

SIEMENS

efesotomasyon.com

SIMATIC

S7ProSim V5.4

COM Object

User Manual

Edition: 01/2007

A5E00992430-01

Copyright and Safety Guidelines

This manual contains notices which you should observe to ensure your own personal safety, as well as to protect the product and connected equipment. These notices are highlighted in the manual by a warning triangle and are marked as follows according to the level of danger:

**Danger**

Indicates an imminently hazardous situation that, if not avoided, will result in death or serious injury.

**Warning**

Indicates a potentially hazardous situation that, if not avoided, could result in death or severe injury.

**Caution**

Used with the safety alert symbol indicates a potentially hazardous situation that, if not avoided, may result in minor or moderate injury.

Caution

Used without the safety alert symbol indicates a potentially hazardous situation that, if not avoided, may result in property damage.

Notice

Used without the safety alert symbol indicates a potential situation that, if not avoided, may result in an undesirable result or state.

Qualified Personnel

The device/system may only be set up and operated in conjunction with this manual. Only qualified personnel should be allowed to install and work on this equipment. Qualified persons are defined as persons who are authorized to commission, to ground, and to tag circuits, equipment, and systems in accordance with established safety practices and standards.

Correct Usage

Note the following:

**Warning**

This device and its components may only be used for the applications described in the catalog or the technical descriptions and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.

This product can only function correctly and safely if it is transported, stored, set up, and installed correctly, and operated and maintained as recommended.

Trademarks

Siemens® and SIMATIC® are registered trademarks of SIEMENS AG.

STEP 7™ and S7™ are trademarks of SIEMENS AG.

Microsoft®, Windows®, Windows 95®, Windows 98®, Windows NT®, Windows ME®, and Windows 2000® are registered trademarks of Microsoft Corporation.

Copyright Siemens AG, 2007

All rights reserved

The reproduction, transmission or use of this document or its contents is not permitted without express written authority. Offenders will be liable for damages. All rights, including rights created by patent grant or registration of a utility model or design, are reserved.

Disclaimer of Liability

We have checked the contents of this manual for agreement with the hardware and software described. Because deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual are reviewed regularly and any necessary corrections included in subsequent editions. Suggestions for improvement are welcomed.

Preface

S7ProSim provides programmatic access to the simulated PLC interface of S7-PLCSIM. With S7ProSim, you can write software to perform such tasks as changing the keyswitch position of the simulated PLC, stepping through the control program a scan at a time, reading or writing controller values, and many other tasks.

Audience

This manual is intended for engineers, programmers, and maintenance personnel who have knowledge and experience with S7 programmable logic controllers, and with developing software in Visual Basic (6.0 or .NET), or Visual C++ (6.0 or .NET).

Scope

This document describes the features and the operation of S7ProSim V5.4.

Other Manuals

You can find additional information in the online help for STEP 7 and S7-PLCSIM, and in the following manuals:

- *Programming with STEP 7 Manual.* This manual provides basic information on designing and programming control programs. Use this manual when creating a control program with the STEP 7 automation software.
- *System Software for S7-300/400 System and Standard Functions Reference Manual.* This manual provides you with descriptions of the system functions, organization blocks, and standard functions that you use when developing a control program.
- *Working with STEP 7 Getting Started Manual.* This manual explains how to use the STEP 7 automation software. This manual provides you with an overview of the procedures used to configure a PLC and to develop control programs.
- *S7-PLCSIM - Testing Your S7-CPU Program.* This manual explains the user interface and operation of S7-PLCSIM, the S7 PLC simulator.

To find these and other manuals, select the **Start > Simatic > Documentation** menu command from the Start menu of the computer where STEP 7 is installed.

Additional Assistance

For assistance in answering technical questions, for training on this product, or for ordering, contact your Siemens distributor or sales office.

North America and South America

Telephone: +1 (800) 333-7421

Fax: +1 (423) 262-2200

simatic.hotline@siemens.com

Europe and Africa

Telephone: +49 (0) 180 5050 222

Fax: +49 (0) 180 5050 223

adsupport@siemens.com

Asia and Pacific region

Telephone: +86 10 64 75 75 75

Fax: +86 10 64 74 74 74

adsupport.asia@siemens.com

Contents

S7ProSim Overview	7
Adding an S7ProSim COM Object to your Project	7
Programming an Interface to S7-PLCSIM with S7ProSim.....	8
Methods	11
BeginScanNotify	13
Connect	14
Continue	15
Disconnect.....	16
EndScanNotify	17
ExecuteNmsScan	18
ExecuteNScans	19
ExecuteSingleScan.....	20
GetPauseState	21
GetScanMode.....	22
GetStartUpSwitch	23
GetState.....	24
HotStartWithSavedValues	25
Pause.....	26
ReadDataBlockValue.....	27
ReadFlagValue	28
ReadOutputImage	29
ReadOutputPoint	30
SavePLC.....	32
SetScanMode	33
SetStartUpSwitch.....	34
SetState	35
StartPLCSim	36
WriteDataBlockValue.....	37
WriteFlagValue	38
WriteInputImage	39
WriteInputPoint	40
Events	41
ConnectionError.....	42
PauseStateChanged.....	43
PLCSimStateChanged.....	44
ScanFinished	45
ScanModeChanged	46

Type Definitions	47
CPURunMode.....	48
ImageDataTypeConstants	49
PauseStateConstants	50
PointDataTypeConstants.....	51
RestartSwitchPosition.....	52
ScanModeConstants	53
tagPauseState	54
ScanInfo Constants	55
Error return codes.....	59
Index.....	61

S7ProSim Overview

S7ProSim provides a COM object that provides programmatic access to the process simulation interface of S7-PLCSIM. You can use S7ProSim in any application that can accept COM objects to attach to an S7-PLCSIM process simulation.

This online document describes how to add S7ProSim to an application as well as the features, interface, and operations of S7ProSim, including software object definitions of the methods and events.

Adding an S7ProSim COM Object to your Project

To use an S7ProSim COM object in your project, you add a reference to it. The steps to add a project reference depend on your programming environment. In Microsoft Visual Basic (6.0 or .NET), for example, follow these steps to add an S7ProSim COM object reference:

1. Select the **Project > References** or **Project > Add Reference** menu command.
2. From the References dialog, select the checkbox for the Siemens S7ProSim COM Object. (For Visual Basic .NET, this selection is on the COM tab of the References dialog.)
3. Click OK.

After you add the project reference you can use the Object Browser to examine the methods and events of the S7ProSim COM object. From the Object Browser, select S7PROSIMLib from the drop-down list of libraries. The class S7ProSim contains the methods and events that you can use for programming an interface to S7-PLCSIM.

In Microsoft Visual Studio C++ V6.0 or in Microsoft Visual C++ .NET, follow the procedures to add a COM object that are relevant for that programming environment.

efesotomasyon.com

Programming an Interface to S7-PLCSIM with S7ProSim

To use S7ProSim to programmatically operate the S7-PLCSIM simulated controller, you must perform these tasks:

- Include the Siemens S7ProSim COM Object in the project.
- Add a declaration to your project for S7ProSim.

Example: Visual Basic 6.0

```
Option Explicit
Private WithEvents S7ProSim As S7PROSIMLib.S7ProSim
...
Private Sub Form_Load()
Set S7ProSim = New S7PROSIMLIB.S7ProSim
...
End Sub
```

Example: Visual Basic .NET

```
Private WithEvents S7ProSim As New S7PROSIMLib.S7ProSim
```

Example: Visual C++ 6.0

```
// the ProSim library/tlb is in the dll

#import <S7wspsmx.dll> named_guids, no_namespace//, raw_interfaces_only

class ProSimWrapper
{
public:
    ProSimWrapper() : m_pProSim(OLESTR("S7wspsmx.S7ProSim"), NULL,
CLSCTX_INPROC_SERVER)
    {}; // the smartptr is automatically created on the stack when the app
starts

    virtual ~ProSimWrapper()
    {}; // no implementation, the smartptr is automatically released when the
app shutdowns

    IS7ProSim * GetPtr()
    {
        return m_pProSim;
    };

// Attributes
protected:
    // IProSimPtr is a CComPtr (smart ptr) to the IProSim interface
    // It is from the dll file from the #import
    // CoCreateInstance will be called automatically on the ptr object in the
constructor of this class
    // release ptr is automatically called by the destructor of this class
    IS7ProSimPtr m_pProSim;
};
```

Example: C#

```
using S7PROSIMLib;
...
private S7ProSim ps;
```

- For Visual Basic, program event handlers for the S7ProSim events. Event handlers are not necessary in Visual C++. Within each event handler, you can insert any custom code for your application.

Example: Visual Basic 6.0

```
Private Sub S7ProSim_PauseStateChanged(ByVal NewState As String)
    DoEvents
    ...
End Sub

Private Sub S7ProSim_ScanFinished(ByVal ScanInfo As Variant)
    DoEvents
    ...
End Sub

Private Sub S7ProSim_PLCSimStateChanged(ByVal NewState As String)
    DoEvents
    ...
End Sub

Private Sub S7ProSim_ConnectionError(ByVal ControlEngine As String, ByVal error
As Long)
    DoEvents
    MsgBox "Connection Error"
End Sub

Private Sub S7ProSim_ScanModeChanged(ByVal NewState As String)
    DoEvents
    ...
End Sub
```

Note

In Visual Basic .NET, the "DoEvents" call is not necessary.

- Add command buttons, textboxes or other objects to your application as needed to access the various S7ProSim methods. Program the code for each command button handler to call S7ProSim methods and set corresponding values for textboxes as appropriate for your application.

Methods

◆ BeginScanNotify	Registers S7ProSim for callbacks from the controller. The ScanFinished event and PLCSimStateChanged event will be sent when these events occur.
◆ Connect	Connects S7ProSim to S7-PLCSIM.
◆ Continue	Continues a simulation that has been paused.
◆ Disconnect	Disconnects S7ProSim from S7-PLCSIM.
◆ EndScanNotify	Unregisters S7ProSim for callbacks from the controller. The ScanFinished event and PLCSimStateChanged event will not be sent.
◆ ExecuteNmsScan	Forces S7-PLCSIM to execute scan cycles for a specified time duration (Nms) and does not wait for the execution of the current scan to finish. If scan notification is enabled, the program will be notified when S7-PLCSIM has finished the scans.
◆ ExecuteNScans	Forces S7-PLCSIM to execute a specified number of scan cycles and does not wait for the execution of the current scan to finish. If scan notification is enabled, the program will be notified when S7-PLCSIM has finished the scans.
◆ ExecuteSingleScan	Forces S7-PLCSIM to execute one scan cycle and does not wait for the execution of the current scan to finish. If scan notification is enabled, the program will be notified when S7-PLCSIM has finished the scan.
◆ GetPauseState	Returns the current pause state of S7-PLCSIM.
◆ GetScanMode	Returns the scan mode of S7-PLCSIM.
◆ GetStartupSwitch	Gets the startup setting (Hot, Warm, or Cold Start) for S7-PLCSIM.
◆ GetState	Returns a string containing the current keyswitch position of S7-PLCSIM (RUN, RUN-P, or STOP).
◆ HotStartWithSavedValues	Sets a boolean to determine whether S7-PLCSIM should load saved peripheral I/O when started in the HotStart state. In order for S7-PLCSIM to start up and load peripheral I/O, the user must call HotStartWithSavedValues with a value of TRUE, save the PLC program (SavePLC), and set the startup state for S7-PLCSIM to HotStart (SetStartUpSwitch). When S7-PLCSIM restarts, it will then load the peripheral I/O.
◆ Pause	Pauses a simulation.
◆ ReadDataBlockValue	Reads a particular bit, byte, word, or double word from the DB memory area of S7-PLCSIM.
◆ ReadFlagValue	Reads a particular bit, byte, word, or double word from the M flag memory area of S7-PLCSIM.
◆ ReadOutputImage	Reads elements from the peripheral output image (PQ memory area) of S7-PLCSIM.

- ◆ **SavePLC** Saves the current simulated PLC data to a file.
The data that is saved consists of the program, the hardware configuration, the keyswitch position as indicated by the CPU view object, the type of scan (continuous or single scan), the I/O status, timer values, symbolic addresses, and the power setting (on or off).
- ◆ **SetScanMode** Sets the scan mode for S7-PLCSIM.
- ◆ **SetStartUpSwitch** Sets the type of startup (Hot, Warm, or Cold) to use when S7-PLCSIM starts up.
- ◆ **SetState** Sets the current keyswitch position of S7-PLCSIM (RUN, RUN-P, or STOP).
- ◆ **StartPLCSim** Starts S7-PLCSIM with the specified PLC simulation file (saved from a previous call to SavePLC).
- ◆ **WriteDataBlockValue** Writes a particular bit, byte, word, or double word to the DB memory area of S7-PLCSIM.
- ◆ **WriteFlagValue** Writes a particular bit, byte, word, or double word to the M flag memory area of S7-PLCSIM.
- ◆ **WriteInputImage** Writes elements to the peripheral input image (PI memory area) of S7-PLCSIM, starting at the StartIndex of the data pointed to by pData.
- ◆ **WriteInputPoint** Writes either a particular bit (Boolean), byte (Byte), a two-byte word (Integer) or a four-byte word (Long) from the Data Variant to the peripheral input image (PI memory area).

BeginScanNotify

STDMETHOD(**CS7ProSim::BeginScanNotify**)()

Description

Registers S7ProSim for callbacks from the controller. The ScanFinished event and PLCSimStateChanged event will be sent when these events occur.

Parameters

None

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off

Visual Basic Usage

Function **BeginScanNotify**() As Long

 **Connect**

```
STDMETHOD(CS7ProSim::Connect)()
```

 **Description**

Connects S7ProSim to S7-PLCSIM.

 **Parameters**

None

 **Error Handling**

Errors are returned in the ConnectionError event, not by the function call.

 **Return Value**

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off

 **Visual Basic Usage**

Function **Connect()** As Long

 **Continue**

```
STDMETHOD(CS7ProSim::Continue)()
```

 **Description**

Continues a simulation that has been paused.

 **Parameters**

None

 **Error Handling**

Errors are returned in the ConnectionError event, not by the function call.

 **Return Value**

Value	Meaning
S_OK	0x00000000 : Success code
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

 **Visual Basic Usage**

```
Sub Continue()
```

 **Disconnect**

```
STDMETHOD(CS7ProSim::Disconnect)()
```

 **Description**

Disconnects S7ProSim from S7-PLCSIM.

 **Parameters**

None

 **Error Handling**

Errors are returned in the ConnectionError event, not by the function call.

 **Return Value**

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off

 **Visual Basic Usage**

Function **Disconnect()** As Long

EndScanNotify

STDMETHOD (CS7ProSim::EndScanNotify) ()

Description

Unregisters S7ProSim for callbacks from the controller. The ScanFinished event and PLCSimStateChanged event will not be sent.

Parameters

None

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off
PS_E_NOTREGISTERED	0x80040209 : S7ProSim is not registered for callbacks from S7-PLCSIM

Visual Basic Usage

Function **EndScanNotify()** As Long


ExecuteNmsScan

STDMETHOD(**CS7ProSim::ExecuteNmsScan**)(long MsNumber)

Description

Forces S7-PLCSIM to execute scan cycles for a specified time duration (Nms) and does not wait for the execution of the current scan to finish. If scan notification is enabled, the program will be notified when S7-PLCSIM has finished the scans. S7-PLCSIM must be in single scan mode to use this method.

Parameters

 MsNumber Time duration (in milliseconds) for which scan cycles are to be executed.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_NOTSINGLESCAN	0x8004020A : S7-PLCSIM is not in single scan mode
PS_E_PLCNOTRUNNING	0x8004020E : S7-PLCSIM is not running
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

Visual Basic Usage

Function **ExecuteNmsScan**(*MsNumber As Long*) As Long


ExecuteNScans

STDMETHOD(**CS7ProSim::ExecuteNScans**)(long NScanNumber)

Description

Forces S7-PLCSIM to execute a specified number of scan cycles and does not wait for the execution of the current scan to finish. If scan notification is enabled, the program will be notified when S7-PLCSIM has finished the scans. S7-PLCSIM must be in single scan mode to use this method.

Parameters

 NScanNumber Number of scan cycles to be executed

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_NOTSINGLESCAN	0x8004020A : S7-PLCSIM is not in single scan mode
PS_E_PLCNOTRUNNING	0x8004020E : S7-PLCSIM is not running
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

Visual Basic Usage

Function **ExecuteNScans**(*NScanNumber As Long*) As Long

 **ExecuteSingleScan**

```
STDMETHOD (CS7ProSim::ExecuteSingleScan) ()
```

 **Description**

Forces S7-PLCSIM to execute one scan cycle and does not wait for the execution of the current scan to finish. If scan notification is enabled, the program will be notified when S7-PLCSIM has finished the scan. S7-PLCSIM must be in single scan mode to use this method.

 **Parameters**

None

 **Error Handling**

Errors are returned in the ConnectionError event, not by the function call.

 **Return Value**

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_PLCSIMNOTRUNNING	0x8004020E : S7-PLCSIM is not running
PS_E_NOTSINGLESCAN	0x8004020A : S7-PLCSIM is not in single scan mode
PS_E_MODALNOTPOSSIBLE	0x8004020C : S7-PLCSIM could not set specified scan mode

 **Visual Basic Usage**

Function `ExecuteSingleScan()` As Long


GetPauseState

STDMETHOD(**CS7ProSim::GetPauseState**)(PauseStateConstants *pVal)

Description

Returns the current pause state of S7-PLCSIM.

Parameters

 pVal Pointer to the returned S7-PLCSIM state, which is one of the PauseStateConstants

Notes

When called from Visual Basic, the pause state is returned in the function return value and there is no pVal parameter.

When called from C++, the state is returned in the value pointed to by pVal.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

Visual Basic Usage

Function **GetPauseState()** As [PauseStateConstants](#)


GetScanMode

STDMETHOD(**CS7ProSim::GetScanMode**)(ScanModeConstants **pVal*)

Description

Returns the scan mode of S7-PLCSIM.

Parameters

 *pVal* Pointer to the returned scan mode. The returned scan mode is one of the ScanModeConstants

Notes

When called from Visual Basic, the scan mode is returned in the function return value and there is no *pVal* parameter.

When called from C++, the state is returned in the value pointed to by *pVal*.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

Visual Basic Usage

Function **GetScanMode()** As [ScanModeConstants](#)


GetStartUpSwitch

STDMETHOD(**CS7ProSim::GetStartUpSwitch**)(RestartSwitchPosition *pPos)

Description

Gets the startup setting (Hot, Warm, or Cold Start) for S7-PLCSIM.

Parameters

 pPos pointer to S7-PLCSIM startup position value, which is one of the RestartSwitchPosition settings

Notes

When called from Visual Basic, the switch position is returned in the function return value and there is no pPos parameter.

When called from C++, the state is returned in the value pointed to by pPos.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

Visual Basic Usage

Function **GetStartUpSwitch()** As **RestartSwitchPosition**


GetState

STDMETHOD(**CS7ProSim::GetState**)(BSTR *pVal)

Description

Returns a string containing the current keyswitch position of S7-PLCSIM (RUN, RUN-P, or STOP).

Parameters

 pVal Pointer to the returned S7-PLCSIM keyswitch position value.

Notes

When called from Visual Basic, the state is returned in the function return value and there is no pVal parameter.

When called from C++, the state is returned in the value pointed to by pVal.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
E_INVALID_STATE	0x00008002 : Invalid state
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

Visual Basic Usage

Function **GetState()** As String

HotStartWithSavedValues


STDMETHOD(**CS7ProSim::HotStartWithSavedValues**)(BOOL val)

Description

Sets a boolean to determine whether S7-PLCSIM should load saved peripheral I/O when started in the HotStart state.

In order for S7-PLCSIM to start up and load peripheral I/O, the user must call **HotStartWithSavedValues** with a value of TRUE, save the PLC program (SavePLC), and set the startup state for S7-PLCSIM to HotStart (SetStartUpSwitch). When S7-PLCSIM restarts, it will then load the peripheral I/O.

Parameters

 **val** A value of TRUE indicates that S7-PLCSIM is to load saved peripheral I/O data on a hot start. A value of FALSE indicates that it should not.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

Visual Basic Usage

Sub **HotStartWithSavedValues**(*val As Long*)

 **Pause**

```
STDMETHOD(CS7ProSim::Pause)()
```

 **Description**

Pauses a simulation.

 **Parameters**

None

 **Error Handling**

Errors are returned in the ConnectionError event, not by the function call.

 **Return Value**

Value	Meaning
S_OK	0x00000000 : Success code
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

 **Visual Basic Usage**

```
Sub Pause()
```






ReadDataBlockValue

```
STDMETHOD (CS7ProSim::ReadDataBlockValue) (
    long BlockNumber,
    long ByteIndex,
    long BitIndex,
    PointDataTypeConstants DataType,
    VARIANT* pData)
```

Description

Reads a particular bit, byte, word, or double word from the DB memory area of S7-PLCSIM.

Parameters

-  **BlockNumber** Data block number to read. Valid values for BlockNumber are dependent on the CPU.
-  **ByteIndex** Byte starting position in the data block to read. Valid values for ByteIndex are dependent on the CPU.
-  **BitIndex** Bit starting position in the data block to read, if reading a boolean (bit) value. Valid values for BitIndex are 0 to 7.
-  **DataType** Type of data to read. DataType must be one of the PointDataTypeConstants.
-  **pData** Pointer to the space for the returned value. You must allocate and free this memory area in your application.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off
PS_E_BADTYPE	0x80040206 : Invalid data type
PS_E_BADBYTENDX	0x80040201 : Byte index is invalid
PS_E_BADBYTECOUNT	0x80040202 : Size of data array is invalid for given starting byte index
PS_E_READFAILED	0x80040203 : Read operation failed

Visual Basic Usage

Sub **ReadDataBlockValue**(*BlockNum As Long, ByteIndex As Long, BitIndex As Long, DataType As [PointDataTypeConstants](#), pData*)





ReadFlagValue

```
STDMETHOD (CS7ProSim::ReadFlagValue) ( long ByteIndex,
                                        long BitIndex,
                                        PointDataTypeConstants DataType,
                                        VARIANT* pData)
```

Description

Reads a particular bit, byte, word, or double word from the flag (M) memory area of S7-PLCSIM.

Parameters

-  **ByteIndex** Represents the byte starting position in M memory to read. Valid values for ByteIndex are dependent on the CPU.
-  **BitIndex** Represents the bit starting position in the M memory byte to read, if reading a boolean (bit) value. Valid values for BitIndex are 0 to 7.
-  **DataType** Represents the type of data to read. DataType must be one of the PointDataTypeConstants.
-  **pData** Pointer to the space for the returned value. You must allocate and free this memory area in your application.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off
PS_E_BADTYPE	0x80040206 : Invalid data type
PS_E_BADBYTENDX	0x80040201 : Byte index is invalid
PS_E_BADBYTECOUNT	0x80040202 : Size of data array is invalid for given starting byte index
PS_E_READFAILED	0x80040203 : Read operation failed

Visual Basic Usage

```
Sub ReadFlagValue(ByteIndex As Long, BitIndex As Long, DataType As PointDataTypeConstants, pData)
```





ReadOutputImage

```
STDMETHOD (CS7ProSim::ReadOutputImage) ( long StartIndex,
                                         long ElementsToRead,
                                         ImageDataTypeConstants DataType,
                                         VARIANT* pData)
```

Description

Reads elements from the peripheral output image (PQ memory area) of S7-PLCSIM.

Parameters

-  **StartIndex** Represents the byte starting position in the peripheral output image buffer to read. Valid values for StartIndex are dependent on the CPU.
-  **ElementsToRead** Represents the number of bytes, words, or double words to read from the image buffer. Valid values for ElementsToRead are dependent on the CPU.
-  **DataType** Represents the type of data to read. The DataType value must be one of the ImageDataTypeConstants.
-  **pData** Pointer to the space for returned elements. Valid values for data are dependent on ElementsToRead. You must allocate and free this memory area in your application.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_BADBYTENDX	0x80040201 : Byte index is invalid
PS_E_BADBYTECOUNT	0x80040202 : Size of data array is invalid for given starting byte index
PS_E_READFAILED	0x80040203 : Read operation failed
PS_E_BADTYPE	0x80040206 : Invalid data type
PS_E_NOTALLREADSWORKED	0x8004020F : All read operations did not succeed
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off

Visual Basic Usage

Function **ReadOutputImage**(*StartIndex* As Long, *ElementsToRead* As Long, *DataType* As **ImageDataTypeConstants**, *pData* As Long


 **ReadOutputPoint**


```
STDMETHOD (CS7ProSim::ReadOutputPoint) ( long ByteIndex,  
                                           long BitIndex,  
                                           PointDataTypeConstants DataType,  
                                           VARIANT* pData)
```

 **Description**


Reads a particular bit (Boolean), a byte (Byte), a two-byte word (Integer) or a four-byte word (Long) from the peripheral output image (PQ memory area).

 **Parameters**

 **ByteIndex** Represents the starting byte position in the peripheral image buffer to read. Valid values for ByteIndex are dependent on the CPU.

 **BitIndex** Represents the Bit position (in bytes) in the peripheral image buffer to read. Valid values are 0 to 7.

 **DataType** One of the PointDataTypeConstants

 **pData** Pointer to the data to read. Valid values for data are dependent on the data type.

 **Notes**

If the DataType parameter is S7_Bit, then ByteIndex and BitIndex must both be set to valid indexes. If successful, the method returns the given bit in pData, and its Variant data type is Boolean.

If the DataType parameter is S7_Byte, S7_Word, or S7_DoubleWord, then ByteIndex must be set to a valid index (BitIndex is ignored). If successful, the method returns the value in pData. The Variant data type is Byte, Integer, or Long, depending on the DataType parameter.

 **Error Handling**

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_BADBYTENDX	0x80040201 : Byte index is invalid
PS_E_BADBYTECOUNT	0x80040202 : Size of data array is invalid for given starting byte index
PS_E_READFAILED	0x80040203 : Read operation failed
PS_E_BADBITNDX	0x80040205 : Bit index is invalid
PS_E_BADTYPE	0x80040206 : Invalid data type
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off

Visual Basic Usage

Function **ReadOutputPoint**(ByteIndex As Long, BitIndex As Long, DataType As [PointDataTypeConstants](#), pData) As Long

 **SavePLC**

STDMETHOD(**CS7ProSim::SavePLC**)(BSTR FileName)

 **Description**

Saves the current simulated PLC data to a file.

The data that is saved consists of the program, the hardware configuration, the keyswitch position as indicated by the CPU view object, the type of scan (continuous or single scan), the I/O status, timer values, symbolic addresses, and the power setting (on or off).

 **Parameters**

 **FileName** Name of file in which to store the simulated PLC data

 **Error Handling**

Errors are returned in the ConnectionError event, not by the function call.

 **Return Value**

Value	Meaning
S_OK	0x00000000 : Success code
STG_E_CANTSAVE	0x80030103 : Can't save
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

 **Visual Basic Usage**

Sub **SavePLC**(*FileName As String*)


SetScanMode

STDMETHOD(**CS7ProSim::SetScanMode**)(ScanModeConstants *newVal*)

Description

Sets the scan mode for S7-PLCSIM.

Parameters

 *newVal* Scan mode to set for S7-PLCSIM. The scan mode must be one of the ScanModeConstants

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

Visual Basic Usage

Sub **SetScanMode**(*newVal* As **ScanModeConstants**)




SetStartUpSwitch

STDMETHOD(**CS7ProSim::SetStartUpSwitch**)(RestartSwitchPosition position)

Description

Sets the type of startup (Hot, Warm, or Cold) to use when S7-PLCSIM starts up.

Parameters

 position S7-PLCSIM startup position value to set

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

Visual Basic Usage

Sub **SetStartUpSwitch**(*position* As **RestartSwitchPosition**)


SetState

STDMETHOD(**CS7ProSim::SetState**)(BSTR newVal)

Description

Sets the current keyswitch position of S7-PLCSIM (RUN, RUN-P, or STOP).

Parameters

 newVal S7-PLCSIM keyswitch position value to set

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
E_INVALID_STATE	0x00008002 : Invalid state
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM

Visual Basic Usage

```
Sub SetState(newVal As String)
```


 **StartPLCSim**

STDMETHOD(**CS7ProSim::StartPLCSim**)(BSTR plcFile)

 **Description**

Starts S7-PLCSIM with the specified PLC simulation file (saved from a previous call to SavePLC).

 **Parameters**

 **plcFile** name of file with which to start S7-PLCSIM

 **Error Handling**

Errors are returned in the ConnectionError event, not by the function call.

 **Return Value**

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error

 **Visual Basic Usage**

Sub **StartPLCSim**(*plcFile As String*)





WriteDataBlockValue

```
STDMETHOD (CS7ProSim::WriteDataBlockValue) (
    long BlockNumber,
    long ByteIndex,
    long BitIndex,
    const VARIANT* pData)
```

Description

Writes a particular bit, byte, word, or double word to the DB memory area of S7-PLCSIM.

Parameters

-  **BlockNumber** Represents which data block number to write. Valid values for BlockNumber are dependent on the CPU.
-  **ByteIndex** Represents the byte starting position in the data block to be written. Valid values for ByteIndex are dependent on the CPU.
-  **BitIndex** Represents the bit starting position in the data block to be written, if writing a boolean (bit) value. Valid values for BitIndex are 0 to 7.
-  **pData** Pointer to the space containing the data to write. You must allocate and free this memory area in your application.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off
PS_E_BADTYPE	0x80040206 : Invalid data type
PS_E_BADBYTENDX	0x80040201 : Byte index is invalid
PS_E_BADBYTECOUNT	0x80040202 : Size of data array is invalid for given starting byte index
PS_E_WRITEFAILED	0x80040204 : Write operation failed

Visual Basic Usage

```
Sub WriteDataBlockValue(BlockNum As Long, ByteIndex As Long, BitIndex As Long, pData)
```




WriteFlagValue

```
STDMETHOD(CS7ProSim::WriteFlagValue)( long ByteIndex,
                                       long BitIndex,
                                       const VARIANT* pData)
```

Description

Writes a particular bit, byte, word, or double word to the flag (M) memory area of S7-PLCSIM.

Parameters

-  **ByteIndex** Represents the byte starting position in the M memory to be written. Valid values for ByteIndex are dependent on the CPU.
-  **BitIndex** Represents the bit starting position in the M memory byte to be written, if writing a boolean (bit) value. Valid values for BitIndex are 0 to 7.
-  **pData** Pointer to the space containing the data to write. You must allocate and free this memory area in your application.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off
PS_E_BADTYPE	0x80040206 : Invalid data type
PS_E_BADBYTENDX	0x80040201 : Byte index is invalid
PS_E_BADBYTECOUNT	0x80040202 : Size of data array is invalid for given starting byte index
PS_E_WRITEFAILED	0x80040204 : Write operation failed

Visual Basic Usage

```
Sub WriteFlagValue(ByteIndex As Long, BitIndex As Long, pData)
```



WriteInputImage

```
STDMETHOD(CS7ProSim::WriteInputImage)( long StartIndex, const  
VARIANT* pData)
```

Description

Writes elements to the peripheral input image (PI memory area) of S7-PLCSIM, starting at the StartIndex of the data pointed to by pData.

Parameters

-  **StartIndex** Represents the byte starting position in the peripheral input image buffer to write. Valid values for StartIndex are dependent on the CPU.
-  **pData** Pointer to the data for S7-PLCSIM to write. Valid values for data are dependent on the CPU. You must allocate and free this memory area in your application.

Notes

The type of elements to be written is determined by the type of the elements of Data. All elements have to be the same data type. An array of Bytes writes bytes, an array of Integer writes words, and an array of Long writes double words. The values written will be “raw” and not interpreted or converted by the method in any way. The number of elements written is determined by the size of the array pointed to by Data.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_BADBYTENDX	0x80040201 : Byte index is invalid
PS_E_BADBYTECOUNT	0x80040202 : Size of data array is invalid for given starting byte index
PS_E_WRITEFAILED	0x80040204 : Write operation failed
PS_E_BADTYPE	0x80040206 : Invalid data type
PS_E_NOTALLWRITESWORKED	0x80040210 : All write operations did not succeed
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off

Visual Basic Usage

Function **WriteInputImage**(*StartIndex As Long, Data As Long*) As Long




WriteInputPoint

```
STDMETHOD (CS7ProSim::WriteInputPoint) ( long ByteIndex,
                                         long BitIndex,
                                         const VARIANT* pData)
```

Description

Writes either a particular bit (Boolean), byte (Byte), a two-byte word (Integer) or a four-byte word (Long) from the Data Variant to the peripheral input image (PI memory area).

Parameters

-  **ByteIndex** Represents the starting byte position in the peripheral input image buffer to write. Valid values for ByteIndex are dependent on the CPU.
-  **BitIndex** Represents the Bit position (in bytes) in the peripheral image buffer to write. Valid values are 0 to 7.
-  **pData** Pointer to the data to write. Valid values for data are dependent on the data type.

Notes

If Boolean is given as the data type, then ByteIndex and BitIndex must both be set to valid indexes. If successful, the method writes the given bit at pData.

If Byte, Integer, or Long is given as the data type, then ByteIndex must be set to a valid index (BitIndex is ignored). If successful, the method writes the elements in pData.

Error Handling

Errors are returned in the ConnectionError event, not by the function call.

Return Value

Value	Meaning
S_OK	0x00000000 : Success code
E_FAIL	0x80004005 : Unspecified error
PS_E_BADBYTENDX	0x80040201 : Byte index is invalid
PS_E_BADBYTECOUNT	0x80040202 : Size of data array is invalid for given starting byte index
PS_E_WRITEFAILED	0x80040204 : Write operation failed
PS_E_BADBITNDX	0x80040205 : Bit index is invalid
PS_E_BADTYPE	0x80040206 : Invalid data type
PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off

Visual Basic Usage

Function **WriteInputPoint**(ByteIndex As Long, BitIndex As Long, Data As Long)

Events

- ◆ **ConnectionError** Generated when unable to connect to control engine ("S7-PLCSIM") or when an error occurs with any S7ProSim method call.
- ◆ **PauseStateChanged** Generated when a Pause/Continue state change is detected. NewState is a string that represents one of the PauseStateConstants.
- ◆ **PLCSimStateChanged** Generated when a new PLC switch state is detected. NewState is the new operating state: "RUN", "RUN_P", or "STOP".
- ◆ **ScanFinished** Generated when single scan is done. ScanInfo provides indexed information about the scan.
- ◆ **ScanModeChanged** Generated when a ScanMode change is detected. NewState is a string that represents one of the ScanModeConstants.

ConnectionError

HRESULT **ConnectionError**(BSTR ControlEngine, long Error)

Description

Generated when unable to connect to control engine ("S7-PLCSIM") or when an error occurs with any S7ProSim method call.

Visual Basic Usage

Event **ConnectionError**(*ControlEngine As String, Error As Long*)

 **PauseStateChanged**

HRESULT **PauseStateChanged**(BSTR NewState)

 **Description**

Generated when a Pause/Continue state change is detected. NewState is a string that represents one of the PauseStateConstants.

 **Visual Basic Usage**

Event **PauseStateChanged**(*NewState As String*)

PLCSimStateChanged

HRESULT **PLCSimStateChanged**(BSTR NewState)

Description

Generated when a new PLC switch state is detected. NewState is the new operating state: "RUN", "RUN_P", or "STOP".

Visual Basic Usage

Event **PLCSimStateChanged**(*NewState As String*)

 **ScanFinished**

HRESULT **ScanFinished**(VARIANT ScanInfo)

 **Description**

Generated when single scan is done. ScanInfo provides indexed information about the scan.

 **Visual Basic Usage**

Event **ScanFinished**(*ScanInfo*)

ScanModeChanged

HRESULT **ScanModeChanged**(BSTR NewState)









Description

Generated when a ScanMode change is detected. NewState is a string that represents one of the ScanModeConstants.

Visual Basic Usage

Event **ScanModeChanged**(*NewState As String*)

Type Definitions

 CPURunMode	Constants for the CPU run mode scan state
 ImageDataTypeConstants	Constants for the ReadOutputImage method
 PauseStateConstants	Constants for the pause state
 PointDataTypeConstants	Constants for the ReadOutputPoint method
 RestartSwitchPosition	Constants for the front panel startup switch position
 ScanModeConstants	Constants for the scan mode
 tagPauseState	Constants for the pause state
 ScanInfo Constants	Constants for information about the scan cycle

efesotomasyon.com

CPURunMode

```
enum CPURunMode { CONTINUOUS_SCAN, SINGLE_SCAN, SINGLE_STEP }
```

Description

Constants for the CPU run mode scan state

Members

CONTINUOUS_SCAN

SINGLE_SCAN

SINGLE_STEP



ImageDataTypeConstants

```
enum {  
    S7Byte = 2,  
    S7Word = 3,  
    S7DoubleWord = 4  
}
```



Description

Constants for the ReadOutputImage method



Members

S7Byte
S7DoubleWord
S7Word

PauseStateConstants

```
enum {  
    Running = 0,  
    Paused = 1,  
    Disabled = 2  
}
```

Description

Constants for the pause state

Members

Disabled

Paused

Running



PointDataTypeConstants

```
enum {  
    S7_Bit = 1,  
    S7_Byte = 2,  
    S7_Word = 3,  
    S7_DoubleWord = 4  
}
```

Description

Constants for the ReadOutputPoint method

Members

S7_Bit

S7_Byte

S7_DoubleWord

S7_Word

RestartSwitchPosition

```
enum {  
    WarmStart = 0,  
    HotStart = 1,  
    ColdStart = 2  
}
```

Description

Constants for the front panel startup switch position

Members

ColdStart Restart position OB102

HotStart Restart position OB101

WarmStart Restart position OB100

 **ScanModeConstants**

```
enum {  
    SingleScan = 0,  
    ContinuousScan = 1  
}
```

 **Description**

Constants for the scan mode

 **Members**

ContinuousScan

SingleScan

tagPauseState

```
enum tagPauseState { ENABLED_RUNNING, ENABLED_PAUSED, DISABLED }
```

Description

Constants for the pause state

Members

DISABLED







ENABLED_PAUSED

ENABLED_RUNNING

ScanInfo Constants

ScanInfo constants

 ScanInfo

 NUM_OF_SCANINFO_ELEMENTS	number of elements in ScanInfo return array.
 EXECUTION_TIME_NDX	index 0: execution time in ms
 MIN_CYCLE_TIME_NDX	index 1: shortest execution time value in ms
 LARGEST_CYCLE_TIME_NDX	index 2: largest execution time value in ms
 AVERAGE_CYCLE_TIME_NDX	index 3: average cycle time in ms
 IS_PLC_RUNNING_NDX	index 4: flag: 1=PLC is running; 0=PLC is not running

 **ScanInfo**

ScanInfo

The **ScanInfo** variant data type represents an array of longs. Each long in the array defines some information about the scan, as defined by the ScanInfo constants.

NUM_OF_SCANINFO_ELEMENTS

```
#define NUM_OF_SCANINFO_ELEMENTS 5
```

Description

number of elements in ScanInfo return array.

EXECUTION_TIME_NDX

```
#define EXECUTION_TIME_NDX 0
```

Description

index 0: execution time in ms

MIN_CYCLE_TIME_NDX

```
#define MIN_CYCLE_TIME_NDX 1
```

Description

index 1: shortest execution time value in ms

 **LARGEST_CYCLE_TIME_NDX**

```
#define LARGEST_CYCLE_TIME_NDX 2
```

 **Description**

index 2: largest execution time value in ms

 **AVERAGE_CYCLE_TIME_NDX**

```
#define AVERAGE_CYCLE_TIME_NDX 3
```

 **Description**

index 3: average cycle time in ms
























 **IS_PLC_RUNNING_NDX**

```
#define IS_PLC_RUNNING_NDX 4
```

 **Description**

index 4: flag: 1=PLC is running; 0=PLC is not running

Error return codes

 PS_E_BADBITNDX	0x80040205 : Bit index is invalid
 PS_E_BADBYTECOUNT	0x80040202 : Size of data array is invalid for given starting byte index
 PS_E_BADBYTENDX	0x80040201 : Byte index is invalid
 PS_E_BADTYPE	0x80040206 : Invalid data type
 PS_E_INVALIDCALLBACK	0x80040207 : Invalid callback
 PS_E_INVALIDDISPATCH	0x80040208 : Invalid dispatch
 PS_E_INVALIDINPUT	0x80040213 : Invalid input
 PS_E_INVALIDSCANTYPE	0x8004020B : Invalid scan type, must be one of the ScanModeConstants
 PS_E_MODENOTPOSSIBLE	0x8004020C : S7-PLCSIM could not set specified scan mode
 PS_E_NOTALLREADSWORKED	0x8004020F : All read operations did not succeed
 PS_E_NOTALLWRITESWORKED	0x80040210 : All write operations did not succeed
 PS_E_NOTCONNECTED	0x80040211 : S7ProSim is not connected to S7-PLCSIM
 PS_E_NOTIFICATION_EXIST	0x8004020D : S7ProSim is already registered for notification
 PS_E_NOTREGISTERED	0x80040209 : S7ProSim is not registered for callbacks from S7-PLCSIM
 PS_E_NOTSINGLESCAN	0x8004020A : S7-PLCSIM is not in single scan mode
 PS_E_PLCNOTRUNNING	0x8004020E : S7-PLCSIM is not running
 PS_E_POWEROFF	0x80040212 : S7-PLCSIM is powered off
 PS_E_READFAILED	0x80040203 : Read operation failed
 PS_E_WRITEFAILED	0x80040204 : Write operation failed
 E_FAIL	0x80004005 : Unspecified error
 E_INVALID_STATE	0x00008002 : Invalid state
 S_OK	0x00000000 : Success code
 STG_E_CANTSAVE	0x80030103 : Can't save

Index

A

AVERAGE_CYCLE_TIME_NDX, 55, 57

B

BeginScanNotify method, 13

C

ColdStart, 52

 setting, 34

Connect method, 14

ConnectionError event, 42

Constants, 47

 CPURunMode, 48

 ImageDataTypeConstants, 49

 PauseStateConstants, 50

 PointDataTypeConstants, 51

 RestartSwitchPosition, 52

 ScanModeConstants, 53

 tagPauseState, 54

Continue method, 15

Continuous scan execution

 GetScanMode, 22

 ScanModeConstants, 53

 SetScanMode, 33

CPURunMode, 48

D

Data Block (DB) values

 reading, 27

 writing, 37

Defined constants, 47

 CPURunMode, 48

 ImageDataTypeConstants, 49

 PauseStateConstants, 50

 PointDataTypeConstants, 51

 RestartSwitchPosition, 52

 ScanModeConstants, 53

 tagPauseState, 54

Disconnect method, 16

E

EndScanNotify method, 17

Enumerated types, 47

 CPURunMode, 48

 ImageDataTypeConstants, 49

 PauseStateConstants, 50

 PointDataTypeConstants, 51

 RestartSwitchPosition, 52

 ScanModeConstants, 53

 tagPauseState, 54

Error return codes, 59

Event handlers, 8

Events, 41

 ConnectionError, 42

 PauseStateChanged, 43

 PLCSimStateChanged, 44

 ScanFinished, 45

 ScanModeChanged, 46

ExecuteNmsScan Method, 18

ExecuteNScans method, 19

ExecuteSingleScan method, 20

EXECUTION_TIME_NDX, 55, 56

F

Flag value

 reading, 28

 writing, 38

G

GetPauseState method, 21

GetScanMode method, 22

GetStartUpSwitch method, 23

GetState method, 24

H

HotStart, 52

 setting, 25, 34

HotStartWithSavedValues method, 25

I

ImageDataTypeConstants, 49

Introduction, 7

IS_PLC_RUNNING_NDX, 55, 57

K

Keyswitch position

 getting, 24

 setting, 35

L

LARGEST_CYCLE_TIME_NDX, 55, 57

Loading peripheral I/O on hot start, 25

Loading saved .plc file, 36

M

M memory

 reading, 28

 writing, 38

Methods, 11

 BeginScanNotify, 13

 Connect, 14

 Continue, 15

 Disconnect, 16

 EndScanNotify, 17

 ExecuteNmsScan, 18

 ExecuteNScans, 19

 ExecuteSingleScan, 20

 GetPauseState, 21

 GetScanMode, 22

 GetStartUpSwitch, 23

 GetState, 24

 HotStartWithSavedValues, 25

 Pause, 26

 ReadDataBlockValue, 27

 ReadFlagValue, 28

 ReadOutputImage, 29

 ReadOutputPoint, 30

 SavePLC, 32

 SetScanMode, 33

 SetStartUpSwitch, 34

 SetState, 35

 StartPLCSim, 36

 WriteDataBlockValue, 37

 WriteFlagValue, 38

 WriteInputImage, 39

 WriteInputPoint, 40

MIN_CYCLE_TIME_NDX, 55, 56

N

NUM_OF_SCANINFO_ELEMENTS, 55, 56

O

Overview, 7

P

Pause method, 26

Pause state

 getting, 21

 setting, 15, 26

PauseStateChanged event, 43

PauseStateConstants, 50

PLCSIM, starting, 36

PLCSimStateChanged event, 44

PointDataTypeConstants, 51

Programming S7ProSim interface to S7-PLCSIM,
8

Project references, 7

R

ReadDataBlockValue method, 27

ReadFlagValue method, 28

Reading

 Data Block (DB) values, 27

 flag value (M memory), 28

 output image, 29

 output point, 30

ReadOutputImage method, 29

ReadOutputPoint method, 30

References, 7

- RestartSwitchPosition, 52
- Return values, 59
- S**
- S7-PLCSIM, starting, 25, 36
- S7ProSim
 - adding to VB project, 7
 - interface to S7-PLCSIM, programming, 8
 - overview, 7
- S7ProSim Pro methods
 - ReadDataBlockValue, 27
 - ReadFlagValue, 28
 - WriteDataBlockValue, 37
 - WriteFlagValue, 38
- SavePLC method, 32
- Saving .plc file, 32
- Scan execution methods
 - ExecuteNmsScan, 18
 - ExecuteNScans, 19
 - ExecuteSingleScan, 20
- Scan mode
 - getting, 22
 - setting, 33
- Scan notification, 13, 17
- ScanFinished event, 45
- ScanInfo constants, 55
- ScanModeChanged event, 46
- ScanModeConstants, 53
- SetScanMode method, 33
- SetStartUpSwitch method, 34
- SetState method, 35
- Siemens S7ProSim COM Object, adding to project, 7
- Single scan execution
 - ExecuteNmsScan, 18
 - ExecuteNScans, 19
 - ExecuteSingleScan, 20
- GetScanMode, 22
- ScanModeConstants, 53
- SetScanMode, 33
- StartPLCSim method, 36
- Startup switch position, 52
 - getting, 23
 - setting, 34
- T**
- tagPauseState, 54
- Type definitions, 47
 - CPURunMode, 48
 - ImageDataTypeConstants, 49
 - PauseStateConstants, 50
 - PointDataTypeConstants, 51
 - RestartSwitchPosition, 52
 - ScanModeConstants, 53
 - tagPauseState, 54
- V**
- Visual Basic project, adding S7ProSim, 7
- W**
- WarmStart, 52
 - setting, 34
- WriteDataBlockValue method, 37
- WriteFlagValue method, 38
- WriteInputImage method, 39
- WriteInputPoint method, 40
- Writing
 - Data Block (DB) values, 37
 - flag value (M memory), 38
 - input image, 39
 - input point, 40