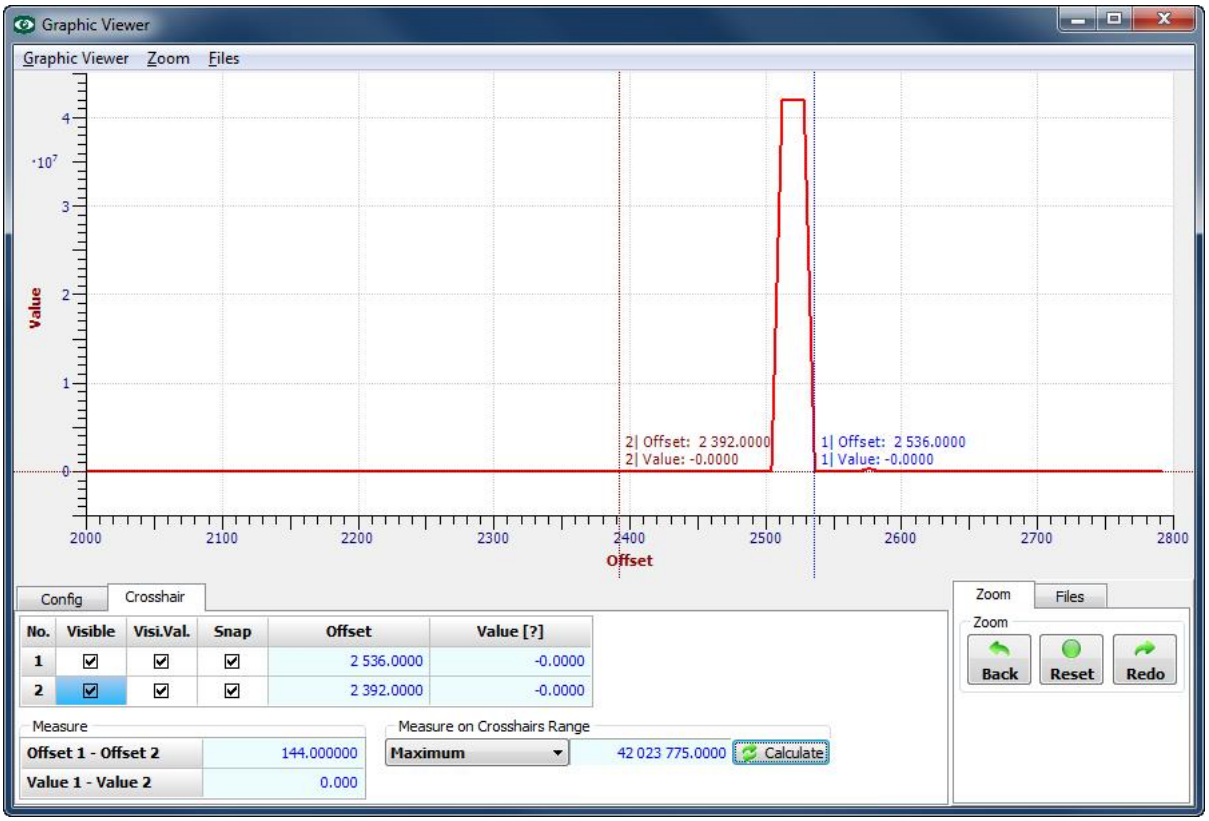


Fig. User interface of Graphic Viewer utility

Graphic Viewer menu

The menu contains the same functions and settings, which are conveniently available at the bottom of **Graphic Viewer** window. Therefore, for their detailed description, see the next paragraphs.

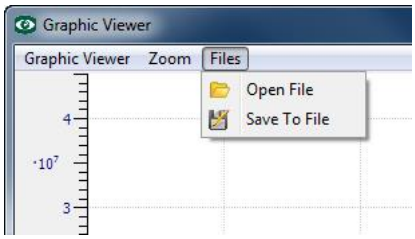


Fig. Graphic Viewer – Menu

Graphic Viewer – contains a command to close the Graphic Viewer.
Zoom – it contains commands for navigation in zoom history and for resetting the history.
Files – saves current data in *.GV_DATA file.

Graphic area

Values of selected register series are graphically represented here. Register offset is plotted on the horizontal axis, register values on the vertical axis. **Graphic Viewer** can adapt automatically the vertical resolution to the current content of the registers displayed.

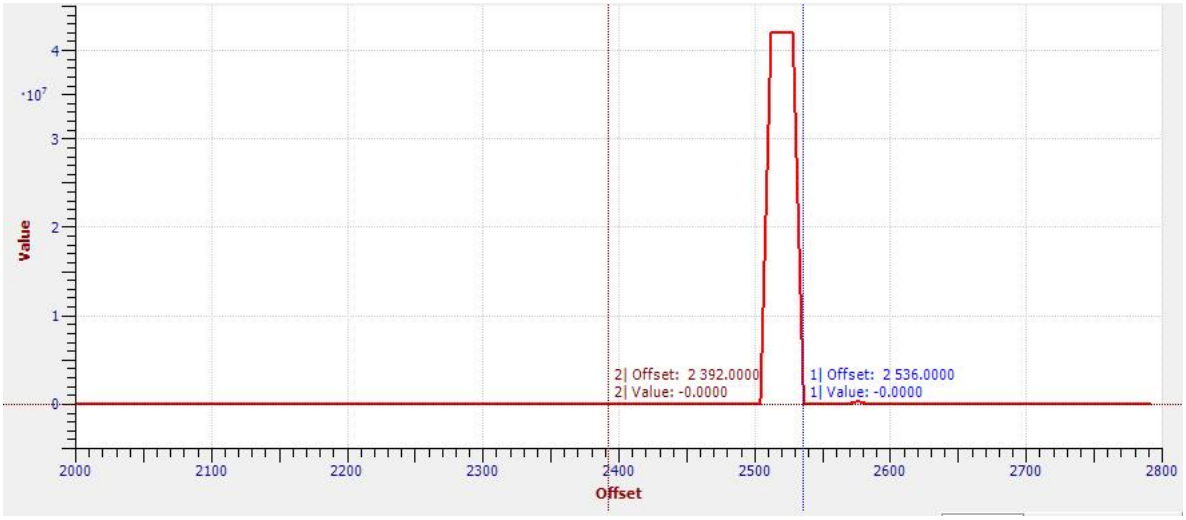


Fig. Graphic area of Graphic Viewer

The charts may contain **Crosshairs** (cursors), by means of which values can be read out, measurements and calculations can be carried out.

Zoom – chart cropping

To enlarge a certain part of the chart, i.e., to show its cropped part, select a rectangle in the graphics area by the mouse.

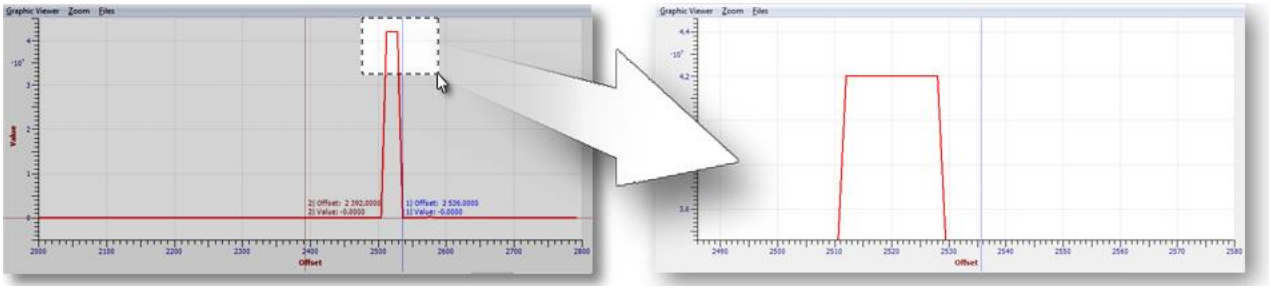


Fig. Selection of Zoom function by the mouse

*If **Auto Update** in **Config** tab has been enabled, the zoom is adjusted continuously (Fit to Window).*

Parameter area

At the bottom left part of the **Graphic Viewer** box there are two tabs to be used for displayed registers and cursors.

Config – it is used to select registers and set their representation.
Crosshair – cursor setting.

Config

This tab is used to select registers and set their representation format.

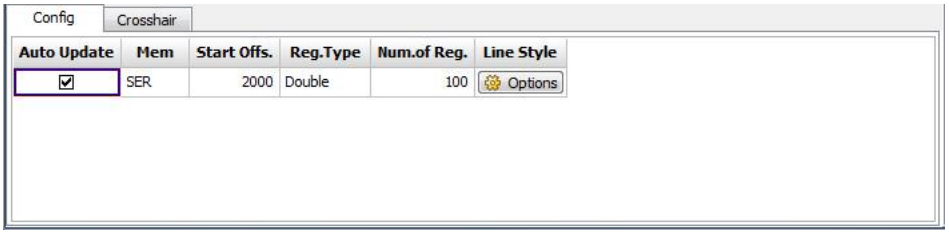


Fig. Graphic Viewer utility Config tab

Following items are available

Auto Update – if enabled, the register values are continuously read and plotted in the chart. At the same time, the range of y axis is changed automatically depending on the current range of the displayed registers. Enabling Auto Update disables Visible parameter of both cursors.
Mem – shared memory section whose content is to be displayed.
Start Offs. – pointer to the memory place where the first displayed register is located.
Reg.Type – type of variables being read from the memory. How many memory bytes are considered by Graphic Viewer as a single displayed register.

***Reg.Type** need not correspond to the actual type of variables existing in the part of the memory concerned. **Reg.Type** specifies the type of the variable into which (by how many bytes) the shared memory part content will be loaded for displaying.*

Num.of Reg. – number of registers displayed. Minimum is 3, the maximum number of register displayed depends on the size of the memory section to be displayed and on the values of ***Start Offs***. and ***Reg.Type***.

Graphic Viewer *offers data display in the scope of a single shared memory only. Data of multiple different shared memories cannot be displayed.*

Line Style – opens a dialog box to set up the color and style of the chart as well as the representation of the chart points.

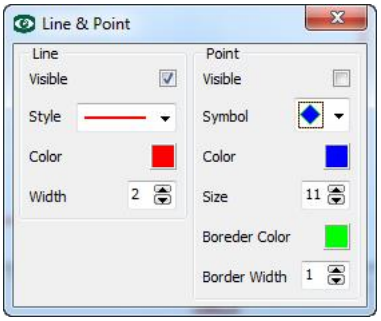


Fig. Setting up the chart visual parameters

Line
Visible – enables the representation of a continuous chart.
Style – offers the style of displaying a line in a chart (continuous, dashed, dash-dotted, etc.).
Color – sets up the color of the line.
Width – sets up the width of the line.

Point
Visible – enables displaying the measurement readings in the form of points.
Symbol – offers a choice of shapes of the displayed points.
Color – sets up the color of the points.
Color – sets up the size of the points.
Border Color – sets up the color of the border lines.
Border Width – sets up the width of the border lines.

Both representation types can be combined arbitrarily. At least one representation must be active.

Adjust the column width by dragging the division line between the headings.

Fig. Changing the table column width

Crosshairs

This tab comprises the setting for two cursors, which are used to take readings and carry out measurements. These cursors are shown in the form of a cross in the graphics of the **Graphic Viewer**.

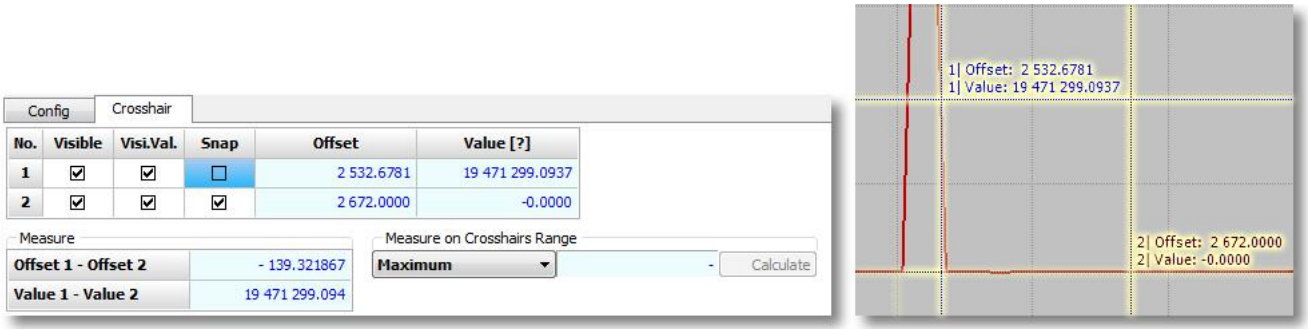


Fig. Crosshair (measuring cursor). Parameters and display in the graphic area

Crosshair setting table – parameters

Visible – enables or disables the cursor.

*If **Auto Update** parameter of **Config**, tab is enabled, every value reading will disable **Visible** parameter of both cursors.*

Visi.Val. – allows to display offset and value in the graphic part.
Snap – makes it possible to snap the cursor to the chart.
Offset – shows the offset of the cursor current position.
Value [?] – indicates the value of the displayed register at the cursor current position.

Measure

Offset 1 – Offset 2 – displays the measurement interval size.
Value 1 – Value 2 – displays the difference between the cursor positions.

Measure is only functional if both cursors are active –parameter Visible.

Measure on Crosshairs Range

It is used to calculate the values within the interval defined by the two cursors.
drop-down bar – it is used to select the calculation of maximum, minimum, average or root-mean-square value.
value box – shows the calculation result.
Calculate – this push button is used to carry out the calculation.

*The Calculate push button is only enabled if **Snap** is enabled for both cursors (both cursors are snapped to the chart and show exact chart values).*

Auxiliary function

At the bottom right corner of **Graphic Viewer** window there are two tabs intended to control the zoom and to save and retrieve the data.

Zoom – allows navigation in zoom history and its resetting.
Files – is used to save data in ***.GV_DATA** file and to retrieve them again.

Zoom

Every change in magnification is saved in the memory. To navigate in the zoom history, functions of Zoom tab are used. To display the entire record, reset the zoom.

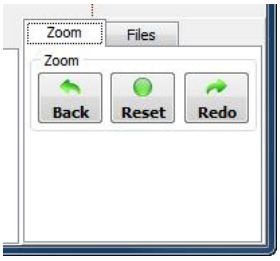


Fig. Graphic Viewer Zoom tab

The tab comprises three push buttons
Back – a step backwards in the zoom history.
Reset – the entire chart is displayed (Fit to Window) and, at the same time, the zoom history is deleted.
Redo – a step forward in the zoom history.

To enlarge a certain part of the chart, i.e., to show its cropped part, select a rectangle in the graphics area by the mouse directly in the chart.

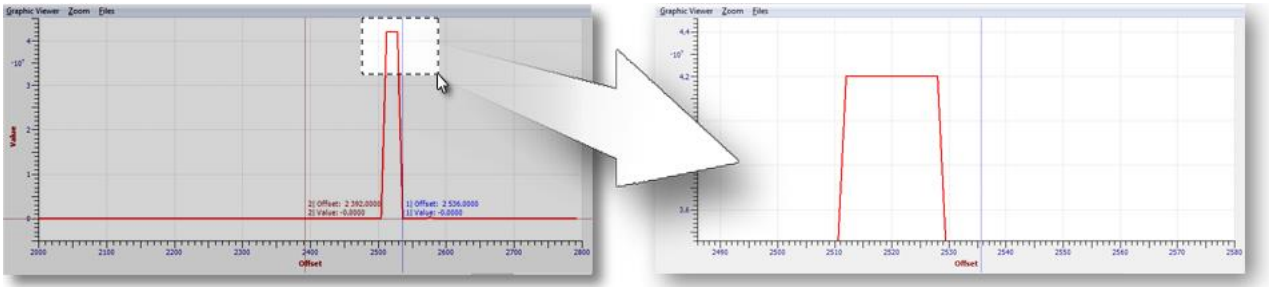


Fig. Enlarging a part of the chart using the mouse

*Provided that **Auto Update** is enabled, at every data retrieval the zoom will be adjusted automatically for the **Graphic Viewer** to show all data.*

Files

The tab is intended for saving and retrieving **Graphic Viewer** data.

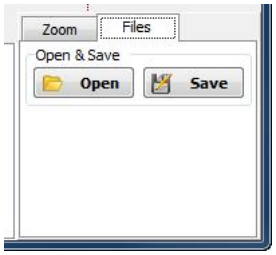


Fig. Graphic Viewer – Files tab

Open & Save

The tab comprises a couple of buttons, namely, for opening and saving in the format of ***.GV_Data** files.

Open – opens a standard dialog box for file opening. After a file is chosen, Graphic Viewer will import the chart data. Progress bar provides information on the progress. After the data are loaded, charts will be displayed automatically. Parameters of Config tab will be set at the same time.

*If **Graphic Viewer** contains already some data, the user will be alerted prior to opening a new file that these data will be overwritten by the new ones.*

***Auto Update** parameter being enabled, the imported data are updated according to the content of the shared memory displayed.*

Save – opens a dialog box for file saving. After the file name and location are chosen, Graphic Viewer will save the current data into a formatted text file. Progress bar provides information on the saving progress.

*The heading of ***.GV_Data** file contains the settings of the different parameters as well as information on the number of data items. The // Data item is followed by the values of the displayed registers, being shown in separate rows.*

Part of *.GV_Data file (example):

```
// Memory type: | SYS | DAT | CAM | OSC | SER | DIO | IPL | I_W | I_R
SER
// Start offset
0
// Data type: | I32-Dec | I32-Hex | I64-Dec | I64-Hex | Double
Double
// Number of Data
200
// Data
2.12199579096527E-314
1.06099789553204E-313
4.85922673504069E-271
6.36598737289582E-314
4.94065645841247E-324
3.95252516672997E-323
0
9.89486637328638E-311
2.12199574155871E-314
0
0
0
0
0
0
```

Select Registers

Select Registers is an auxiliary utility for **Oscilloscope** and **Watch Lists**. It is used to select the registers to be inserted in the mentioned utilities. **Control Observer** allows to launch one instance of **Select Registers**.

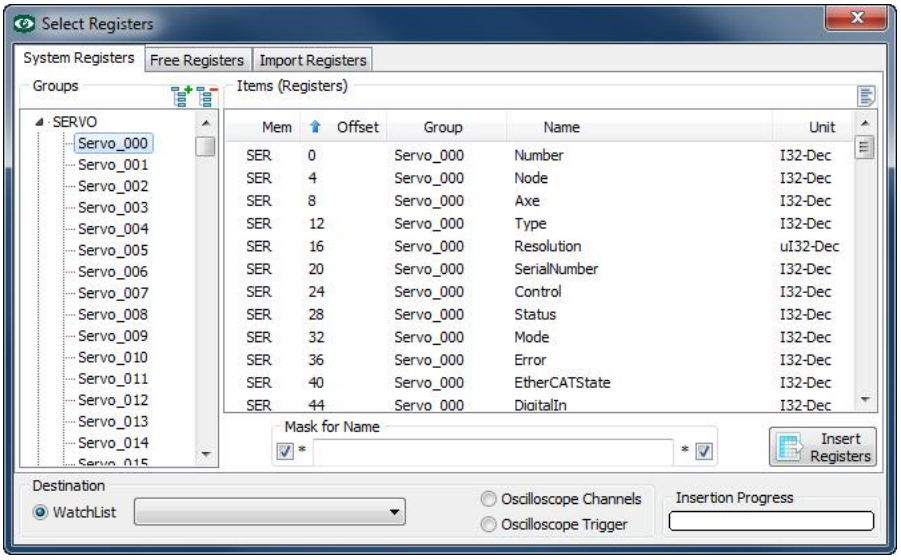


Fig. Select Registers utility window

The window consists of an upper part intended for register selection and a lower part in which a target utility for register insertion can be chosen.

Destination (insertion target)

It is used to select the target utility, into which the selected registers will be inserted.

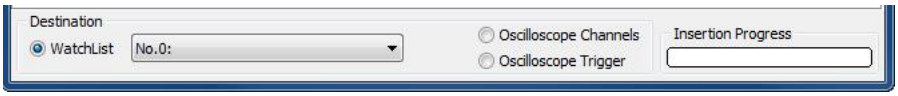


Fig. Select Registers – Destination box

Three options are available:
WatchList – inserts selected registers into the chosen WatchList.
Oscilloscope Channels – inserts selected registers into Channels tab of Oscilloscope.
Oscilloscope Trigger – inserts selected registers into Trigger tab of Oscilloscope.

If **Select Registers** utility is called directly from **Oscilloscope** (Channels or Trigger tabs) or from a particular **WachList**, the parameters of the Destination box will be set automatically to the relevant values.

WatchList

If the WatchList option is enabled, selected registers will be (after ***Insert Registers*** command) inserted into the selected **WatchList.** The target **WatchList** will be set in the drop-down bar next to the "WatchList" radio button. In it a choice of existing **WatchLists** is offered.

Fig. Select Registers – WatchList option

***WatchList** is a utility designed to track the values of selected registers and allowing to insert explicit values into the different registers. See Chapter **WatchLists**.*

*No **WatchList** exists on starting the **Control Observer**. **New WatchList** command in **WatchLists** item in **Control Observer** menu is used to establish a new **WatchList**.*

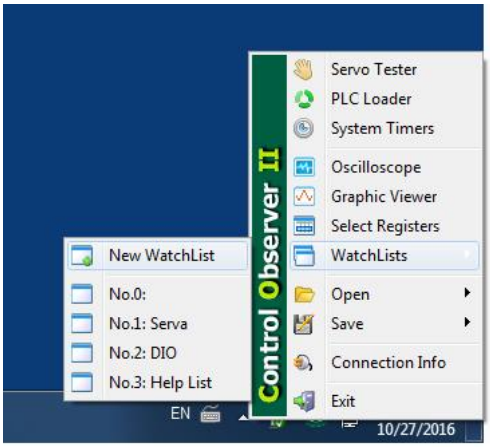


Fig. Establishing a new WatchList

A **WatchList** may contain up to 600 items. The user will be informed on the maximum number being exceeded. The **WatchList** in question will then be completed from selected registers to contain the maximum number of registers.

*One **WatchList** may contain several identical registers.*

Oscilloscope Channels

This option will insert selected registers into **Oscilloscope** utility, **Channels** tab. (see chap. Oscilloscope).

The **Oscilloscope/Channels** tab can contain up to 64 registers. The user will be informed on the maximum number being exceeded. The table is completed to contain the maximum register number.

Example: The table contains 50 items (registers), the user desires to add 20 registers. Only 14 first registers will be added (in this way, the maximum register number in the table is reached) and the addition of 6 more registers is ignored.

Oscilloscope Trigger

This option will insert selected registers into **Oscilloscope** utility, **Trigger** tab. (see chap. Oscilloscope)

The **Oscilloscope/Trigger** tab may contain up to 4 registers. The user will be informed on the maximum number being exceeded. The table is completed to contain the maximum register number.

Example: The table contains 2 items (registers), the user desires to add 5 registers. Only 2 first registers will be added (in this way, the maximum register number in the table is reached) and the addition of 3 more registers is ignored.

Insertion Progress

At the bottom on the right hand side of the window, there is a Progress bar informing on the register insertion process status.

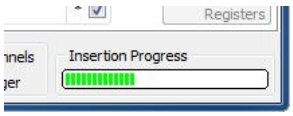


Fig. WatchList – Insertion process progress bar

Register selection area

The upper part of the window of **Select Registers** utility is divided into three tabs:

System Registers – it is used to select system registers, which are defined in TG Motion.
Free Registers – it allows to create any form/mask of the view in shared memories.
Import Registers – it offers data insertion from *.tgmr files.

**.tgmr files are text files containing information on registers and their data types. No register values are contained in *.tgmr files.*

System Registers

The tab is used to select system registers that are explicitly defined in **TG Motion**. It comprises two boxes for the selection definition, display filter and ***Insert Register*** command button.

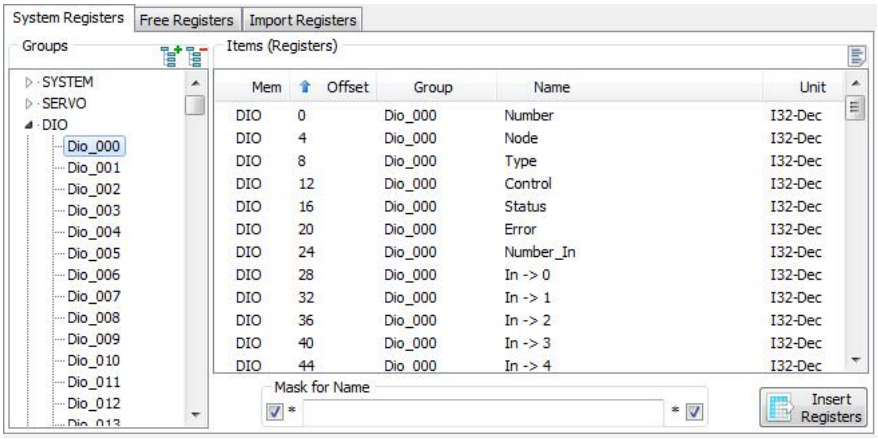


Fig. Select Registers – System Registers tab

Groups

This is a section for selecting register groups pursuant to the distribution of registers in the shared memories. The system register group is displayed as a drop-down tree, where at each branch all of its subfolders are available.

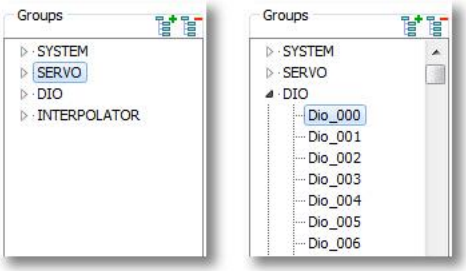


Fig. Drop-down list of register groups by shared memory

The user can select from five basic groups:
SYSTEM – contains system registers, Oscilloscope registers, Timers and others.
SERVO – offers the registers of all of the 256 virtual servo drives.
DIO – contains the register of all of the 256 virtual I/O modules.
INTERPOLATOR – offers the registers of all interpolators.
CNC_EX – contains registers of the CNC module

Clicking on any group makes its subgroups available. A subgroup may also be selected by means of multiple selection. A group may be selected by means of Shift key, any items may be added or removed by means of Ctrl key.

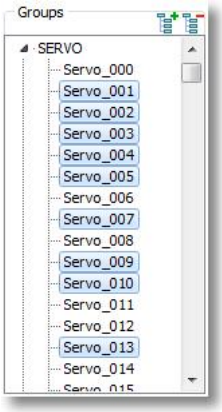


Fig. Multiple selection of register groups

Expand All, Collapse All
In the upper right corner of Groups box there are two icons intended to expand and collapse ale branches.

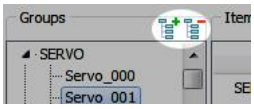


Fig. Expanding and collapsing all branches

Expand All – expands all branches.

Collapse All – collapses all branches.

Items (Registers)

It is used to select particular registers and provides basic information on them.

Items (Registers)

Mem	Offset	Group	Name	Unit
SER	40	Servo_000	EtherCATState	I32-Dec
SER	44	Servo_000	DigitalIn	I32-Dec
SER	48	Servo_000	DigitalOut	I32-Dec
SER	52	Servo_000	TorqueFeedForward	I32-Dec
SER	56	Servo_000	ControlWord	I32-Dec
SER	60	Servo_000	StatusWord	I32-Dec
SER	64	Servo_000	SysTimeDifference	I32-Dec
SER	464	Servo_000	WritePosition	Double
SER	472	Servo_000	Position	Double
SER	480	Servo_000	RefPosition	Double
SER	488	Servo_000	ExtPosition	Double
SER	496	Servo_000	Offset	Double

Fig. Items (Registers) window

The window is divided into five columns:
Mem – memory group (chosen in Groups box).
Offset – register offset in the range of Mem group.
Group – subgroup in the Mem group.
Name – name of the register.
Unit – type of the register variable.

When clicking on the heading of a column, the displayed registers will be sorted alphabetically with respect to this column. Repeated clicking on the same heading will induce ascending or descending sorting. The sorting method is indicated by an arrow in the column heading.

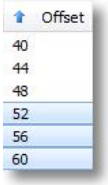


Fig. Ascending sorting of registers with respect to Offset column

The column width can be adjusted by dragging the division line between the headings.

Fig. Changing the column width

Selection of registers

A group may be selected by means of Shift key, any items may be added or removed by means of Ctrl key. **Select All** push button in the right upper corner of the box can be used to select all displayed registers.

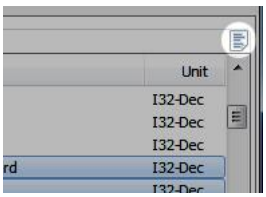


Fig. Select All icon for selecting all items

Mask for Name

To facilitate the orientation in the list of registers, a filter of register names is available in which conditions for register display can be set. Prefix, key word and suffix.

Fig. Window for the definition of the display mask

The term looked for

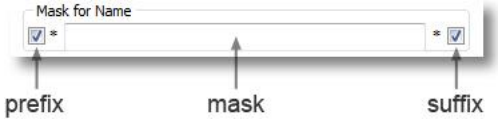
A key word may be inserted in the text box. The registers, which contain the specified term will be displayed. The remaining ones will be masked.

The character size is ignored. The masking routine considers Position and position to be the same terms.

Spaces at the beginning and at the end make a component of the mask, so that _position, position and position_ are three different terms from the masking routine viewpoint.

Prefix and Suffix

The parameters specify whether arbitrary characters may be placed in front of or behind the term being looked for or whether only a strictly defined plain term is to be searched for. Prefix (arbitrary characters in front of) and Suffix (arbitrary characters behind) may be enabled separately by means of check boxes in front of and behind the text box.



The whole masking template looks as follows: *Prefix & Mask & Suffix*.

In the following examples the term looked for is Position:

In neither Prefix nor Suffix is checked, only the plain term is looked for. The routine will display the registers named Position.

If only Prefix is checked, the routine will search for the registers, whose component is the term in question and, in addition, other character may be placed in front of it. Therefore, registers Position, WritePosition, RefPosition, etc. will be displayed.

If only Suffix is checked, the routine will search for the registers, whose component is the term in question and, in addition, other character may be placed behind it. Therefore, the routine will display registers Position, PositionError.

If both the Prefix and the Suffix is checked, the routine will search for the registers, whose component is the term in question and, in addition, other character may be placed in front of it and behind it. Therefore, the routine will display registers Position, WritePosition, RefPosition i PositionError.

Insert Registers

Registers may be added to the selected destination by pressing **Insert Registers** button at the bottom right corner of Registers box.

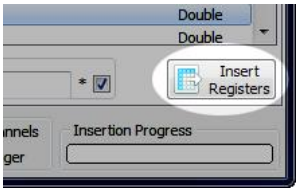


Fig. Insert button for inserting selected registers

Free Registers

This tab is used to create the user's own register and to insert it into the appropriate utility. A series/sequence of registers can be created by means of **Counter C1** and **Counter C2**. The newly created registers may not correspond to the system registers of **TG Motion** at the corresponding place of the shared memory both from the offset and the variable type viewpoint. This is a mask of the shared memory view. Once being created the new registers behave like regular registers. In **WatchLists**, their status may be tracked and values can be entered in them.

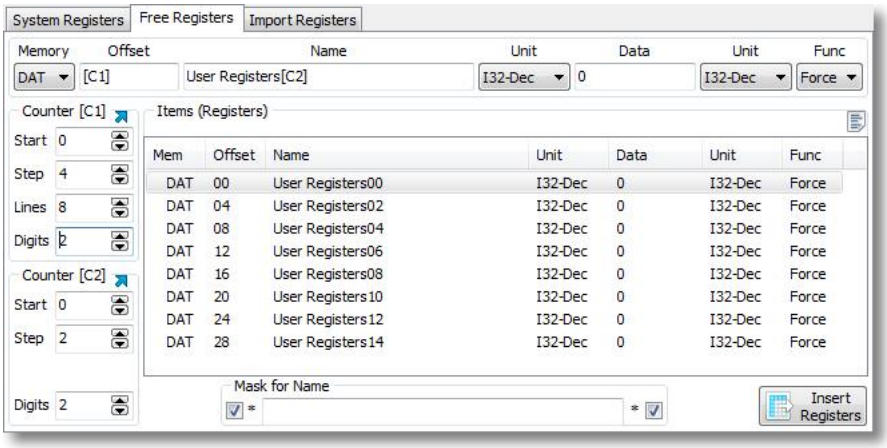


Fig. Free Registers window

Basic parameters of the user register can be defined in the upper box.

Fig. Heading of User Registers table

Memory – determines the shared memory in which the register is defined.
- ***SYS*** – memory of system registers timers, oscilloscope, etc.
- ***DAT*** – memory reserved for user data.
- ***CAM*** –memory designated for cam data.
- ***OSC*** – memory reserved for Oscilloscope record.
- ***SER*** – servo drive registers.
- ***DIO*** – I/O unit registers.
- ***IPL*** – interpolator memory.
- ***I_W*** – interpolator memory for writing from a Windows application.
- ***I_W*** – interpolator memory for reading from a Windows application.
- ***ODS*** – auxiliary debug listings.
- ***CNC*** – registers and data for controlling CNC modules.
- ***>G<*** – comprises a G-code.

The sizes of the memories are determined by **TGM_System.HEADER.Mem_Size_**xxx registers.

Offset – defines the register offset in the memory part specified by Mem value
Name – offers register naming.
Unit – defines the register variable type, the method of displaying the register value (in other words, how many memory bytes will be occupied by the variable).
Data – determination of the default value for inserting into the register.
Unit – defines the format for displaying and inserting the value of Data item.
Func – defines the method of inserting the Data value into the register.
- ***Force*** – direct value insertion.
- ***Set B*** – (set bit) sets the appropriate bit to 1. The Data parameters acts as a mask.
- ***Clr B*** – (clear bit) sets the appropriate bit to 0. The Data parameter acts as a mask.
- ***Tgl B*** – (toggle) inverts the bit value according to Data mask.

Two counters, by means of which a parameter sequence can be created, are used to define the register series. Instead of explicit values, counter values are inserted into the appropriate boxes.

Counter [C1]

It is used to generate a number series, by means of which the required parameters can be defined simultaneously, the number of newly created registers is determined by means of the counter (C1).

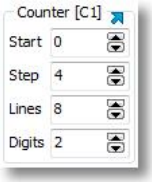


Fig. Counter C1

Start – counter initial value (0–999999).
Step – step between two values (1–65636).
Lines – number of newly created registers (1–256).
Digits – number of digits in the resulting value including the leading zeros (0–8).

To insert the counter parameters, the ***Insert Counter*** push button in the upper right corner of the box is used. When it is pressed, the counter is inserted at the cursor current position in the current box.

Counter [C2]

The other independent counter is used to generate the number series, by means of which the required parameters can be defined.

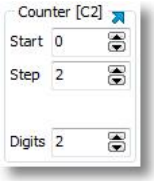


Fig. Counter C2

Start – counter initial value (0–999999).
Step – step between two values (1–65636).
Digits – number of digits in the resulting value including the leading zeros (0–8).

To insert the counter parameters, the ***Insert Counter*** push button in the upper right corner of the box is used. When it is pressed, the counter is inserted at the cursor current position in the current box.

Items (Registers)

It is used to select particular registers and provides basic information on them.

Items (Registers)

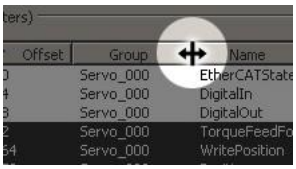
Mem	Offset	Name	Unit	Data	Unit	Func
SER	00	User Registers00	I32-Dec	0	I32-Dec	Force
SER	04	User Registers02	I32-Dec	0	I32-Dec	Force
SER	08	User Registers04	I32-Dec	0	I32-Dec	Force
SER	12	User Registers06	I32-Dec	0	I32-Dec	Force
SER	16	User Registers08	I32-Dec	0	I32-Dec	Force
SER	20	User Registers10	I32-Dec	0	I32-Dec	Force
SER	24	User Registers12	I32-Dec	0	I32-Dec	Force
SER	28	User Registers14	I32-Dec	0	I32-Dec	Force

Fig. Items (Registers) table

The window is divided into seven columns:

Mem – memory part (chosen in Groups box).
Offset – register offset in the framework of Mem group.
Name – name of the register.
Unit – type of the register variable.
Data – determination of the default value for inserting into the register.
Unit – defines the format for displaying and inserting the value of Data item.
Func – form of inserting the Data item into the register value.

The column width can be adjusted by dragging the division line between the headings.



Selection of registers

A group may be selected by means of Shift key, any items may be added or removed by means of Ctrl key. **Select All** push button in the right upper corner of the box can be used to select all displayed registers.

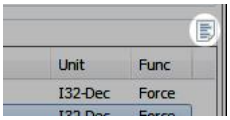


Fig. Select All icon for selecting all items

Mask for Name

To facilitate the orientation in the list of registers, a filter of register names is available in which conditions for register display can be set. Prefix, key word and suffix.

Fig. Window for the definition of the display mask

The term looked for

A key word may be inserted in the text box. The registers, which contain the entered term will be displayed. The remaining ones will be masked.

The character size is ignored. The masking routine considers Position and position to be the same terms.

Spaces at the beginning and at the end make a component of the mask, so that _position, position and position_ are three different terms from the masking routine viewpoint.

Prefix and Suffix

The parameters specify whether arbitrary characters may be placed in front of or behind the term being looked for or whether a strictly defined term is to be searched for Prefix (arbitrary characters in front of) and Suffix (arbitrary characters behind) may be enabled independently by means of check boxes in front of and behind the text box.

The whole masking template looks as follows: *Prefix & Term_looked_for & Suffix*.

In the following examples the term looked for is Position:

In neither Prefix nor Suffix is check, exactly plain term is looked for. The routine will display the registers named Position.

If only Prefix is checked, the routine will search for the registers, whose component is the term in question and, in addition, other character may be placed in front of it. Therefore, registers Position, WritePosition, RefPosition, etc. will be displayed.

If only Suffix is checked, the routine will search for the registers, whose component is the term in question and, in addition, other character may be placed behind it. Therefore, the routine will display registers Position, PositionError.

If both the Prefix and the Suffix is checked, the routine will search for the registers, whose component is the term in question and, in addition, other character may be placed in front of it and behind it. Therefore, the routine will display registers Position, WritePosition, RefPosition i PositionError.

Insert Registers

Registers may be added to the selected destination by pressing **Insert Registers** button at the bottom right corner of Registers box.

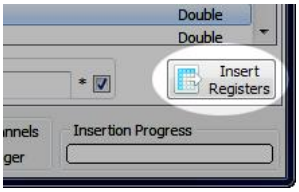


Fig. Insert Registers button for inserting selected registers

Import Registers

The tab is used to import *.tgmr files which contain a register list. The utility is employed to create and debug PLC, to import cam tables and, in general, to import data sequences of other applications into the **TG Motion** shared memories.

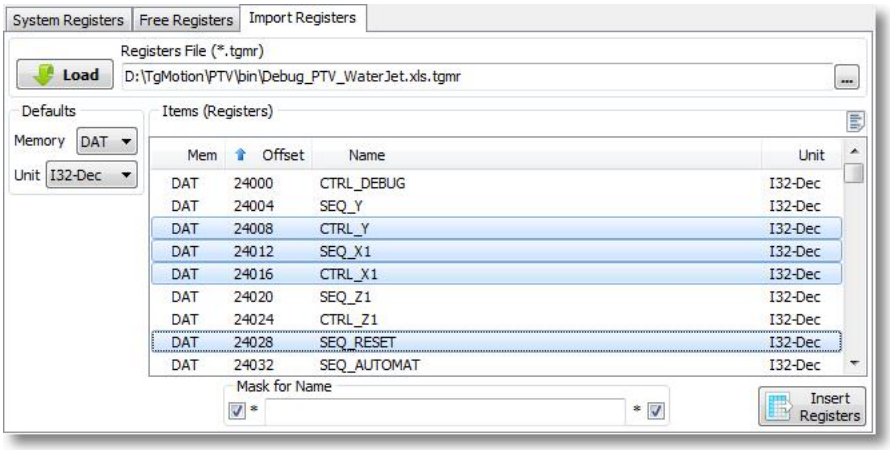


Fig. Import Registers tab

Registers File (*.tgmr)

This utility can open files of the *.tgmr type.

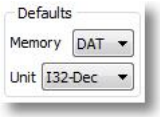
Write the path and the file name *.tgmr into the text box.

Use push button to open a standard dialog box for file opening.

This Load push button is used to import data from a file into the table of Items box, from which it can be added to **Oscilloscope** or **Watchlists**.

Defaults

The *.tgmr files comprise offsets, register names and variable types. Therefore, a part of the shared memory is to be specified, into which the registers will be imported.



Memory – specifies the shared memory section, into which the data will be imported.

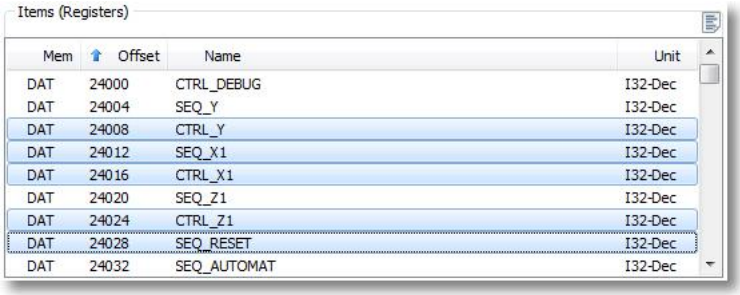
- ***SYS*** – memory of system registers timers, oscilloscope, etc.
- ***DAT*** – memory reserved for user data.
- ***CAM*** –memory designated for cam data.
- ***OSC*** – memory reserved for Oscilloscope record.
- ***SER*** – servo drive registers.
- ***DIO*** – I/O unit registers.
- ***IPL*** – interpolator memory.
- ***I_W*** – interpolator memory for writing from a Windows application.
- ***I_W*** – interpolator memory for reading from a Windows application.
- ***ODS*** – auxiliary debug listings.
- ***CNC*** – registers and data for controlling CNC modules.
- ***>G<*** – comprises a G-code.

The sizes of the memories are determined by **TGM_System.HEADER.Mem_Size_**xxx.

Unit – imported register default type. If the type is given in *.tgmr file, Unit will be ignored.

Items (Registers)

The import being completed, the values and parameters of the imported registers will be written in a table of ***Items*** box, from which they can be added to **Oscilloscope** or **WatchLists.** The items of the ***Items (Registers)*** table refer to places in the shared memory, to which the registers were imported.



Items (Registers)

Mem	Offset	Name	Unit
DAT	24000	CTRL_DEBUG	I32-Dec
DAT	24004	SEQ_Y	I32-Dec
DAT	24008	CTRL_Y	I32-Dec
DAT	24012	SEQ_X1	I32-Dec
DAT	24016	CTRL_X1	I32-Dec
DAT	24020	SEQ_Z1	I32-Dec
DAT	24024	CTRL_Z1	I32-Dec
DAT	24028	SEQ_RESET	I32-Dec
DAT	24032	SEQ_AUTOMAT	I32-Dec

Fig. Import Registers – Items (Registers)

The Items (Registers) box comprises four columns to show clearly the different parameters of the imported registers.

Mem – part of the memory to which the register values have been imported.
Offset – register offset in the framework of Mem group.
Name – name of the register.
Unit – type of the register variable.

Mask for Name

To facilitate the orientation in the list of registers, a filter of register names is available in which conditions for register display can be set. Prefix, key word and suffix.

Fig. Window for the definition of the display mask

The term looked for

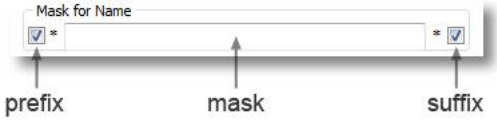
A key word may be inserted in the text box. The registers, whose title contains the specified term, will be displayed. The remaining ones will be masked.

The character size is ignored. The masking routine considers Position and position to be the same terms.

Spaces at the beginning and at the end make a component of the mask, so that _position, position and position_ are three different terms from the masking routine viewpoint.

Prefix and Suffix

The parameters specify whether arbitrary characters may be placed in front of or behind the term being looked for or whether a strictly defined term is to be searched for Prefix (arbitrary characters in front of) and Suffix (arbitrary characters behind) may be enabled separately by means of check boxes in front of and behind the text box.



The whole masking template looks as follows: *Prefix & Term_looked_for (Mask) & Suffix*.

In the following examples the term looked for is Position:

In neither Prefix nor Suffix is check, exactly plain term is looked for. The routine will display the registers named Position.

If only Prefix is checked, the routine will search for the registers, whose component is the term in question and, in addition, other character may be placed in front of it. Therefore, registers Position, WritePosition, RefPosition, etc. will be displayed.

If only Suffix is checked, the routine will search for the registers, whose component is the term in question and, in addition, other character may be placed behind it. Therefore, the routine will display registers Position, PositionError.

If both the Prefix and the Suffix is checked, the routine will search for the registers, whose component is the term in question and, in addition, other character may be placed in front of it and behind it. Therefore, the routine will display registers Position, WritePosition, RefPosition i PositionError.

Insert Registers

Registers may be added to the selected destination by pressing **Insert Registers** button at the bottom right corner of Registers box.

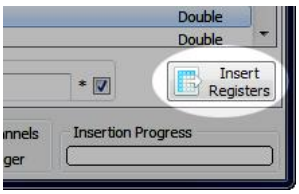


Fig. Insert button for inserting selected registers

Watch Lists

It is a utility designed for tracking any combination of as many as 600 registers, allowing direct value insertion into the different registers. **WatchList** utility allows to open several instances. Maximum number of **WatchLists** is 16.

New WatchList

No **WatchList** exists on starting the **Control Observer**. To establish a new **WatchList** a ***New WatchList*** command from **Control Observer** menu is to be used.

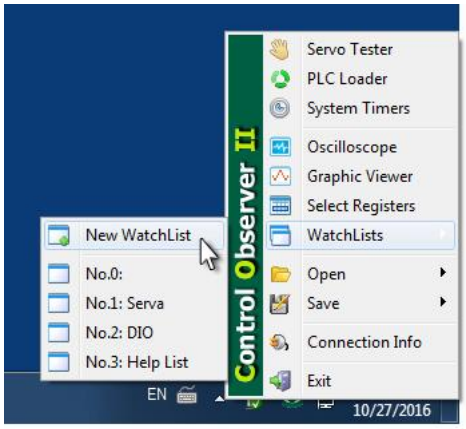


Fig. Establishing a new WatchList from Control Observer menu

Once a new **WatchList** is established, it opens automatically. Maximum number of **WatchLists** is 16.

WatchList opening

To open an already existing **WatchList**, use an item from **Control Observer** ***WatchLists/No.x*** menu, where there is a list of already existing **WatchLists.** The number of the **WatchList** is followed its name (for the naming procedure see the paragraphs below).

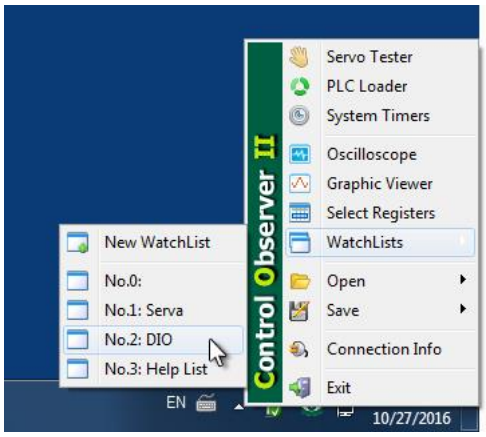


Fig. Opening an existing WatchList from Control Observer menu

WatchList

The **WatchList** window can be divided into three parts. Menu, service function push buttons and register table.

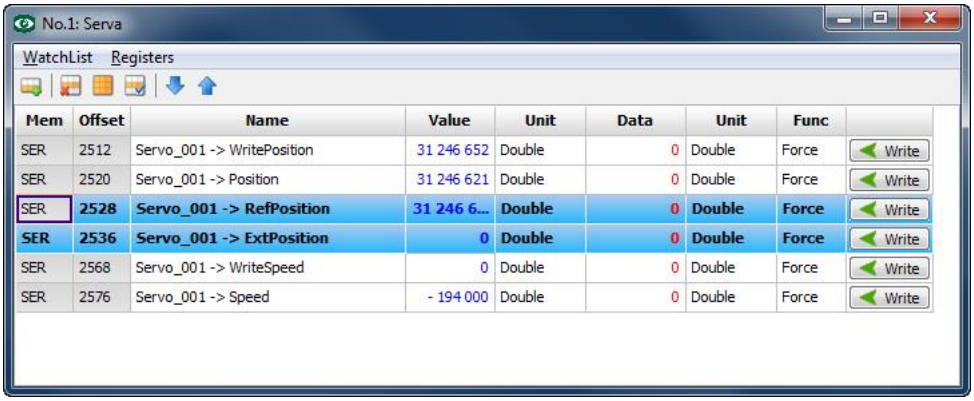


Fig. WatchList window

Menu

The menu of every **WatchList** comprises two items:

WatchList – offers three commands related to a particular WatchList.

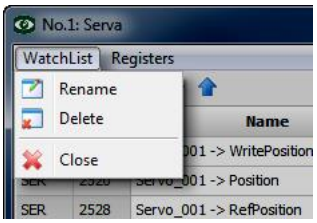


Fig. WatchList – menu – WatchList

Rename – to rename **WatchList**. The new name will appear in the heading of **WatchList** following the numerical designation of **WatchList No.x:**, where **x** is the number of the **WatchList**. In the same way, they will be displayed in the list of WatchLists in the **Control Observer** menu. Default value: empty string.

Delete – deletes the **WatchList** including its settings.

Close – closes the **WatchList** window. However, the WatchList continues to exist, including its parameters.

Registers – it is designed to handle the **WatchList**.table items. As the same functions are concerned here as those offered by the function push buttons of the service functions, they will be dealt with in detail in the next paragraph.

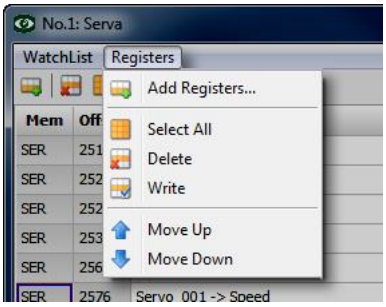


Fig. WatchList – menu – Registers

Function Buttons

They are used to add, remove, select and handle the registers in the table of **WatchList.**

Fig. WatchList – function push buttons

Add Register…

After **Add Register** tab is selected, **Select Registers** dialog box is open to select and add registers, whose values are to be displayed in **Oscilloscope.** For a detailed description of **Select Registers** utility, see the **Select Registers** chapter.

The registers which were added are shown in a table, in which their parameters are displayed and which allows to set the parameters of their graphic representation.

Every WatchList may contain 600 items at the most. The user will be informed on the maximum number being exceeded. WatchList is subsequently completed to contain the maximum register number.

WatchList may contain several identical registers.

Delete

It removes the selected register or registers from the table.

To select a register in the table just click on any cell, which pertains to it. To select multiple registers or table cells use the Drag&Drop method over the cell area pertaining to the required registers.

Select All

It selects all items of the table.

Write

It is used to multiple writing of prepared values into registers. After the **Write** push button is pressed, the value of ***Data*** cell is written in ***Value*** cell for all selected registers.

*Example: Values to be inserted into the Value column can be found in Data column of selected registers. When **Write** push button is pressed, the values of **Data** column will be copied into the **Value** column for the selected registers only. Other registers will not be copied.*

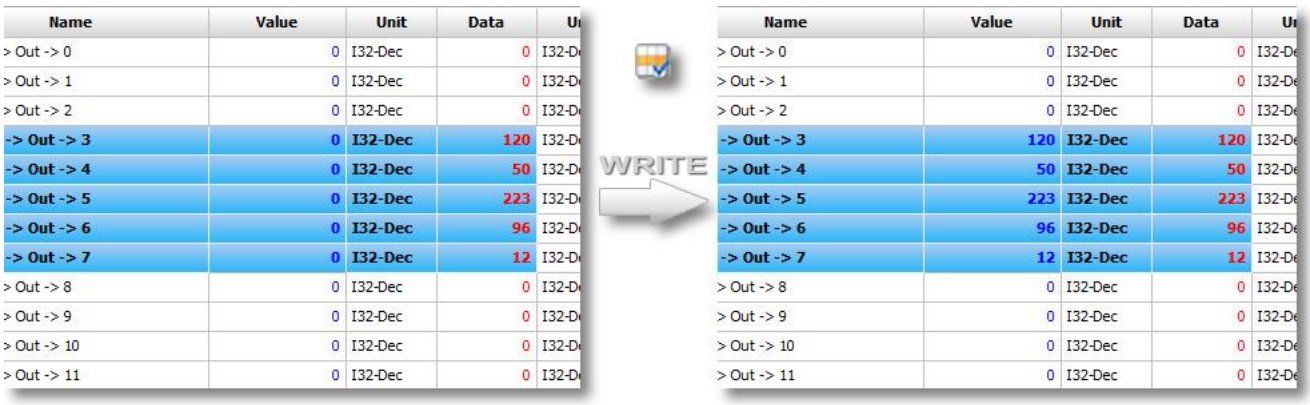


Fig. Multiple value copy into registers by means of Write function

Move Down, Move Up
It moves the selected rows of the table (registers) by one position upwards or downwards.

Register table

The register table displays in a well arranged way the inserted registers, their values and allows direct value insertion into a register or bit manipulations. The rows are representing the different registers. The table has nine columns.

Mem	Offset	Name	Value	Unit	Data	Unit	Func	
SER	464	Servo_000 -> WritePosition	0	Double	0	Double	Force	Write

Mem – part of the shared memory in which the selected register is situated.
- ***SYS*** – memory of system registers timers, oscilloscope, etc.
- ***DAT*** – memory reserved for user data.
- ***CAM*** –memory designated for cam data.
- ***OSC*** – memory reserved for Oscilloscope record.
- ***SER*** – servo drive registers.
- ***DIO*** – I/O unit registers.
- ***IPL*** – interpolator memory.
- ***I_W*** – interpolator memory for writing from a Windows application.
- ***I_W*** – interpolator memory for reading from a Windows application.
- ***ODS*** – auxiliary debug listings.
- ***CNC*** – registers and data for controlling CNC modules.
- ***>G<*** – comprises a G-code.

The sizes of the memories are determined by **TGM_System.HEADER.Mem_Size_**xxx registers.

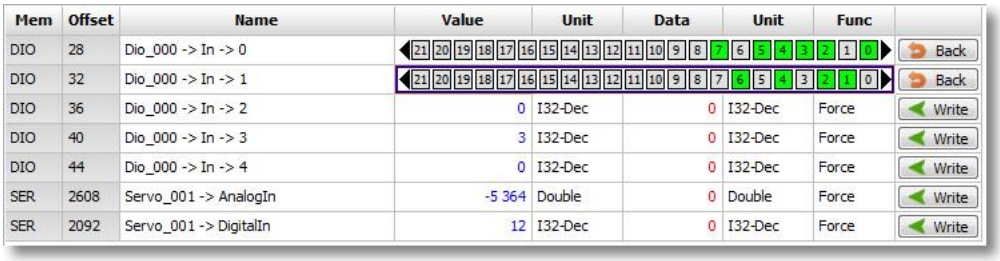
Offset – register offset in the framework of Mem group.

Name – name of the register. If need be, the name can be changed – the content of a table cell can be changed. The new name is only valid for the current **WatchList**. The renaming will not influence the name of the register proper.

Value – register current value.

Unit – register display format.

If I32-BinEx or I64-BinEx are selected, the register value bit mask will be displayed in the Value box instead of a numerical value. Back push Back *button restores the register numerical value display.*



Mem	Offset	Name	Value	Unit	Data	Unit	Func	
DIO	28	Dio_000 -> In -> 0	21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0					Back
DIO	32	Dio_000 -> In -> 1	21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0					Back
DIO	36	Dio_000 -> In -> 2	0	I32-Dec	0	I32-Dec	Force	Write
DIO	40	Dio_000 -> In -> 3	3	I32-Dec	0	I32-Dec	Force	Write
DIO	44	Dio_000 -> In -> 4	0	I32-Dec	0	I32-Dec	Force	Write
SER	2608	Servo_001 -> AnalogIn	-5 364	Double	0	Double	Force	Write
SER	2092	Servo_001 -> DigitalIn	12	I32-Dec	0	I32-Dec	Force	Write

Fig. Displaying the register value in the form of a bit mask

Data – value to be inserted into a register (into Value cell) or a bit mask.

Unit – format for displaying and inserting the value of Data item.

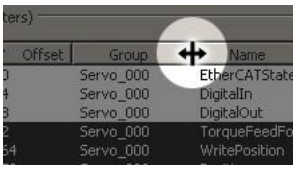
Func – form of interaction of *Data* item with the register value (*Value* cell) after the *Write* push button is pressed. *Func* has four possibilities.
- ***Force*** – direct insertion of *Data* value into *Value* item.
- ***Set B*** – binary representation of *Data* value acts as a mask according to which the corresponding bits of *Value* are set.
- ***Clr B*** – binary representation of *Data* value acts as a mask according to which the corresponding bits of *Value* are cleared (set to zero).

Tgl B – binary representation of *Data* value acts as a mask according to which the corresponding bits of *Value* are inverted.

*If **Func** = **Set B**, **Clr B** or **Tgl B**, the **Data** value acts as a bit mask for handling the **Value** bits.*
*The bits of the **Data** mask, which have been set (bit=1) specify the **Value** bits which can be changed. The bits of the **Data** mask, which are zero, specify the **Value** bits which will not be influenced by any manipulation.*

Write – command push button for inserting the *Data* value into the *Value* cell (register value).

The column width can be adjusted by dragging the division line between the headings.



Open

The Open command is used to open projects, read data or import the settings of different utilities.

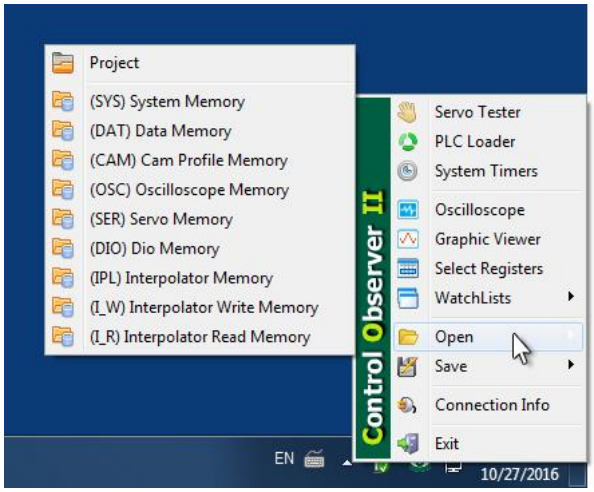


Fig. Open command from Control Observer menu

Project

A Project is defined as list of **Control Observer** open utilities, including their parameter settings and layout of their windows in the desktop. In **TG Motion**, one can deal with several jobs and a separate project can be used for each of them. Multiple projects may also be employed when a single work station is used by several users.

Open/Project being selected, a standard dialog box for file opening will open. The type of the project files is *.Proj_CO2.

Shared memory content import

Further options are opening the files in which the content of the corresponding part of the shared memory is saved. When the file is open, the target content of the memory will be overwritten by the data, which are contained in the file being opened. The memory part selection results from the file extension. Therefore, the memory content cannot be overwritten by the data which are designated for other utilities (which have been saved in another memory part).

(SYS) System Memory

It opens a file of *.Mem_SYS type, whose data will overwrite the SYS shared memory designated for **TG Motion** system data.

(DAT) Data Memory

It opens a file of *.Mem_DAT type, whose data will overwrite the DAT shared memory designated for user data.

(CAM) Cam Profile Memory

It opens a file of *.Mem_CAM type, whose data will overwrite the CAM shared memory designated for cam data.

(OSC) Oscilloscope Memory

It opens a file of *.Mem_OSC type, whose data will overwrite the OSC shared memory, which is reserved for the data recorded by **Oscilloscope** utility.

(SER) Servo Memory

It opens a file of *.Mem_SER type, whose data will overwrite the SER shared memory designated for virtual servo drive registers.

(DIO) Dio Memory

It opens a file of *.Mem_DIO, whose data will overwrite the DIO shared memory designated for I/O registers.

(IPL) Interpolator Memory

It opens a file of *.Mem_IPL type, whose data will overwrite the IPL shared memory reserved for interpolator data.

(I_W) Interpolator Write Memory

It opens a file of *.Mem_I_W type, whose data will overwrite the I_W shared memory reserved for interpolator writing.

(I_R) Interpolator Read Memory

It opens a file of *.Mem_I_R type, whose data will overwrite the I_R shared memory reserved for data reading by interpolators.

*Entire content of the respective shared memory is always saved in the different files, irrespective of the particular memory part, which was employed at the rime of saving (**Oscilloscope**, Cam, etc.).*

Save

The Save command is used to save projects, to save data and the settings of different utilities.

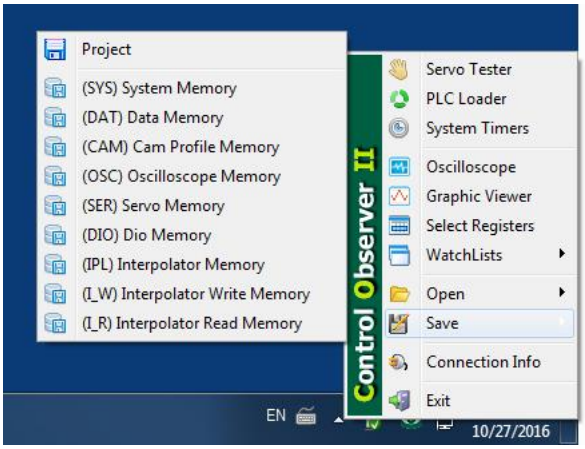


Fig. Save command from Control Observer menu

Project

A Project is defined as list of **Control Observer** active utilities, including their parameter settings and deployment of their windows in the desktop.

Save/Project being selected, a standard dialog box for file saving will open. The type of the project files is *.Proj_CO2.

Shared memory content export

Further options of *Save* command are creating the files in which the content of the corresponding part of the shared memory is saved. The files differ from each other by the extension, which corresponds to the shared memory part, whose data it contains.

(SYS) System Memory

Data of the SYS shared memory (designated for **TG Motion** system data) will be saved in a *.Mem_SYS type file.

(DAT) Data Memory

Data of the DAT shared memory (designated for user data) will be saved in a *.Mem_DAT type file.

(CAM) Cam Profile Memory

Data of the CAM shared memory (designated for cam data) are saved in a *.Mem_CAM type file.

(OSC) Oscilloscope Memory

Data of the OSC shared memory (memory for data recorded by **Oscilloscope** utility) will be saved as a *.Mem_OSC type file.

(SER) Servo Memory

Data of the SER shared memory (designated for virtual servo drive registers) will be saved in a *.Mem_SER type file.

(DIO) Dio Memory

Data of the DIO shared memory (designated for I/O registers) will be saved in a *.Mem_DIO type file.

(IPL) Interpolator Memory

Data of the IPL shared memory (reserved for interpolator data) will be saved in a *.Mem_IPL type file.

(I_W) Interpolator Write Memory

Data of the I_W shared memory (reserved for interpolator data writing) will be saved in a *.Mem_I_W type file.

(I_R) Interpolator Read Memory

Data of the I_R shared memory (reserved for interpolator data reading) will be saved in a *.Mem_I_R type file.

Entire content of the respective shared memory is always saved in the different files, irrespective of the particular memory part, which was employed at the time of saving.

*Example: **Oscilloscope** utility used 300 000 bytes for recording, however, Save/OSC utility will save the content of the entire memory of **TGM_Oscilloscope**.*

TG Drives, s.r.o.
Olomoucká 1290/79
CZ - 627 00 Brno

Tel.: +420 548 141 811
Fax: +420 548 141 890
E-mail: info@tgdrives.cz

Connection Info

This utility is used to set and check the interconnection between **Control Observer** and **TG Motion** and to assist with troubleshooting. It provides the graphical representation of active components, present libraries, component interconnection and, furthermore, information on component versions and their compatibility.

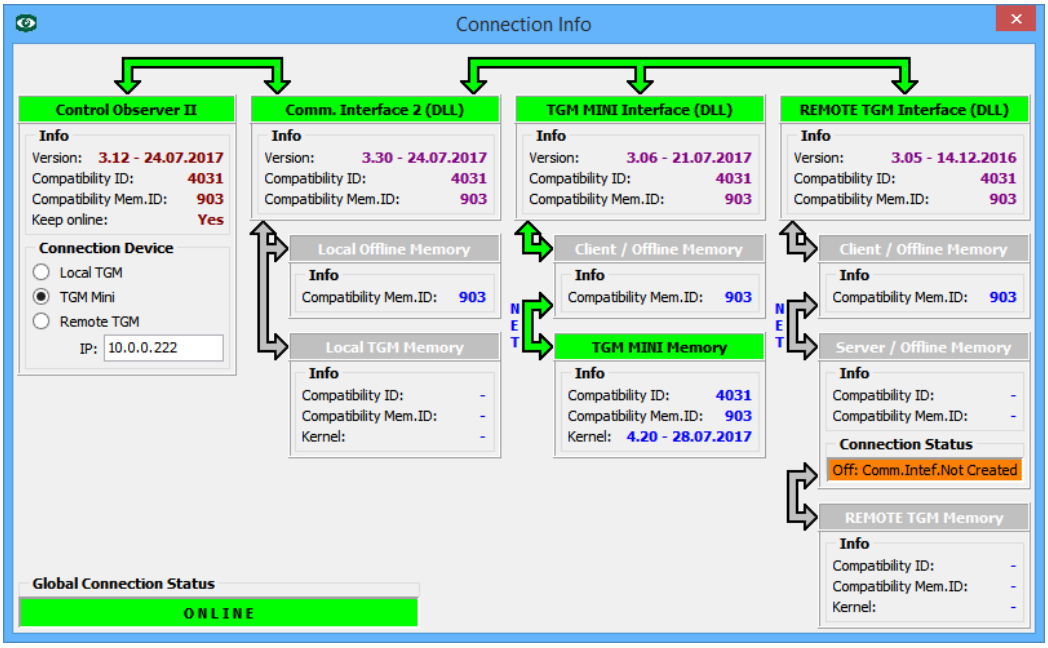


Fig. Connection Info utility

Active and functioning components are shown in green color, whereas non-functional or inactive components are grey. The same colors apply to the arrows representing the component interconnection.

Interconnection options

Control Observer connects to Comm Interface telling it in which direction it intends to continue communicating. There are three options.

TG Motion on the same computer as Control Observer.
TGM Mini – external box with implemented TG Motion called TGMmini.
TG Motion on another computer. Network connection may also take part in the communication here.

Connection components

Control Observer – provides general information on Control Observer II.
Comm. Interface 2 (DLL) – provides information on TG Motion components installed on the same computer as Control Observer II.
TGM MINI Interface (DLL) – provides information on the interconnection with TGMmini.
REMOTE TGM Interface (DLL) – provides information on the interconnection with TG Motion on another computer via a network.

All of these four components comprise in their box three parameters to determine the compatibility of the modules entering the communication.

Version – version number.
Compatibility ID – defines the version mutual compatibility.
Compatibility Memory ID – determines the compatibility with shared memories.

Control Observer

Info
The Info box provides general information on the version and on the compatibility of Control Observer II.

Version – version of Control Observer (comprising the version number and the distribution date).
Compatibility ID – code specifying the compatibility with TG Motion (with dll).
Compatibility Memory ID – determines the compatibility with shared memories.
Keep online – displays the Control Observer status.

Connection Device
It is used to choose a target device with running **TG Motion**, to which we desire to connect **Control Observer**.

Local TG Motion – implements the interconnection with TG Motion running on the same computer.
TGM Mini – interconnection with TG Motion running on TGMmini.
Remote TG Motion – interconnection with TG Motion running on another computer, via a network.
IP – address of the computer to which we wish to implement the interconnection.

*On the computer, to which we wish to implement the interconnection, there must be running **TG Motion** compatible with the **TG Motion** version on the computer to be connected.*

Global Connection Status

It displays the current status of the whole **TG Motion** system (Online, Offline, version conflict).

Comm. Interface 2 (DLL)

Interface 2 (DLL) – provides information on TG Motion components installed on the same computer as **Control Observer II**.

Version – version of Control Observer (comprising the version number and the distribution date).
Compatibility ID – code specifying the compatibility with TG Motion.
Compatibility Memory ID – code that determines the compatibility with shared memories.

Local Offline Memory
– shared memories in offline mode with **Control Observer** running.
Compatibility Memory ID – code that determines the compatibility with shared memories.

Local TG Motion Memory
– existing shared memories created by **TG Motion** – **Control Observer** is in online mode.
Compatibility ID – code specifying the compatibility with TG Motion.
Compatibility Memory ID – code that determines the compatibility with shared memories.
Kernel – TG Motion version.
MathDll – transformation mathematical library (not available from version 420 upwards – Compatibility ID 4031 and higher).

TGM MINI Interface (DLL)

It provides information on the connection to **TGMmini**.

TGMmini

It is an external single-purpose device on which **TG Motion** is running. It is designed to control the servo drives and I/O boxes without using another PC.

Version – version of Control Observer (comprising the version number and the distribution date).
Compatibility ID – code specifying the compatibility with TG Motion.
Compatibility Memory ID – code that determines the compatibility with shared memories.

Client/Offline Memory

– shared memory client address/code for **Control Observer** in offline mode.
Compatibility Memory ID – code that determines the compatibility with shared memories.

TGM Mini Memory

– existing shared memories created by **Motion** – **Control Observer** is in online mode.
Compatibility ID – code specifying the compatibility with TG Motion.
Compatibility Memory ID – code that determines the compatibility with shared memories.
Kernel – TG Motion version.

REMOTE TGM Interface (DLL)

Remote TG Motion – interconnection with TG Motion running on another computer, via a network.

Version – version of Control Observer (comprising the version number and the distribution date).
Compatibility ID – code specifying the compatibility with TG Motion.
Compatibility Memory ID – code that determines the compatibility with shared memories.

Client/Offline Mem.

– address of a client or a shared memory for **Control Observer** in offline mode.
Compatibility Memory ID – code that determines the compatibility with shared memories

Local TG Motion Memory

– existing shared memories created by **Motion** – **Control Observer** is in online mode.
Compatibility ID – code specifying the compatibility with TG Motion.
Compatibility Memory ID – code that determines the compatibility with shared memories.

Connection Status

– network communication status.

Remote TG Motion Memory

Compatibility ID – code specifying the compatibility with TG Motion.
Compatibility Memory ID – code that determines the compatibility with shared memories.
Kernel – TG Motion version.
MathDll – transformation mathematical library (not available in version 903).

EXIT

Control Observer menu item which is intended to close it.

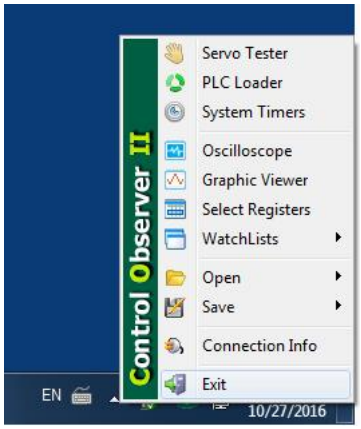


Fig. Exit function of Control Observer menu

When **Control Observer** is closed, all **WatchLists** including their settings will be deleted. Setting data of **Graphic Viewer** will be deleted likewise. Closing the **Control Observer** will not influence the operation of **TG Motion** in any way.

*Save/Project command of **Control Observer** menu may be used to preserve the settings of the **Control Observer** utilities.*