SIEMENS

SIMATIC

Industrial software

Readme SIMATIC S7-PLCSIM Advanced V2.0 SP1

Readme

General information

Content

This Readme file contains information about SIMATIC S7-PLCSIM Advanced V2.0 SP1. The information should be considered more up-to-date than the product documentation and, where necessary, replaces corresponding information in the product documentation and installation instructions.

Compatibility

Compatibility of API and Runtime versions

PLCSIM Advanced V2.0 SP1 contains the Runtime version V2.0 SP1 and the API versions V2.0 (SP1) and V1.0 (SP1).

The installation of PLCSIM Advanced V2.0 SP1 leads to an upgrade of an existing earlier version. The Runtime Manager of PLCSIM Advanced V2.0 SP1 is compatible with projects that were created with earlier API versions. You can therefore continue to use already created projects.

Note

An API with a higher version number (for example V2.0) cannot connect with an earlier Runtime version (for example V1.0).

Compatibility to TIA Portal and to CPU firmware versions

The firmware used in PLCSIM Advanced V2.0 SP1 corresponds to that of a CPU S7-15xx V2.6.

The following versions are compatible:

Table 1 Compatibility to TIA Portal and to CPU firmware versions

PLCSIM Advanced	TIA Portal	Supported CPU firmware version
V1.0 SP1	V14 to V15.1	V1.8, V2.0
V2.0	V14 to V15.1	V1.8 to V2.5
V2.0 SP1	V14 to V15.1	V1.8 to V2.6

System requirements

You should preferably install PLCSIM Advanced on a SIMATIC Field PG M5 Advanced or comparable PC.

For PLCSIM Advanced to operate efficiently, the following minimum requirements for computer hardware or for a virtual machine must be met.

Table 2 System requirements

	Hardware	Virtual machine
Processor	• One logical Intel Core™ i7-6820EQ core per started instance • At least one additional core for the operating system • At least one additional core for the additional active applications	• One virtual CPU per started instance has to be assigned to the VM • A corresponding number of processors has to be physically available on the host • At least one additional core for the operating system • At least one additional core for the additional active applications
RAM	• 1 GB per started instance • At least 4 GB for the Windows operating system • Additional RAM corresponding to the requirements of the remaining active applications	• 1 GB per started instance • At least 4 GB for the Windows operating system • Additional RAM corresponding to the requirements of the remaining active applications
Free hard disk space	5 GB	5 GB
Screen resolution	Minimum 1024 x 768	Minimum 1024 x 768

Operating systems (64-bit versions)

PLCSIM Advanced V2.0 SP1 supports the following operating systems:

- Windows 7 Home Premium SP1
- Windows 7 Professional SP1
- Windows 7 Enterprise SP1
- Windows 7 Ultimate SP1
- Windows 10 Home Version 1709 and 1803
- Windows 10 Pro Version 1709 and 1803
- Windows 10 Enterprise Version 1709 and 1803 (for SIMATIC Field PG M5)
- Windows 10 (IoT) Enterprise 2016 LTSB
- Windows Server 2012 R2 StdE (full installation)
- Windows Server 2016 Standard (full installation)

Note

Make sure that the Windows operating system you are using is up to date.

Licenses

Number of instances per license

Contrary to the specifications in the Function Manual "SIMATIC S7-PLCSIM Advanced V2.0", Edition 12/2017, Preface and Chapter "Licenses", the following applies:

A license is valid for two instances within a PLCSIM Advanced installation.

Floating license

PLCSIM Advanced is supplied with a floating type license. It can be stored locally and shared for a network.

Note

Validity

PLCSIM Advanced V2.0 SP1 cannot be used with a V1.0 license.

PLCSIM Advanced V1.0 can be used with a V2.0 license.

Handling of licenses is described in the Help for SIMATIC Automation License Manager (ALM).

Notes

Working with multiple instances

When you are working with instances **without unique IP addresses**, note the following procedure for downloading from the TIA Portal via "PLCSIM" (Softbus):

1. Start **only one** instance with the symbol  in the Control Panel.
2. In the TIA Portal, download the program to this instance.
3. Repeat the steps until you have created all instances and downloaded all projects.

Saving retentive data securely

To securely save retentive data on the virtual SIMATIC Memory Card, the virtual controller must be shut down correctly. Use one of the following functions for this:

- The `PowerOff()` API function
- In the Control Panel, the function "Switch off instance"  , "Log off instance"  or the Exit function "Log off all instances"

Safety system version V1.6 or V2.0 for fail-safe I/O

To successfully simulate and test a project with fail-safe input and output modules, you need to use safety system version V1.6 or V2.0 for the project. Simulation of the fail-safe input and output modules does not work correctly with an older version.

Password for module exchange

When you replace an existing S7-1500 CPU (**firmware < V2.0**) with an S7-1500 CPU (**firmware ≥ V2.0**), you are not informed in the TIA Portal that a more up-to-date algorithm for password encryption is available.

PLCSIM Advanced does not support any password encryption for CPU versions with firmware < V2.0.

In order to use protection levels, the Web server and the access protection of the F-CPU in the simulation, click on the "Update password encryption" button. The button is located in the CPU properties in the "Protection & Security" tab under "Access level".

Addition to section 4 Communication paths

Note

Local communication via TCP/IP

Make sure that communication is only local and cannot be downloaded to real hardware. To this purpose no other adapters of your Windows PC may be configured in the physical network and in the subnet protocol of the PLCSIM Virtual Ethernet adapter. Microsoft KB 175767 provides background.

HMI devices and CPU protection levels

- PLCSIM Advanced supports HMI devices as of version 14. Connections to HMI devices prior to V14 are not supported.
- PLCSIM Advanced supports protection levels if the virtual S7-1500 controller is configured with a firmware version V2.0 or higher.
- It is possible to connect HMI devices as of V14 to virtual S7-1500 controllers that are configured with a firmware version V2.0 or higher, with or without protection levels.
- It is possible to connect HMI devices as of V14 to virtual S7-1500 controllers which are configured with a firmware version lower than V2.0 without protection levels.

Remedy

To establish a connection to the HMI device V13 or earlier, you have to update this HMI device to version V14.

To establish a connection from the virtual controller that is configured with a firmware version lower than V2.0 to the HMI device, you have to remove existing protection levels from the project.

Priority for hardware interrupt OB

The hardware interrupts triggered via the PLCSIM Advanced API are transmitted in sequence to the user program.

The priority of the assigned hardware interrupt OB determines the sequence of execution only if events occur simultaneously.

Instance does not switch to RUN

The PLCSIM Advanced instance may not automatically switch to the RUN operating state after a download or after restoring a project.

Remedy

In this case, set the instance to the RUN operating state using the corresponding buttons in TIA Portal or PLCSIM Advanced Control Panel.

User interface

S7-PLCSIM Advanced Symbol

After installing PLCSIM Advanced, the following icons are on the Windows desktop:

Figure 1 PLCSIM Advanced Symbol

A double-click on the symbol opens the Control Panel for PLCSIM Advanced. If the Control Panel is in the background, it is moved to the foreground with another double-click.

You can use Windows functions to permanently display the icon in the system tray of the taskbar.

Opening a graphical interface

Right-clicking the icon in the taskbar opens the Control Panel with the quick view. Double-click to start the Control Panel as a window.

Figure 2 Opening a graphical interface

You can use the mouse-over function to display messages about the current status of the instances.

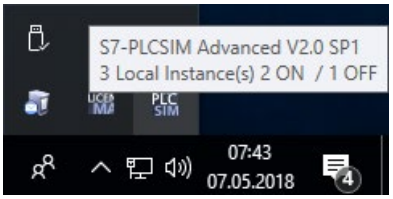


Figure 3 Example: Message in the taskbar

Graphical interface

The graphical interfaces synchronize by means of API commands. They are optional and are not needed to operate PLCSIM Advanced via the API.

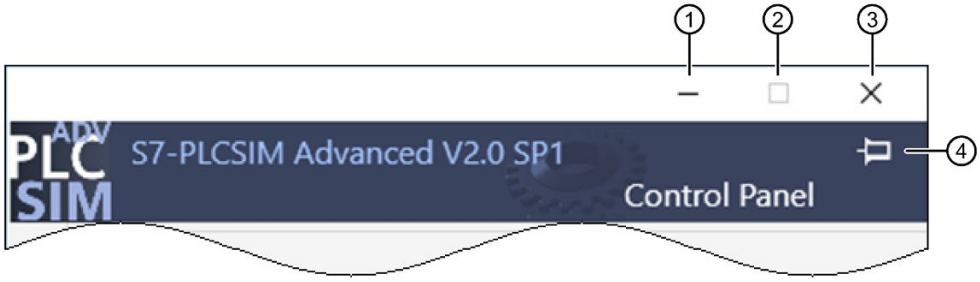
S7-PLCSIM Advanced V2.0 SP1 provides the Control Panel with two views.

- **Control Panel as quick view**
  Right-clicking on the icon in the taskbar opens the quick view.
  Clicking on an empty area on the desktop minimizes the quick view. The instances are not affected.
- **Control Panel as window**
  Double-clicking the icon on the desktop or in the taskbar opens the Control Panel as a window.

Control Panel as window

Unlike the quick view, you can operate the Control Panel with the buttons in the title bar.

You can close this window without exiting the simulation Runtime process.



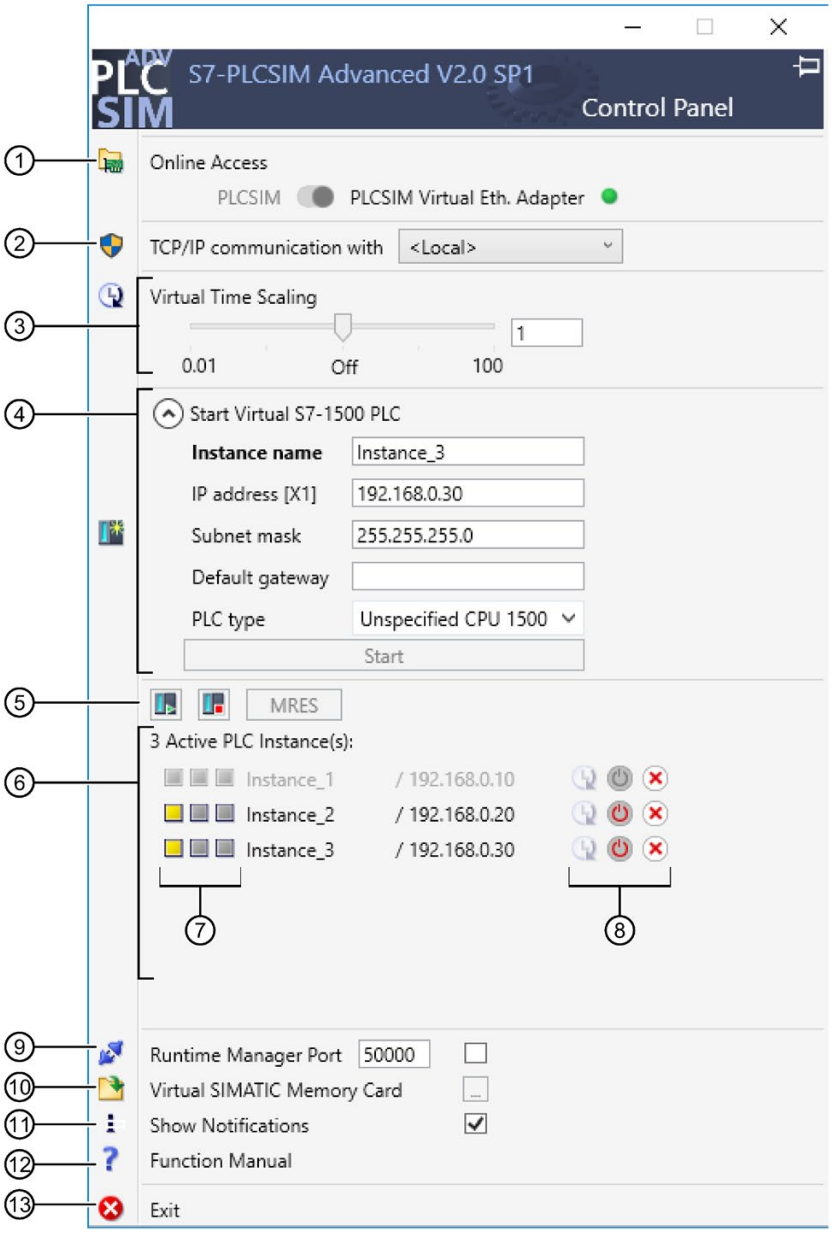
① Stores the Control Panel as icon in the taskbar.
② No function. The window size cannot be changed.
③ Closes the Control Panel and stores it in the system tray of the taskbar.
The instances and the simulation Runtime process remain active.
This function therefore differs from the Exit function .
The Exit function switches off the local instances, logs them off and closes the Control Panel.
④ Pins the Control Panel on the screen so that it remains in the foreground.

Figure 4 Control Panel: Title bar

S7-PLCSIM Advanced Control Panel

The Control Panel is available in English in version V2.0 SP1.

Design



①	Online access	Switch to select the communication interface
②	TCP/IP communication	Selection of network adapter for distributed communication
③	Virtual time	Slider to adjust the scaling factor

④	**Start Virtual S7-1500 PLC**	Opens and closes the text boxes for creating the instance (virtual controller).
	• Name of the instance	Here you enter a unique name for the instance. Enter a minimum of 3, a maximum of 64 characters. If the name is unique in the network, the button "Start" is enabled.
	• IP address • Subnet mask • Standard gateway	The input boxes are visible when you switch the communication interface to "PLCSIM Virtual Ethernet Adapter". The IP address is entered automatically.
	• CPU type	Here you select the type of CPU to be simulated.
	• "Start" button	Create with the button and and start the instance.
⑤	Buttons	Buttons for operating the selected instances.
⑥	Instance list	The list shows the available **local** instances. The instances can be resorted using the mouse cursor.
⑦	LED displays	The meaning of the LED is displayed when you move the mouse over it.
⑧	Icons	Icons for operating the instance
⑨	Runtime Manager Port	Here you open a port on the local PC.
⑩	Virtual SIMATIC Memory Card ...	Open an Explorer window here in which you select the path to the virtual memory card.
⑪	Display messages	Here you disable the PLCSIM Advanced messages in the Windows task bar for the duration of the operation.
⑫	Function manual	This is where you open the S7-PLCSIM Advanced Function Manual in a standard PDF viewer.
⑬	Exit	Exit logs off all instances and closes the Control Panel.

Figure 5 Control Panel V2.0 SP1

Switch for communication interface

Use the switch to select the communication interface for all instances to be created:

- "PLCSIM" corresponds to the local communication via softbus (default).
- "PLCSIM Virtual Ethernet Adapter" corresponds to the communication via TCP/IP.

The setting applies to all other instances. The selected communication interface for starting an instance is maintained until all instances are shut down.

When an instance is already started, it sets "its" communication interface as the default for other instances.

To change the communication interface, switch off all instances and enable the other interface.

TCP/IP communication

You can select a real network adapter from the drop-down list during operation. You thus activate the PLCSIM Virtual Switch and establish TCP/IP communication between the instances and the real network.

The <Local> setting disables the PLCSIM Virtual Switch and disconnects the instances from the real network. Only local TCP/IP communication over virtual adapter is possible in this case.

Virtual time

Use the slider or the mouse wheel to select the scaling factor for the virtual time.

The selected scaling factor applies to the instances for which the virtual time is enabled.

Clicking on "Off" restores the default (1) again.

Creating an instance (locally) and starting it

To create an instance, enter a unique name under "Instance Name". If the name already exists in the directory of the Virtual SIMATIC Memory Card, the existing instance is started.

In the "PLC-Type" drop-down list, select the Unspecified CPU 1500 or Unspecified ET 200SP CPU type. Create the instance with the "Start" button and start this instance.

The instance / virtual controller is named with the first download from the TIA Portal.

Instance list

The list contains the instances that are available locally on the PC or virtualization platform. Instances that have already been started on the runtime API are detected and displayed in the list.

Select the operating mode of the instance with the "RUN" and "STOP" buttons. Select one or more instances for this purpose. Perform a memory reset with the "MRES" button.

The LED displays show the status of the instance that corresponds to those of the hardware CPU.
RUN and STOP are displayed depending on the current operating state of the instance.

You can "operate" the instance with icons:

 Apply scaling factor for the virtual time,  disable virtual time,

 Switch on instance ("`PowerOn`"),  Switch off instance ("`PowerOff`"),

 Switch off instance and log off from Runtime Manager ("`Unregister`")

Runtime Manager Port

A remote connection can be established to another Runtime Manager via the specified port. The value must be greater than 1024.

If you select the check box, the port remains stored. You can use the remote connection without having to make this setting every time you start the Control Panel. To use this functionality, the Control Panel must be started and running in the background.

Virtual SIMATIC Memory Card

The user program, the hardware configuration and the retentive data are stored on the Virtual SIMATIC Memory Card. Use the button to open an Explorer window in which you select the path to the virtual memory card or in which the path is displayed.

Display messages

Each time the panel starts, help information and messages relating to the Control Panel are displayed, for example, when changing the IP address or when a license is missing. Disable the display if you do not need the messages.

Exit - Log off all instances

- The command switches off all local instances on the PC or the VM and logs them off from the Runtime Manager and closes the Control Panel.
- This command closes the Runtime Manager if there are no remote connections to other Runtime Managers.
- If the Runtime Manager has remote connections to instances on additional PCs, these instances and the Runtime Manager continue to run.

Importing instances

Requirement

This function is only available if you do not start the Control Panel with admin rights.

Importing instances

You can use the drag-and-drop function to import instances from a folder directly into the instance list of the Control Panel.

1. Open a folder with instances, for example, using the "Virtual SIMATIC Memory Card" button.
2. Select one or more instances and drag them into the highlighted area.

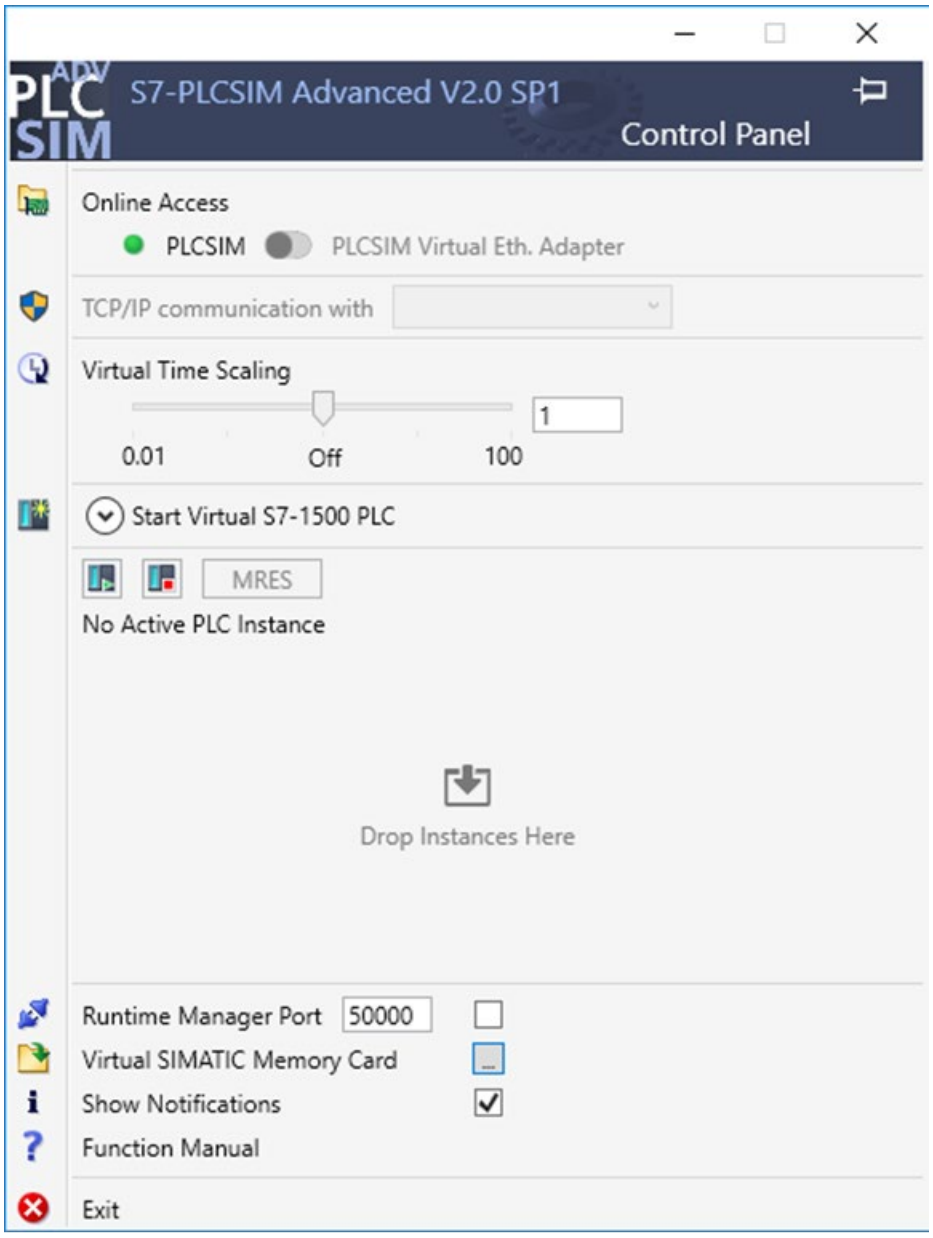


Figure 6 Control Panel: Importing instances

Messages and remedy

The settings for TCP/IP communication are checked in the S7-PLCSIM Advanced Control Panel.
The messages and the corresponding remedies are listed below:

Message

"Siemens PLCSIM Virtual Ethernet Adapter was not found. Please reinstall PLCSIM Advanced."

Remedy

The PLCSIM Virtual Ethernet Adapter cannot be found on the system.

Run PLCSIM Advanced Setup again:

1. Double-click the download package or insert the installation medium into the drive. The setup program starts up automatically, provided you have not disabled the Autostart function on the computer. If the setup program does not start up automatically, start it manually by double-clicking the "Start.exe" file.
2. Follow the prompts until you reach the "Configuration" window. Select the "Repair" check box.
3. Follow the remaining prompts to repair your installation.
4. Complete the repair operation by restarting your computer.

Message

"Siemens PLCSIM Virtual Ethernet Adapter is disabled. Please enable it."

Remedy

The PLCSIM Virtual Ethernet Adapter is deactivated on the system. In the Control Panel, under "Network and Sharing Center" > "Change Adapter Settings" and activate the network adapter.

Message

"NetGroup Packet Filter Driver (NPF) is not running. Start it from cmd with 'net start npf'."

Remedy

The NetGroup Packet Filter Driver (NPF) is not active on the system. Open a command line in administrator mode and enter the command "net start npf".

Message

"You have to set a valid IP address for the Siemens PLCSIM Virtual Ethernet Adapter."

Remedy

Assign a static IP address to the Siemens PLCSIM Virtual Ethernet Adapter or obtain an IP address via DHCP (default setting).

Addition to Section 6

Interruption of the process

Since PLCSIM Advanced runs in a Windows environment, Windows may temporarily suspend the virtual controller process. In such a case, both the virtual and the real clock stop in the virtual controller. They only continue to run when Windows resumes processing.

Change to section 6.2 Stop simulation

Note

Freeze state retained during download

The freeze state is no longer canceled during a download. To complete the download, the virtual controller must, however, pass a cycle control point at the end of the download.

User interfaces (API)

Additions to section 7 User interfaces (API)

Transfer parameters for API functions

All API functions that return a value using the function parameters expect a user-allocated memory area as a transfer parameter. Null pointers are not permitted. Exceptions to this are the functions that return an interface of a virtual controller:

- An ISimulationRuntimeManager interface
- An IRemoteRuntimeManager interface
- An IInstance interface

API methods

The following return values and exceptions have been added for the API methods `AlarmNotification`(), `ProcessEvent`(), `StatusEvent`(), `ProfileEvent`(), `UpdateEvent`():

Table 3 AlarmNotification(), ProcessEvent(), StatusEvent(), ProfileEvent(), UpdateEvent() - Native C++

Return value	Runtime error code	Condition
	`SREC_WRONG_MODULE_STATE`	The module is currently unplugged.

Table 4 AlarmNotification(), ProcessEvent(), StatusEvent(), ProfileEvent(), UpdateEvent() - .NET (C#)

Exception	Runtime error code	Condition
	`ERuntimeErrorCode.WrongModuleState`	The module is currently unplugged.

Addition to Section 7.6.4 Tag list

Elements with data types not known to the API (`EDataType.Unknown`) are not included in the tag list.

Addition to section 7.6.5 I/O access

7.6.5.4 I/O access via tag name - Reading

The following return values and exceptions have been added:

Table 5 Read(), Read...() - Native C++

Return value	Runtime error code	Condition
	SREC_INDEX_OUT_OF_RANGE	The offset lies outside the area range. No value could be read.

Table 6 Read(), Read...() - .NET (#)

Exception	Runtime error code	Condition
	ERuntimeErrorCode.IndexOutOfRange	The offset lies outside the area range. No value could be read.

7.6.5.5 I/O access via tag name - Writing

The following return values and exceptions have been added:

Table 7 Write(), Write...() - Native C++

Return value	Runtime error code	Condition
	SREC_INDEX_OUT_OF_RANGE	The offset lies outside the area range. No value could be written.

Table 8 Write(), Write...() - .NET (#)

Exception	Runtime error code	Condition
	ERuntimeErrorCode.IndexOutOfRange	The offset lies outside the area range. No value could be written.

Addition to Section 7.6.7 Cycle control

SetCycleTimeMonitoringMode()

With this function the source of the timer for the maximum cycle time monitoring can be changed.

Table 9 SetCycleTimeMonitoringMode() - Native C++

<table>
<tr><td>Syntax</td><td colspan="2">
<pre>ERuntimeErrorCode SetCycleTimeMonitoringMode(
 ECycleTimeMonitoringMode in_CycleTimeMonitoringMode
)
ERuntimeErrorCode SetCycleTimeMonitoringMode(
 ECycleTimeMonitoringMode in_CycleTimeMonitoringMode,
 INT64 in_MaxCycleTime_ns
)</pre>
</td></tr>
<tr><td>Parameter</td><td colspan="2">
• ECycleTimeMonitoringMode in_CycleTimeMonitoringMode:
Select one of the following options for the maximum cycle time monitoring:
– SRCTMM_DOWNLOADED:
The maximum cycle time from the project that was downloaded from STEP 7 is used as maximum cycle time monitoring.
– SRCTMM_IGNORED (default):
A timer value of one minute is used as maximum cycle time monitoring to prevent a potential error in case of an overflow of cyclic events.
– SRCTMM_SPECIFIED:
A value that is specified with the in_MaxCycleTime_ns parameter is used as maximum cycle time monitoring.
The default setting is 150 ms.
• INT64 in_MaxCycleTime_ns:
The user-specific value for the maximum cycle time monitoring.
A value between 1000000 and
60000000000 ns (1 millisecond to 1 minute) is valid. If no value is specified in the API, the default value of 150 ms applies.
</td></tr>
<tr><td rowspan="6">Return values</td><td>Runtime error code</td><td>Condition</td></tr>
<tr><td>SREC_OK</td><td>The function is successful.</td></tr>
<tr><td>SREC_INTERFACE_REMOVED</td><td>The instance is not registered in Runtime Manager.</td></tr>
<tr><td>SREC_TIMEOUT</td><td>The function does not return on time.</td></tr>
<tr><td>SREC_WRONG_ARGUMENT</td><td>The cycle time monitoring mode is invalid.</td></tr>
<tr><td>SREC_INDEX_OUT_OF_RANGE</td><td>The user-specific value for the maximum cycle time monitoring is outside the limits.</td></tr>
</table>

Table 10 SetCycleTimeMonitoringMode() - .NET (C#)

Syntax	`void SetCycleTimeMonitoringMode(`
	` ECycleTimeMonitoringMode in_CycleTimeMonitoringMode`
	`)`
	
	`void SetCycleTimeMonitoringMode(`
	` ECycleTimeMonitoringMode in_CycleTimeMonitoringMode,`
	` Int64 in_MaxCycleTime_ns`
	`)`	

Parameter	• `ECycleTimeMonitoringMode in_CycleTimeMonitoringMode:` Select one of the following options for the maximum cycle time monitoring: – `ECycleTimeMonitoringMode.Downloaded:` The maximum cycle time from the project that was downloaded from STEP 7 is used as maximum cycle time monitoring. – `ECycleTimeMonitoringMode.Ignored` (default): A timer value of one minute is used as maximum cycle time monitoring to prevent a potential error in case of an overflow of cyclic events. – `ECycleTimeMonitoringMode.Specified:` A value that is specified with the `in_MaxCycleTime_ns` parameter is used as maximum cycle time monitoring. The default setting is 150 ms. • `Int64 in_MaxCycleTime_ns:` The user-specific value for the maximum cycle time monitoring. A value between 1000000 and 60000000000 ns (1 millisecond to 1 minute) is valid. If no value is specified in the API, the default value of 150 ms applies.	
Return values	None	
Exceptions	`Siemens.Simatic.Simulation.Runtime.SimulationRuntimeException`	
	Runtime error code	Condition
	`ERuntimeErrorCode.InterfaceRemoved`	The instance is not registered in Runtime Manager.
	`ERuntimeErrorCode.Timeout`	The function does not return on time.
	`ERuntimeErrorCode.WrongArgument`	The cycle time monitoring mode is invalid.
	`ERuntimeErrorCode.IndexOutOfRange`	The user-specific value for the maximum cycle time monitoring is outside the limits.

GetCycleTimeMonitoringMode()

This function returns information on the source of the timer for the maximum cycle time monitoring.

Table 11 GetCycleTimeMonitoringMode() - Native C++

Syntax	`ERuntimeErrorCode GetCycleTimeMonitoringMode(`
	` ECycleTimeMonitoringMode* out_CycleTimeMonitoringMode,`
	` INT64* out_MaxCycleTime_ns`
	`)`	

Parameter	• `ECycleTimeMonitoringMode* out_CycleTimeMonitoringMode:` The configured mode for cycle time monitoring. The default setting is `SRCTM_IGNORED`. • `INT64 in_MaxCycleTime_ns:` The user-specific value for the maximum cycle time monitoring. If no value is specified in the API, the default value of 150 ms is returned.	
Return values	Runtime error code	Condition
	`SREC_OK`	The function is successful.
	`SREC_INTERFACE_REMOVED`	The instance is not registered in Runtime Manager.
	`SREC_TIMEOUT`	The function does not return on time.

Table 12 GetCycleTimeMonitoringMode() - .NET (C#)

Syntax	`void GetCycleTimeMonitoringMode(`
	` out ECycleTimeMonitoringMode out_CycleTimeMonitoringMode,`
	` out Int64 out_MaxCycleTime_ns`
	`)`	

Parameter	• `ECycleTimeMonitoringMode out_CycleTimeMonitoringMode:` The configured mode for cycle time monitoring. The default setting is `ECycleTimeMonitoring-Mode.Ignored`. • `Int64 in_MaxCycleTime_ns:` The user-specific value for the maximum cycle time monitoring. If no value is specified in the API, the default value of 150 ms is returned.	
Return values	None	
Exceptions	`Siemens.Simatic.Simulation.Runtime.SimulationRuntimeException`	
	Runtime error code	Condition
	`ERuntimeErrorCode.InterfaceRemoved`	The instance is not registered in Runtime Manager.
	`ERuntimeErrorCode.Timeout`	The function does not return on time.

Addition to Section 7.6.8.2 ReadRecordDone

ReadRecordDone()

The following return values and exceptions have been added:

Table 13 ReadRecordDone() - Native C++

Return value	Runtime error code	Condition
	`SREC_INDEX_OUT_OF_RANGE`	The byte array of the read data record exceeds the length `DDATARECORD_MAX_SIZE` = 64000.

Table 14 ReadRecordDone() - .NET (C#)

Exception	Runtime error code	Condition
	`ERuntimeErrorCode.IndexOutOfRange`	The byte array of the read data record exceeds the length `DDATARECORD_MAX_SIZE` = 64000.

Addition to section 7.6.8.3 AlarmNotification

AlarmNotification

The `in_ModuleState` parameter is derived from the sum (ORed) of the severity level in the `SDiagExtChannelDescription` field.

If a diagnostic interrupt should be generated for both "Maintenance demanded" as well as "Maintenance required", select "6" as the module status.

Table 15 AlarmNotification() - Native C++

Parameter	• `UINT16 in_ModuleState` DMODULE_STATE_OK = 0 DMODULE_STATE_ERROR = 1 DMODULE_STATE_MAINT_DEMANDED = 2 DMODULE_STATE_MAINT_REQUIRED = 4	
Return values	Runtime error code	Condition

	`SREC_WRONG_ARGUMENT`
	instead of
	`SREC_INDEX_OUT_OF_RANGE`
		The field size of `in_ArrayOfDiagnosisEvents` is less than the number of the currently available diagnostic events.

Table 16 AlarmNotification() - .NET (C#)

Parameter	• `ushort in_ModuleState` ModuleState.Ok = 0 ModuleState.Error = 1 ModuleState.MaintenanceDemanded = 2 ModuleState.MaintenanceRequired = 4	
Return values	Runtime error code	Condition

	`ERuntimeErrorCode.WrongArgument`
	instead of
	`ERuntimeErrorCode.IndexOutOfRange`
		The field size of `in_ArrayOfDiagnosisEvents` is less than the number of the currently available diagnostic events.

Example	`ushort seqNumber;`
	`var In_ArrayOfDiagnosisEvent = new SDiagExtChannelDescription[] {`
	`new SDiagExtChannelDescription() {ChannelNumber = 0x8000, ErrorType = 0x0001, ExtErrorType = 0, Direction = EDiagProperty.Appear,Severity =EDiagSeverity.MaintDemanded},`
	`new SDiagExtChannelDescription() {ChannelNumber = 0x8000, ErrorType = 0x0002, ExtErrorType = 0, Direction = EDiagProperty.Appear,Severity =EDiagSeverity.Failure},`
	`new SDiagExtChannelDescription() {ChannelNumber = 0x8000, ErrorType = 0x0003, ExtErrorType = 0, Direction = EDiagProperty.Appear,Severity =EDiagSeverity.MaintRequired},`
	`Instance.AlarmNotification(269, 7, 3, In_ArrayOfDiagnosisEvent, out seqNumber);`
	`//ModuleState parameter is sum of the severities in the SDiagExtChannelDescription array above: 4+2+1`	

Addition to Section 7.6.10.1 OnDataRecordRead / OnDataRecordWrite events

RegisterOnDataRecordWriteCallback()

When the event occurs, the registered callback function is called. Only one callback function can be registered for the event. Registering a new callback function causes the previous callback function to be deleted.

Table 17 RegisterOnDataRecordWriteCallback() - Native C++

Syntax	`void RegisterOnDataRecordWriteCallback (`
	`  EventCallback_II_SREC_ST_SDRI_BYTE in_CallbackFunction`
	`);`

Parameter	• `EventCallback_II_SREC_ST_SDRI_BYTE in_CallbackFunction:` A callback function that subscribes to the event. See EventCallback_II_SREC_ST_SDRI_BYTE (Page 19).
Return values	None
Note	The callback function runs in a separate thread.

UnregisterOnDataRecordWriteCallback()

Unregisters the callback function. When the event occurs, no callback function is called.

Table 18 UnregisterOnDataRecordWriteCallback() - Native C++

Syntax	`void UnregisterOnDataRecordWriteCallback();`
Parameter	None
Return values	None

Additions to section 7.8 Data types

7.8.2.10 EventCallback_II_SREC_ST_SDRI_BYTE

The function replaces `EventCallback_II_SREC_ST_SDR`.

Description

Table 19 EventCallback_II_SREC_ST_SDRI_BYTE - Native C++

Syntax	`typedef void (*EventCallback_II_SREC_ST_SDRI_BYTE)(`
	`IInstance* in_Sender,`
	`ERuntimeErrorCode in_ErrorCode,`
	`SYSTEMTIME in_SystemTime,`
	`SDataRecordInfo in_DataRecordInfo`
	`const BYTE* in_Data`
	`);`

Parameters	• `IInstance* in_Sender:` An interface of the instance that receives this event. • `ERuntimeErrorCode in_ErrorCode:` A possible error code. • `SYSTEMTIME in_SystemTime:` The virtual system time of the virtual controller at the time when this event was triggered. • `SDataRecordInfo in_DataRecordInfo:` The structure `SDataRecordInfo` contains the following information: – The HW identifier to which the CPU wants to write the data record – The index of the supplied data record – Size of data record • `const BYTE* in_Data:` The data record. This pointer becomes invalid after the callback function has returned.
Return values	None

7.8.6.15 SDataRecord

This structure is omitted for Native C++.

Description

This structure contains read/write data record information and data records.

Table 20 SDataRecord - .NET (C#)

Syntax	`struct SDataRecord`
	`{`
	`UInt32 HardwareId;`
	`byte[] Data`
	`}`

Member	• `SDataRecordInfo Info:` The data record information, see SDataRecordInfo • `byte[] Data:` The array length

7.8.7.9 EPrimitiveDataType

Compatible primitive data types

The following table shows the primitive data types of the user interface (API) and the data types of the PLCSIM Advanced instance that are configured in the stored tag list. The data types that can be used as compatible are marked with "X".

Table 21 Compatible primitive data types - Write

API	PLCSIM Advanced instance											
	Bool	INT				UINT				Float	Dou-ble	Char	WChar
		8	16	32	64	8	16	32	64				
Bool	X												
INT8		X	X	X	X								
INT16			X	X	X								
INT32				X	X								
INT64					X								
UINT8			X	X	X	X	X	X	X				
UINT16				X	X		X	X	X				
UINT32					X			X	X				
UINT64									X				
Float										X			
Double											X		
Char												X	
WChar													X

7.8.7.20 ECycleTimeMonitoringMode

Description

This enumeration contains the sources of the timer for the maximum cycle time monitoring.

Table 22 ECycleTimeMonitoringMode - Native C++

Syntax	`enum ECycleTimeMonitoringMode`
	`{`
	`  SRCTMM_DOWNLOADED = 0,`
	`  SRCTMM_IGNORED = 1,`
	`  SRCTMM_SPECIFIED = 2`
	`};`

Table 23 ECycleTimeMonitoringMode - .NET (C#)

Syntax	`enum ECycleTimeMonitoringMode`
	`{`
	`  Downloaded = 0,`
	`  Ignored = 1,`
	`  Specified = 2`
	`}`

Siemens AG
Division Digital Factory
Postfach 48 48
90026 NÜRNBERG
GERMANY