

SIEMENS

SIMATIC
S7ProSim

Version: 5.0
User Manual

Edition 06/2001
2809923-0002

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 which, if not avoided, will result in death or serious injury.



Warning

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



Caution

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

Caution

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

Notice

NOTICE used without the safety alert symbol indicates a potential situation which, 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. ActiveX[™] is a trademark of Microsoft Corporation.

Copyright Siemens Energy & Automation, Inc. 2001 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.

Siemens Energy & Automation, ISBU
1 Internet Plaza
Johnson City, TN 37602-4991, USA

© Siemens Energy & Automation, Inc. 2001
Technical data subject to change.

Contents

S7ProSim Overview	1
Basic Tasks	3
Inserting the S7ProSim Control into a Visual Basic Application	3
Accessing S7ProSim Control Properties in Visual Basic.....	4
Properties	5
Properties of the S7ProSim Control.....	5
AutoConnect Property	6
ControlEngine Property	6
Enabled Property.....	6
ScanMode Property.....	6
Methods	7
Methods of the S7ProSim Control	7
AboutBox Method.....	8
BeginScanNotify Method	9
Return Codes of the BeginScanNotify Method	9
EndScanNotify Method.....	10
Return Codes of the EndScanNotify Method	10
Connect Method.....	11
Return Codes of the Connect Method	11
Disconnect Method.....	12
Return Codes of the Disconnect Method	12
ExecuteNmsScan Method	13
Return Codes of the ExecuteNmsScan Method	13
ExecuteNScans Method	14
Return Codes of the ExecuteNScans Method	14
ExecuteSingleScan Method	15
Return Codes of the ExecuteSingleScan Method	15
ReadOutputImage Method	16
Return Codes of the ReadOutputImage Method.....	16
ReadOutputPoint Method.....	17

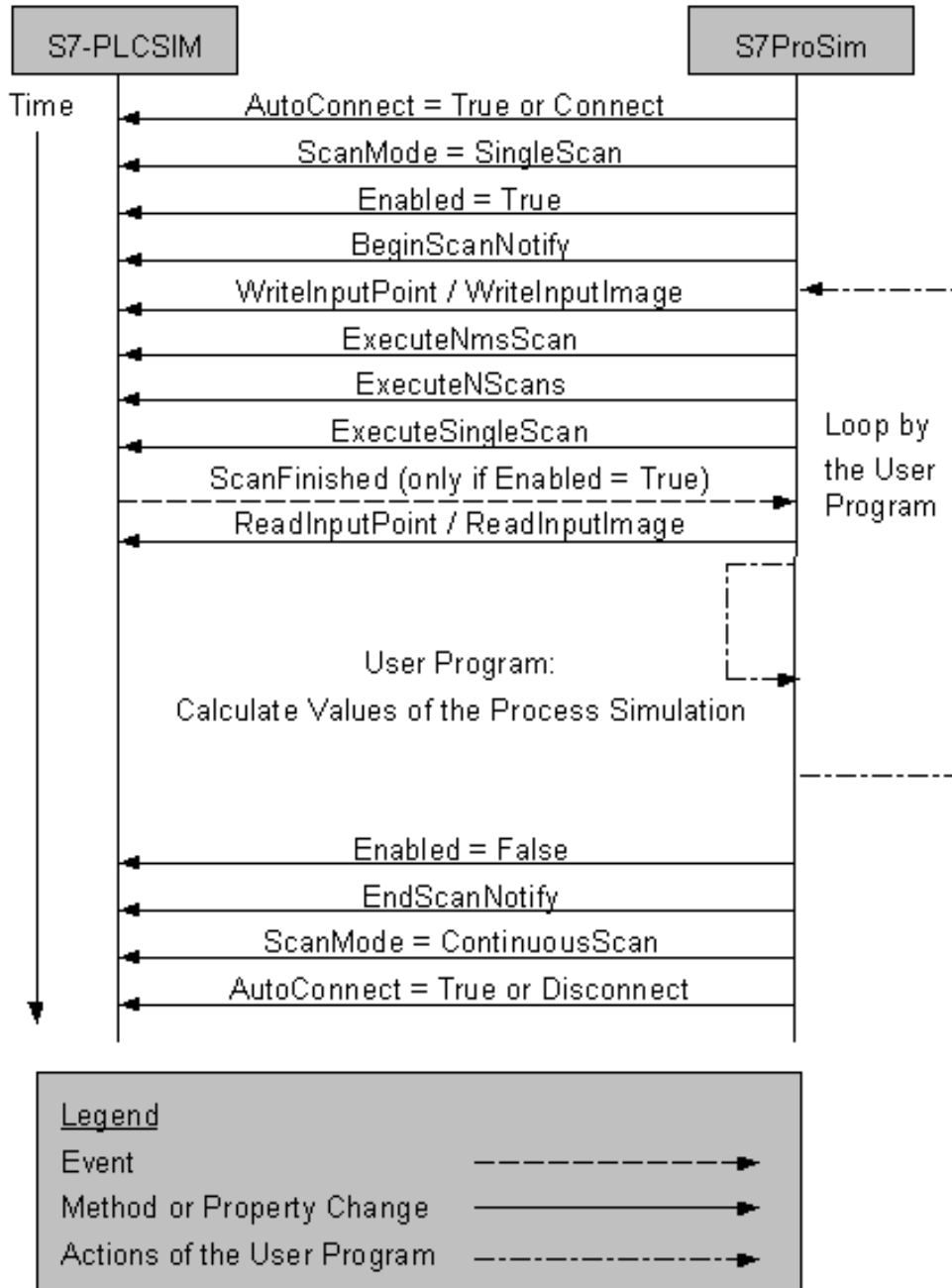
Return Codes of the ReadOutputPoint Method	17
WriteInputImage Method	18
Return Codes of the WriteInputImage Method.....	18
WriteInputPoint Method.....	19
Return Codes of the WriteInputPoint Method	19
Events	21
Events of the S7ProSim Control	21
ConnectionError Event	22
PLCSIMStateChanged Event	22
ScanFinished Event	22
Reference Information	23
Predefined Constants.....	23
Example Project	25
Code of the Example Project.....	26

S7ProSim Overview

S7ProSim is an ActiveX™ Control that provides programmatic access to the process simulation interface of S7-PLCSIM. This control may be used in any application that can accept ActiveX controls. You can use this control to attach a process simulation to S7-PLCSIM.

This online help describes the features and operations of the S7ProSim Control, Version 5.0. All properties, methods and events are used in an example project.

This ActiveX control provides all methods and events that are required for the interaction with S7-PLCSIM. The figure below shows the sequence chart for the different methods and events.



Basic Tasks

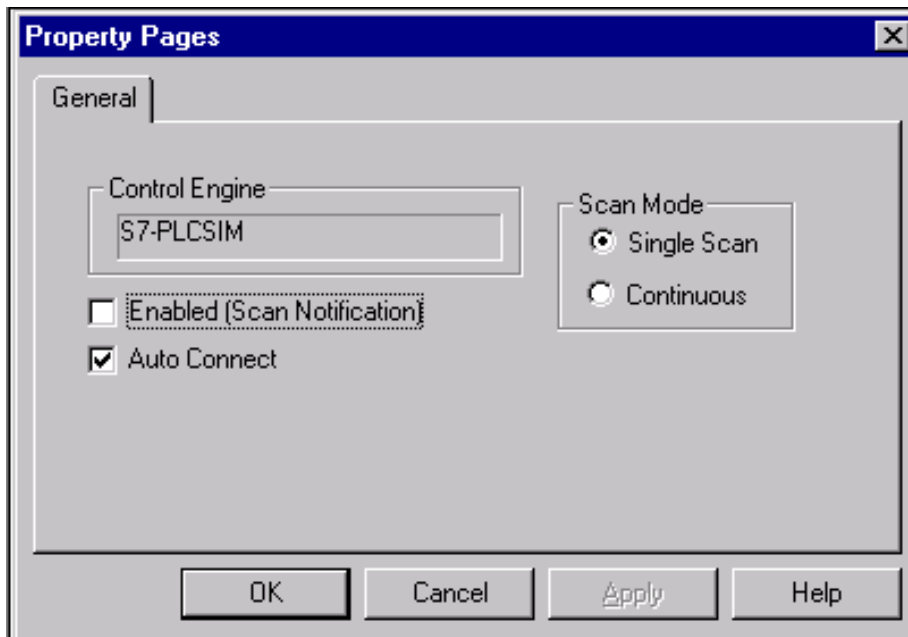
Inserting the S7ProSim Control into a Visual Basic Application

The S7ProSim Control can be used in a variety of third-party containers. Use the following procedure to use S7 controls in a Visual Basic form.

1. Select the menu command **Project -> Components** to display the Components dialog box.
2. From the scrollable list, select "Siemens S7ProSim Control".
3. Click Apply. An S7ProSim Control appears in the toolbox on the left of the Visual Basic form.
4. Click OK.
5. Select the S7ProSim Control in the toolbox and paste it into the form.

Accessing S7ProSim Control Properties in Visual Basic

When you select the S7ProSim Control, Visual Basic displays a secondary window with a list of properties. You can edit the properties in an additional Property Window.



The property window allows you to configure the following parameters of the S7ProSim control:

- **Auto Connect:** The AutoConnect property determines whether the control is connected to S7-PLCSIM automatically at startup or at the change from design mode to run mode.
- **Control Engine:** The ControlEngine property (read-only) defines the address of the control engine to which the S7ProSim Control connects. The address is S7-PLCSIM.
- **Enabled:** The Enabled property determines whether the control is registered or not. (ScanFinished event and PLCSIMStateChanged event are available.)
- **Scan Mode:** The ScanMode property sets the scan mode of S7-PLCSIM. The valid execution modes are SingleScan Mode or Continuous Mode.

Properties

Properties of the S7ProSim Control

The properties of the S7ProSim ActiveX control are the following:

- AutoConnect Property
- ControlEngine Property
- Enabled Property
- ScanMode Property

AutoConnect Property

The AutoConnect property determines whether the control is connected to S7-PLCSIM automatically at startup or at the change from design mode to run mode. If the AutoConnect property is set to False, the control has to be connected by the Connect method and disconnected by the Disconnect method.

Syntax: `S7ProSim1.AutoConnect = [Boolean]`

where:

Boolean is a boolean expression that specifies whether the ActiveX Control is automatically connected to S7-PLCSIM or not. The settings for Boolean are:

- True (default): The ActiveX control is automatically connected to and disconnected from the ControlEngine specified in S7-PLCSIM.
- False: The ActiveX control has to be connected and disconnected programmatically by the Connect method and Disconnect method.

ControlEngine Property

This read-only property defines the address of the control engine to which the S7ProSim Control connects. The address is S7-PLCSIM.

Enabled Property

The Enabled property determines whether the control is registered or not. (ScanFinished event and PLCSIMStateChanged event are available.)

Syntax: `S7ProSim1.Enabled = [Boolean]`

where:

Boolean is a boolean expression that specifies whether the ActiveX Control is registered for callbacks from the control engine or not (determines if the ScanFinished event and the PLCSIMStateChanged event are available or not). The settings for Boolean are:

- True: The ActiveX control is registered for callbacks from the control engine.
- False (default): The ActiveX control will be unregistered for the ScanFinished event and the PLCSIMStateChanged event for callbacks from the control engine set.

ScanMode Property

This property sets the scan mode of S7-PLCSIM. The valid execution modes are SingleScan Mode or ContinuousScan Mode.

Syntax: `S7ProSim1.ScanMode = [ScanModeConstants]`

where:

ScanModeConstants determines the scan mode of the control. Valid values are:

- ContinuousScan: S7-PLCSIM is set to continuous scan mode.
- SingleScan (default): S7-PLCSIM is set to single scan mode. S7-PLCSIM has to be in this mode for an attached process simulation.

Methods

Methods of the S7ProSim Control

The methods of the S7ProSim ActiveX control are the following:

- AboutBox Method
- BeginScanNotify Method
- EndScanNotify Method
- Connect Method
- Disconnect Method
- ExecuteNmsScan Method
- ExecuteNScans Method
- ExecuteSingleScan Method
- ReadOutputImage Method
- ReadOutputPoint Method
- WriteInputImage Method
- WriteInputPoint Method

AboutBox Method

With this method you can get specific information about the control and its version.

Syntax: S7ProSim1>AboutBox

The About Dialog Box can be seen in the figure below.



The AboutBox method has no parameters and no return code.

BeginScanNotify Method

The BeginScanNotify method registers the ActiveX control for callbacks from the control engine. (ScanFinished event and PLCSIMStateChanged event are available.)

Syntax: *result* = S7ProSim1.BeginScanNotify

where:

result is a Long value that indicates whether an error has occurred. The result is zero if no error occurs.

Return Codes of the BeginScanNotify Method

The following table lists possible return codes for the return value *result* of the BeginScanNotify method:

Constant	Value	Description
S_OK	&H0	Method was successful
PS_E_FAIL	&H80004005	Unknown error occurred
PS_E_NOTCONNECTED	&H80040211	The S7ProSim control is not connected to S7-PLCSIM
PS_E_POWEROFF	&H80040212	S7-PLCSIM is in Power-off state

EndScanNotify Method

The EndScanNotify method un-registers the ActiveX control for callbacks from the control engine. (ScanFinished event and PLCSIMStateChanged event are not available.)

Syntax: *result* = S7ProSim1.EndScanNotify

where:

result is a Long value that indicates whether an error has occurred. The result is zero if no error occurs.

Return Codes of the EndScanNotify Method

The following table lists possible return codes for the return value *result* of the EndScanNotify method:

Constant	Value	Description
S_OK	&H0	Method was successful
PS_E_FAIL	&H80004005	Unknown error occurred
PS_E_NOTCONNECTED	&H80040211	The S7ProSim control is not connected to S7-PLCSIM
PS_E_POWEROFF	&H80040212	S7-PLCSIM is in Power-off state
PS_E_NOTREGISTERED	&H80040209	The application is not registered

Connect Method

With this method the S7ProSim control can be connected programmatically to the S7-PLCSIM that is defined in the ControlEngine.

Syntax: *result* = S7ProSim1.Connect

where:

result is a Long value that indicates whether an error has occurred. The result is zero if no error occurs.

Return Codes of the Connect Method

The following table lists possible return codes for the return value *result* of the Connect method:

Constant	Value	Description
S_OK	&H0	Method was successful
PS_E_FAIL	&H80004005	Unknown error occurred
PS_E_NOTCONNECTED	&H80040211	The S7ProSim control is not connected to S7-PLCSIM
PS_E_POWEROFF	&H80040212	S7-PLCSIM is in Power-off state

Disconnect Method

With this method the S7ProSim control can be disconnected programmatically.

Syntax: *result* = S7ProSim1.Disconnect

where:

result is a Long value that indicates whether an error has occurred. The result is zero if no error occurs.

Return Codes of the Disconnect Method

The following table lists possible return codes for the return value *result* of the Disconnect method:

Constant	Value	Description
S_OK	&H0	Method was successful
PS_E_FAIL	&H80004005	Unknown error occurred
PS_E_POWEROFF	&H80040212	S7-PLCSIM is in Power-off state

ExecuteNmsScan Method

This method 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 the Enabled property is set to True, the program will be notified when S7-PLCSIM has finished the scans.

Syntax: *result* = S7ProSim1.ExecuteNmsScan (MsNumber)

where:

- *result* is a Long value that indicates whether an error has occurred. The result is zero if no error occurs.
- MsNumber is a Long value that indicates the time duration (in milliseconds) for which scan cycles are to be executed.

Note

This method is used only in SingleScan mode.

Return Codes of the ExecuteNmsScan Method

The following table lists possible return codes for the return value *result* of the ExecuteNmsScan method:

Constant	Value	Description
S_OK	&H0	Method was successful
PS_E_FAIL	&H80004005	Unknown error occurred
PS_E_NOTSINGLESCAN	&H8004020A	S7-PLCSIM is not in single scan mode
PS_E_PLCSIMNOTRUNNING	&H8004020E	S7-PLCSIM is not in Run or Run-P mode
PS_E_NOTCONNECTED	&H80040211	The S7ProSim control is not connected to S7-PLCSIM

ExecuteNScans Method

This method 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 the Enabled property is set to True, the program will be notified when S7-PLCSIM has finished the scans.

Syntax: *result* = S7ProSim1.ExecuteNScans (NScanNumber)

where:

- *result* is a Long value that indicates whether an error has occurred. The result is zero if no error occurs.
- NScanNumber is a Long value that indicates the number of scan cycles to be executed.

Note

This method is used only in SingleScan mode.

Return Codes of the ExecuteNScans Method

The following table lists possible return codes for the return value *result* of the ExecuteNScans method:

Constant	Value	Description
S_OK	&H0	Method was successful
PS_E_FAIL	&H80004005	Unknown error occurred
PS_E_NOTSINGLESCAN	&H8004020A	S7-PLCSIM is not in single scan mode
PS_E_PLCSIMNOTRUNNING	&H8004020E	S7-PLCSIM is not in Run or Run-P mode
PS_E_NOTCONNECTED	&H80040211	The S7ProSim control is not connected to S7-PLCSIM

ExecuteSingleScan Method

This method forces S7-PLCSIM to execute one scan cycle and does not wait for the execution of the current scan to finish. If the Enabled property is set to True, the program will be notified when S7-PLCSIM has finished the scan.

Syntax: *result* = S7ProSim1.ExecuteSingleScan

where:

result is a Long value that indicates whether an error has occurred. The result is zero if no error occurs.

Return Codes of the ExecuteSingleScan Method

The following table lists possible return codes for the return value *result* of the ExecuteSingleScan method:

Constant	Value	Description
S_OK	&H0	Method was successful
PS_E_FAIL	&H80004005	Unknown error occurred
PS_E_NOTSINGLESCAN	&H8004020A	S7-PLCSIM is not in single scan mode
PS_E_PLCSIMNOTRUNNING	&H8004020E	S7-PLCSIM is not in Run or Run-P mode
PS_E_NOTCONNECTED	&H80040211	The S7ProSim control is not connected to S7-PLCSIM

ReadOutputImage Method

Starting at the StartIndex location in the peripheral output image (PQ memory area), the number of bytes, words (2 bytes) or double words (4 bytes) (ElementsToRead) is read from the peripheral output image (PQ memory area) of S7-PLCSIM and placed in an array held in pData. The type of elements to be read is determined by the type requested in DataType parameter. All elements have the same data type. S7_Byte returns bytes, S7_Word returns words and S7_DoubleWord returns double words. The values read will be “raw” and not interpreted or converted by the method in any way.

Syntax: *result* = S7ProSim1.ReadOutputImage (StartIndex, ElementsToRead, DataType, pData)

where:

- StartIndex is an input parameter of type Long and represents the byte starting position in the peripheral image buffer to read. Valid values for StartIndex are dependent on the CPU.
- ElementsToRead is an input parameter of type Long and represents the number of bytes, words or double words to read in the image buffer. Valid values for ElementsToRead are dependent on the CPU.
- DataType is an input parameter and is one of the ImageDataTypeConstants. It has to be S7_Byte, S7_Word or S7_DoubleWord.
- pData is an output variable of type Variant and represents the space for returned elements. Valid values for pData are dependent on ElementsToRead. The memory is allocated by the server and freed by the application.
- *result* is a Long value that indicates whether an error has occurred. The result is zero if no error occurs.

Return Codes of the ReadOutputImage Method

The following table lists possible return codes for the return value *result* of the ReadOutputImage method:

Constant	Value	Description
S_OK	&H0	Method was successful
PS_E_FAIL	&H80004005	Unknown error occurred
PS_E_INVBYTENDX	&H80040201	'ByteIndex' value out of Range
PS_E_INVBYTECOUNT	&H80040202	'BytesToRead' out of range
PS_E_READFAILED	&H80040203	S7-PLCSIM refused read request
PS_E_INVTYPE	&H80040206	Invalid data type
PS_S_ALLREADSNOTPOSSIBLE	&H8004020F	The method could only read the configured outputs successfully
PS_E_NOTCONNECTED	&H80040211	The S7ProSim control is not connected to S7-PLCSIM
PS_E_POWEROFF	&H80040212	S7-PLCSIM is in Power-off state

ReadOutputPoint Method

The method 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).

If the `DataType` parameter is set to `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 set to `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`, and its Variant data type is Byte, Integer, or Long, depending on the `DataType` parameter.

Syntax: `result = S7ProSim1.ReadOutputPoint (ByteIndex, BitIndex, DataType, pData)`

where:

- `ByteIndex` is an input parameter of type Long and represents the byte position in the peripheral image buffer to read. Valid values for `ByteIndex` are dependent on the CPU.
- `BitIndex` is an input parameter of type Long and represents the bit position (of the byte) in the peripheral image buffer to read. Valid values are 0 to 7.
- `DataType` is an input parameter and is one of the `PointDataTypeConstants` and can be set to `S7_Bit`, `S7_Byte`, `S7_Word`, or `S7_DoubleWord`.
- `pData` is an output parameter of the type Variant and holds the read data. Valid values for `pData` are dependent on the data type.
- `result` is a Long value that indicates whether an error has occurred. The result is zero if no error occurs.

Return Codes of the ReadOutputPoint Method

The following table lists possible return codes for the return value `result` of the `ReadOutputPoint` method:

Constant	Value	Description
<code>S_OK</code>	<code>&H0</code>	Method was successful
<code>PS_E_FAIL</code>	<code>&H80004005</code>	Unknown error occurred
<code>PS_E_INVBYTENDX</code>	<code>&H80040201</code>	'ByteIndex' value out of Range
<code>PS_E_INVBYTECOUNT</code>	<code>&H80040202</code>	'ByteIndex' and size of Data array out of range
<code>PS_E_READFAILED</code>	<code>&H80040203</code>	S7-PLCSIM refused read request
<code>PS_E_INVBITNDX</code>	<code>&H80040205</code>	'BitIndex' value out of range
<code>PS_E_INVTYPE</code>	<code>&H80040206</code>	Invalid data type
<code>PS_E_NOTCONNECTED</code>	<code>&H80040211</code>	The S7ProSim control is not connected to S7-PLCSIM
<code>PS_E_POWEROFF</code>	<code>&H80040212</code>	S7-PLCSIM is in Power-off state

WriteInputImage Method

Starting at the StartIndex value, the elements held in Data will be written to the peripheral input image (PI memory area) of S7-PLCSIM. 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. Note that the number of elements written is determined by the size of the array held in Data.

Syntax: *result* = S7ProSim1.WriteInputImage (StartIndex, Data)

where:

- StartIndex is an input parameter of type Long and represents the byte starting position in the peripheral input image buffer to write. Valid values for StartIndex are dependent on the CPU.
- Data is an input parameter of type Variant and represents the array containing the elements to write. It is also dependent on the CPU. The memory is allocated and freed by the application. The elements will be written in S7-PLCSIM.
- *result* is a Long value that indicates whether an error has occurred. The result is zero if no error occurs.

Return Codes of the WriteInputImage Method

The following table lists possible return codes for the return value *result* of the WriteInputImage method:

Constant	Value	Description
S_OK	&H0	Method was successful
PS_E_FAIL	&H80004005	Unknown error occurred
PS_E_INVBYTENDX	&H80040201	'ByteIndex' value out of Range
PS_E_INVBYTECOUNT	&H80040202	'ByteIndex' and size of Data array out of range
PS_E_WRITEFAILED	&H80040204	S7-PLCSIM refused write request
PS_E_INVTYPE	&H80040206	Invalid data type
PS_S_ALLWRITESNOTPOSSIBLE	&H80040210	The method could only write the configured inputs successfully
PS_E_NOTCONNECTED	&H80040211	The S7ProSim control is not connected to S7-PLCSIM
PS_E_POWEROFF	&H80040212	S7-PLCSIM is in Power-off state

WriteInputPoint Method

The method 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).

If Boolean is given as the Variant's data type, then ByteIndex and BitIndex must both be set to valid indexes. If successful, the method writes the given bit in Data.

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

Syntax: *result* = S7ProSim1.WriteInputPoint (ByteIndex, BitIndex, Data)

where:

- ByteIndex is an input parameter of type Long and represents the starting byte position in the peripheral input image buffer to write. Valid values for ByteIndex are dependent on the CPU.
- BitIndex is an input parameter of type Long and represents the Bit position (in bytes) in the peripheral image buffer for write. Valid values are 0 to 7.
- Data is an input parameter of type Variant and represents the value to write. Valid values for Data are dependent on the data type.
- *result* is a Long value that indicates whether an error has occurred. The result is zero if no error occurs.

Return Codes of the WriteInputPoint Method

The following table lists possible return codes for the return value *result* of the WriteInputPoint method:

Constant	Value	Description
S_OK	&H0	Method was successful
PS_E_FAIL	&H80004005	Unknown error occurred
PS_E_INVBYTENDX	&H80040201	'ByteIndex' value out of Range
PS_E_INVBYTECOUNT	&H80040202	'ByteIndex' and size of Data array out of range
PS_E_WRITEFAILED	&H80040204	S7-PLCSIM refused write request
PS_E_INVBITNDX	&H80040205	'BitIndex' value out of range
PS_E_INVTYPE	&H80040206	Invalid data type
PS_E_NOTCONNECTED	&H80040211	The S7ProSim control is not connected to S7-PLCSIM
PS_E_POWEROFF	&H80040212	S7-PLCSIM is in Power-off state

Events

Events of the S7ProSim Control

The events of the S7ProSim ActiveX Control are the following:

- ConnectionError Event
- PLCSIMStateChanged Event
- ScanFinished Event

ConnectionError Event

This event occurs when an error on the connection occurs.

Syntax: ConnectionError(ByVal ControlEngine As String, ByVal Error As Long)

where:

- ControlEngine holds the name of the control engine.
- Error holds the Windows system error code. For more information see the Windows documentation.

PLCSIMStateChanged Event

This event occurs when a new PLC operating mode is detected.

Syntax: PLCSIMStateChanged(NewState As String)

where:

NewState holds the new operating state of S7-PLCSIM. Valid values are STOP, RUN and RUN_P.

ScanFinished Event

This event occurs when one scan of S7-PLCSIM has finished.

Syntax: ScanFinished (ByVal ScanInfo As Variant)

where:

ScanInfo is a SafeArray of Longs that represent various pieces of information about the scan. (See the table below for detailed information about the array.)

Elements of ScanInfo

<i>Index</i>	<i>Information</i>
0	Execution time (ms)
1	Programmed minimum cycle time (ms)
2	Largest execution time (ms)
3	Average cycle time (ms)
4	Running PLC flag

Reference Information

Predefined Constants

The following constants are used in the S7ProSim ActiveX control:

Constants	Value	Description
Scan mode values for SetScanMode() and GetScanMode()		
ContinuousScan	1	Simulation in ContinuousScan
SingleScan	0	Simulation in Single Scan
Read values for ReadOutputImage()		
S7_Byte	2	Byte data type
S7_Word	3	Word (2 byte) data type
S7_DoubleWord	4	Double Word (4 byte) data type
Read values for ReadOutputPoint()		
S7_Bit	1	Bit data byte
S7_Byte	2	Byte data type
S7_Word	3	Word (2 byte) data type
S7_DoubleWord	4	Double Word (4 byte) data type
Return code		
S_OK	&H0	Method was successful
PS_E_FAIL	&H80004005	Unknown error occurred
PS_E_INVBYTENDX	&H80040201	'ByteIndex' value out of Range
PS_E_INVBYTECOUNT	&H80040202	'ByteIndex' and size of Data array out of range or 'BytesToRead' out of range
PS_E_READFAILED	&H80040203	S7-PLCSIM refused read request
PS_E_WRITEFAILED	&H80040204	S7-PLCSIM refused write request
PS_E_INVBITNDX	&H80040205	'BitIndex' value out of range
PS_E_INVTYPE	&H80040206	Invalid data type
PS_E_NOTREGISTERED	&H80040209	The application is not registered
PS_E_NOTSINGLESCAN	&H8004020A	S7-PLCSIM is not in single scan mode
PS_E_PLCSIMNOTRUNNING	&H8004020E	S7-PLCSIM is not in Run or Run-P mode
PS_S_ALLREADSNOTPOSSIBLE	&H8004020F	The method could only read the configured outputs successful
PS_S_ALLWRITESNOTPOSSIBLE	&H80040210	The method could only write the configured inputs successful
PS_E_NOTCONNECTED	&H80040211	The S7ProSim control is not connected to S7-PLCSIM

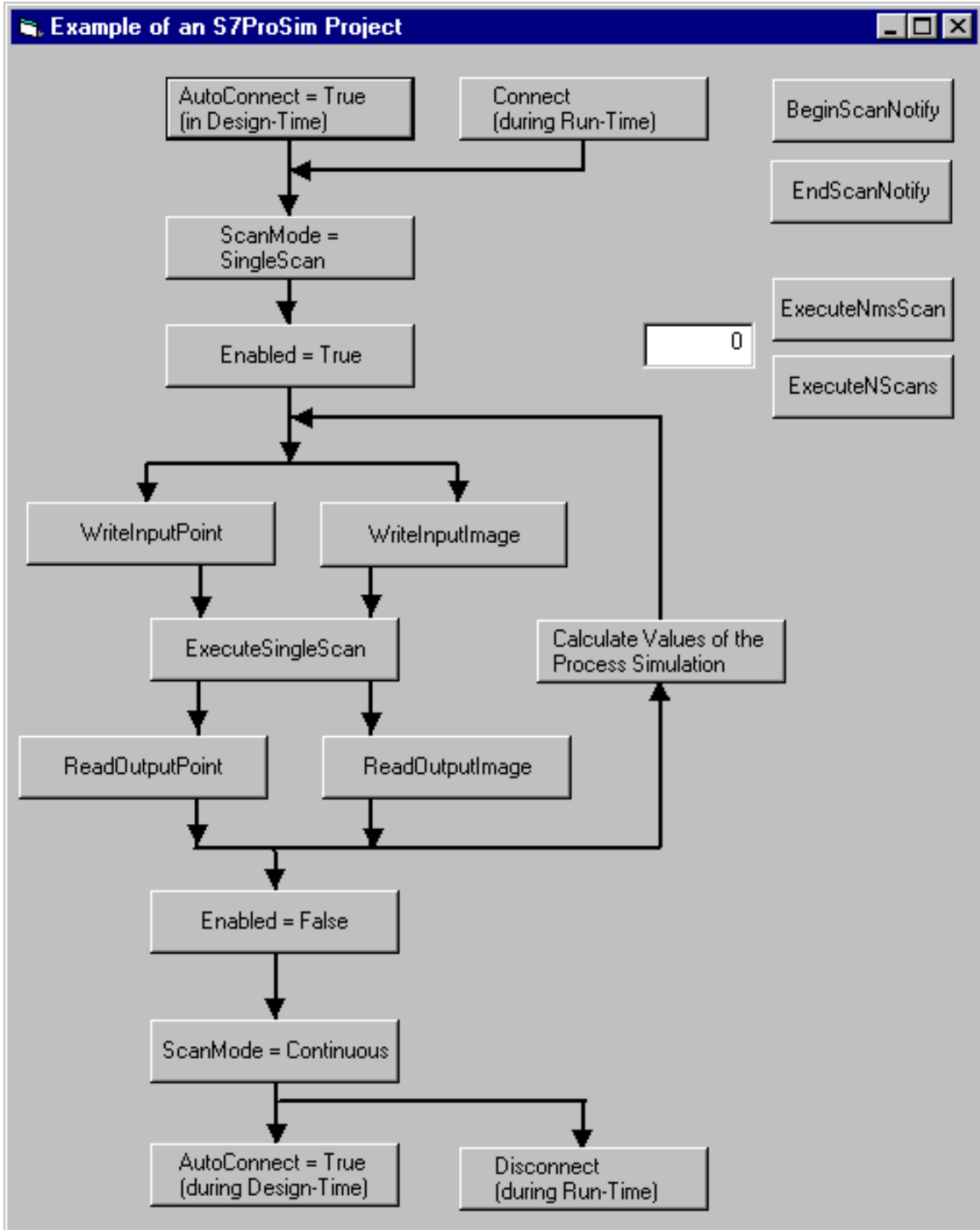
Predefined constants, continued...

PS_E_POWEROFF	&H80040212	S7-PLCSIM is in Power-off state
PS_E_INVALIDINPUT	&H80040213	Invalid input entered
<i>Number of elements in ScanInfo</i>		
NUM_OF_SCANINFO_ELEMENTS	4	
<i>Indexes of the ScanInfo</i>		
EXECUTION_TIME_NDX	0	Execution time (ms)
MIN_CYCLE_TIME_NDX	1	Programmed minimum cycle time (ms)
LARGEST_CYCLE_TIME_NDX	2	Largest execution time (ms)
AVERAGE_CYCLE_TIME_NDX	3	Average cycle time (ms)

Example Project

This example shows the usage of all properties, methods and events of the control. The form below is a representation of the sequence chart in the S7ProSim overview. All controls in the form are Visual Basic Buttons. See also the code of the example project.

Form of the example project



Code of the Example Project

The following code listing shows the implementation of the example project:

```
'=====
'DECLARATION PART OF THE FORM
'=====
'Variables must be declared
Option Explicit

'Default Error Code Values of S7ProSim
'-----
Private Const S_OK = &H0
Private Const PS_E_FAIL = &H80004005
Private Const PS_E_INVBYTENDX = &H80040201
Private Const PS_E_INVBYTECOUNT = &H80040202
Private Const PS_E_READFAILED = &H80040203
Private Const PS_E_WRITEFAILED = &H80040204
Private Const PS_E_INVBITNDX = &H80040205
Private Const PS_E_INVTYPE = &H80040206
Private Const PS_E_NOTREGISTERED = &H80040209
Private Const PS_E_NOTSINGLESCAN = &H8004020A
Private Const PS_E_MODENOTPOSSIBLE = &H8004020C
Private Const PS_E_NOTIFICATION_EXIST = &H8004020D
Private Const PS_E_PLCSIMNOTRUNNING = &H8004020E
Private Const PS_S_ALLREADSNOTPOSSIBLE = &H8004020F
Private Const PS_S_ALLWRITESNOTPOSSIBLE = &H80040210
Private Const PS_E_NOTCONNECTED = &H80040211
Private Const PS_E_POWEROFF = &H80040212

'Default Error Text
'-----
Private Const MSG_OK = "&H0: Method was successful"
Private Const MSG_FAIL = "&H80004005: Unknown error occurred"
Private Const MSG_INVBYTENDX = _
    "&H80040201: ByteIndex value out of Range"
Private Const MSG_INVBYTECOUNT = _
    "&H80040202: ByteIndex + size of Data array out of range or BytesToRead out of
Range"
Private Const MSG_READFAILED = _
    "&H80040203: S7-PLCSIM refused read request"
Private Const MSG_WRITEFAILED = _
    "&H80040204: S7-PLCSIM refused write request"
Private Const MSG_INVBITNDX = _
    "&H80040205: BitIndex value out of range"
Private Const MSG_INVTYPE = "&H80040206: Invalid data type"
Private Const MSG_NOTREGISTERED = _
    "&H80040209: The application is not registered"
Private Const MSG_NOTSINGLESCAN = _
    "&H8004020A: S7-PLCSIM is not in single scan mode"
Private Const MSG_NOTIFICATION_EXIST = _
    "&H8004020D: Application is already registered"
Private Const MSG_PLCSIMNOTRUNNING = _
    "&H8004020E: S7-PLCSIM is not in Run or Run-P mode"
Private Const MSG_ALLREADSNOTPOSSIBLE = _
    "&H8004020F: Only the configured outputs could be read successful"
Private Const MSG_ALLWRITESNOTPOSSIBLE = _
    "&H80040210: Only the configured inputs could be written successful"
Private Const MSG_NOTCONNECTED = _
    "&H80040211: The S7ProSim control is not connected to S7-PLCSIM"
Private Const MSG_POWEROFF = _
    "&H80040212: S7-PLCSIM is in Power-off state"
```

```
'=====
'CODE FOR THE BUTTONS
'=====

'cmdAutoConnectTrueStart
'-----
Private Sub cmdAutoConnectTrueStart_Click()
    S7ProSim1.AutoConnect = True
End Sub

'cmdAutoConnectTrueEnd
'-----
Private Sub cmdAutoConnectTrueEnd_Click()
    S7ProSim1.AutoConnect = True
End Sub

'cmdCalculateValuesOfProcessSimulation
'-----
Private Sub cmdCalculateValuesOfProcessSimulation_Click()
    '***** TODO by the User *****
    '***** In this function you have to implement the *****
    '***** Code for the Process Simulation. Take the *****
    '***** values from the Outputs of S7-PLCSIM and *****
    '***** calculate the new values for the Input of *****
    '***** S7-PLCSIM. *****
End Sub

'cmdConnect
'-----
Private Sub cmdConnect_Click()
    Dim errConnect As Long

    errConnect = S7ProSim1.Connect
    If errConnect = S_OK Then
        MsgBox MSG_OK, vbInformation, "S7ProSim Example"
    Else
        ShowError errConnect
    End If
End Sub

'cmdDisconnect
'-----
Private Sub cmdDisconnect_Click()
    Dim errDisconnect As Long

    errDisconnect = S7ProSim1.Disconnect
    If errDisconnect = S_OK Then
        MsgBox MSG_OK, vbInformation, "S7ProSim Example"
    Else
        ShowError errDisconnect
    End If
End Sub

'cmdEnableTrue
'-----
Private Sub cmdEnableTrue_Click()
    S7ProSim1.Enabled = True
End Sub

'cmdEnableFalse
'-----
```

```
Private Sub cmdEnableFalse_Click()
    S7ProSim1.Enabled = False
End Sub

'cmdScanModeSingle
'-----
Private Sub cmdScanModeSingle_Click()
    S7ProSim1.ScanMode = SingleScan
End Sub

'cmdScanModeCont
'-----
Private Sub cmdScanModeCont_Click()
    S7ProSim1.ScanMode = ContinuousScan
End Sub

'cmdExecuteSingleScan
'-----
Private Sub cmdExecuteSingleScan_Click()
    Dim errExecuteSingleScan As Long

    errExecuteSingleScan = S7ProSim1.ExecuteSingleScan
    If errExecuteSingleScan = S_OK Then
        MsgBox MSG_OK, vbInformation, "S7ProSim Example"
    Else
        ShowError errExecuteSingleScan
    End If
End Sub

'cmdReadOutputImage
'-----
Private Sub cmdReadOutputImage_Click()
    'Long
    Dim errReadOutputImage As Long
    Dim lStartIndex As Long
    Dim lElementsToRead As Long
    'ImageDataTypeConstants
    Dim DataType As ImageDataTypeConstants
    'Variant
    Dim vData As Variant

    '***** Read 2 Bytes at the starting address Q 8.0 *****
    DataType = S7Byte 'Read type Byte
    lStartIndex = 8 'Start at address Q 8.0
    lElementsToRead = 2 'Read 2 elements (Bytes)

    errReadOutputImage = S7ProSim1.ReadOutputImage(lStartIndex, _
    lElementsToRead, DataType, vData)
    If errReadOutputImage = S_OK Then
        MsgBox "Value of QB 8 is: " & CByte(vData(0)) & vbCrLf & _
        "Value of QB 9 is: " & CByte(vData(1)), _
        vbInformation, "S7ProSim Example"
    Else
        ShowError errReadOutputImage
    End If
End Sub
```

```

***** Read 2 Words at the starting address Q 10.0 *****
DataType = S7Word 'Read type Word
lStartIndex = 10 'Start at address Q 10.0
lElementsToRead = 2 'Read 2 Elements (Words)

errReadOutputImage = S7ProSim1.ReadOutputImage(lStartIndex, _
lElementsToRead, DataType, vData)
If errReadOutputImage = S_OK Then
    MsgBox "Value of QW 10 is: " & CInt(vData(0)) & vbCrLf & _
    "Value of QW 12 is: " & CInt(vData(1)), _
    vbInformation, "S7ProSim Example"
Else
    ShowError errReadOutputImage
End If

***** Read 2 DoubleWords at the starting address Q 14.0 *****
DataType = S7DoubleWord 'Read type DoubleWord
lStartIndex = 14 'Start at address Q 14.0
lElementsToRead = 2 'Read 2 Elements (DoubleWords)

errReadOutputImage = S7ProSim1.ReadOutputImage(lStartIndex, _
lElementsToRead, DataType, vData)
If errReadOutputImage = S_OK Then
    MsgBox "Value of QD 14 is: " & CLng(vData(0)) & vbCrLf & _
    "Value of QD 18 is: " & CLng(vData(1)), _
    vbInformation, "S7ProSim Example"
Else
    ShowError errReadOutputImage
End If

***** After this section the calculations for the *****
***** Process Simulation can be done if the return *****
***** value is S_OK. *****
End Sub

'cmdReadOutputPoint
'-----
Private Sub cmdReadOutputPoint_Click()
    'Long
    Dim errReadOutputPoint As Long
    Dim lByteIndex As Long
    Dim lBitIndex As Long
    'PointDataTypeConstants
    Dim DataType As PointDataTypeConstants
    'Variant
    Dim vData As Variant

    ***** Read the Bit at the address Q 0.5 *****
    lByteIndex = 0 'Start at address 0.0
    lBitIndex = 5 'Read specific Bit 5 (of Byte 0)
    DataType = S7_Bit 'Read type Bit

    errReadOutputPoint = S7ProSim1.ReadOutputPoint(lByteIndex, _
    lBitIndex, DataType, vData)
    If errReadOutputPoint = S_OK Then
        MsgBox "The current value of Q 0.5 is: " & CInt(vData), _
        vbInformation, "S7ProSim Example"
    Else
        ShowError errReadOutputPoint
    End If

```

```

'***** Read the Byte at the address Q 1.0 *****
lByteIndex = 1 'Start at address 1.0
DataType = S7_Byte 'Read type Byte

errReadOutputPoint = S7ProSim1.ReadOutputPoint(lByteIndex, _
lBitIndex, DataType, vData)
If errReadOutputPoint = S_OK Then
    MsgBox "The current value of QB 1 is: " & CByte(vData), _
        vbInformation, "S7ProSim Example"
Else
    ShowError errReadOutputPoint
End If

'***** Read the Word at the address Q 2.0 *****
lByteIndex = 2 'Start at address 2.0
DataType = S7_Word 'Read type Word

errReadOutputPoint = S7ProSim1.ReadOutputPoint(lByteIndex, _
lBitIndex, DataType, vData)
If errReadOutputPoint = S_OK Then
    MsgBox "The current value of QW 2 is: " & CInt(vData), _
        vbInformation, "S7ProSim Example"
Else
    ShowError errReadOutputPoint
End If

'***** Read the DoubleWord at the address Q 4.0 *****
lByteIndex = 4 'Start at address 4.0
DataType = S7_DoubleWord 'Read type DoubleWord

errReadOutputPoint = S7ProSim1.ReadOutputPoint(lByteIndex, _
lBitIndex, DataType, vData)
If errReadOutputPoint = S_OK Then
    MsgBox "The current value of QD 4 is: " & CLng(vData), _
        vbInformation, "S7ProSim Example"
Else
    ShowError errReadOutputPoint
End If

'***** After this section the calculations for the *****
'***** Process Simulation can be done if the return *****
'***** value is S_OK. *****
End Sub
'cmdWriteInputImage
'-----
Private Sub cmdWriteInputImage_Click()
    'Byte
    Dim cByteArray(0 To 1) As Byte
    'Integer
    Dim iWordArray(0 To 1) As Integer
    'Long
    Dim errWriteInputImage As Long
    Dim lDoubleWordArray(0 To 1) As Long
    Dim lStartIndex As Long
    'Variant
    Dim vData As Variant

```

```

'***** Write 2 Bytes and start at address I 8.0 *****
cByteArray(0) = 8 'Write 8 in first element (Byte)
cByteArray(1) = 9 'Write 9 in second element (Byte)
lStartIndex = 8 'Start at address I 8.0

vData = cByteArray
errWriteInputImage = S7ProSim1.WriteInputImage(lStartIndex, _
vData)
'***** After this section the calculations for the *****
'***** Process Simulation can be done if the return *****
'***** value is S_OK. *****
If errWriteInputImage = S_OK Then
    MsgBox MSG_OK, vbInformation, "S7ProSim Example"
Else
    ShowError errWriteInputImage
End If

'***** Write 2 Words and start at address I 10.0 *****
iWordArray(0) = 10 'Write 10 in first element (Word)
iWordArray(1) = 12 'Write 12 in second element (Word)
lStartIndex = 10 'Start at address I 10.0

vData = iWordArray
errWriteInputImage = S7ProSim1.WriteInputImage(lStartIndex, _
vData)
If errWriteInputImage = S_OK Then
    MsgBox MSG_OK, vbInformation, "S7ProSim Example"
Else
    ShowError errWriteInputImage
End If

'***** Write 2 DoubleWords and start at address I 14.0 *****
lDoubleWordArray(0) = 14 'Write 14 in first element (DoubleWord)
lDoubleWordArray(1) = 18 'Write 18 in second element (DoubleWord)
lStartIndex = 14 'Start at address I 14.0

vData = lDoubleWordArray
errWriteInputImage = S7ProSim1.WriteInputImage(lStartIndex, _
vData)
If errWriteInputImage = S_OK Then
    MsgBox MSG_OK, vbInformation, "S7ProSim Example"
Else
    ShowError errWriteInputImage
End If
End Sub

'cmdWriteInputPoint
'-----
Private Sub cmdWriteInputPoint_Click()
    'Boolean
    Dim bBoolIn As Boolean
    'Byte
    Dim cByteIn As Byte
    'Integer
    Dim iWordIn As Integer
    'Long
    Dim errWriteInputPoint As Long
    Dim lBitIndex As Long
    Dim lByteIndex As Long
    Dim lDoubleWordIn As Long
    'Variant
    Dim vData As Variant

```

```
***** Write 1 Bit to the address I 0.5 *****
bBoolIn = 1 'Write value 1
lByteIndex = 0 'Start at address 0.0
lBitIndex = 5 'Write specific Bit 5 (of Byte 0)

vData = bBoolIn
errWriteInputPoint = S7ProSim1.WriteInputPoint(lByteIndex, _
lBitIndex, vData)
***** After this section the calculations for the *****
***** Process Simulation can be done if the return *****
***** value is S_OK. *****
If errWriteInputPoint = S_OK Then
    MsgBox MSG_OK, vbInformation, "S7ProSim Example"
Else
    ShowError errWriteInputPoint
End If

***** Write 1 Byte to the address I 1.0 *****
cByteIn = 1 'Write value 1
lByteIndex = 1 'Start at address 1.0

vData = cByteIn
errWriteInputPoint = S7ProSim1.WriteInputPoint(lByteIndex, _
lBitIndex, vData)
If errWriteInputPoint = S_OK Then
    MsgBox MSG_OK, vbInformation, "S7ProSim Example"
Else
    ShowError errWriteInputPoint
End If

***** Write 1 Word to the address I 2.0 *****
iWordIn = 2 'Write value 2
lByteIndex = 2 'Start at address 2.0

vData = iWordIn
errWriteInputPoint = S7ProSim1.WriteInputPoint(lByteIndex, _
lBitIndex, vData)
If errWriteInputPoint = S_OK Then
    MsgBox MSG_OK, vbInformation, "S7ProSim Example"
Else
    ShowError errWriteInputPoint
End If

***** Write 1 DoubleWord to the address I 4.0 *****
lDoubleWordIn = 4 'Write value 4
lByteIndex = 4 'Start at address 4.0

vData = lDoubleWordIn
errWriteInputPoint = S7ProSim1.WriteInputPoint(lByteIndex, _
lBitIndex, vData)
If errWriteInputPoint = S_OK Then
    MsgBox MSG_OK, vbInformation, "S7ProSim Example"
Else
    ShowError errWriteInputPoint
End If
End Sub
```

```

'=====
'EVENTS IMPLEMENTATION FOR THE CONTROL
'=====

'ConnectionError
'-----
Private Sub S7ProSim1_ConnectionError(ByVal ControlEngine As String, _
    ByVal Error As Long)
    Dim errMessage As String
    errMessage = "Unable to connect to " & ControlEngine & vbCrLf
    errMessage = errMessage & vbCrLf & _
        "Start " & ControlEngine & vbCrLf
    errMessage = errMessage & "and connect with Connect method"
    MsgBox errMessage, vbExclamation, "Connection Error"
End Sub

'PLCStateChanged
'-----
Private Sub S7ProSim1_PLCSIMStateChanged(ByVal NewState As String)
    Dim cMessage As String

    cMessage = "PLCSIM changed the operating state to " & NewState
    MsgBox cMessage, vbInformation, "S7ProSim Example"
End Sub

'ScanFinished
'-----
Private Sub S7ProSim1_ScanFinished(ByVal ScanInfo As Variant)
    Dim cMessage As String
    Dim vArrayInfo As Variant
    '***** Before this section of code, the calculations *****
    '***** for the Process Simulation should be done. *****
    vArrayInfo = ScanInfo
    cMessage = "Last scan took " & vArrayInfo(0) & vbCrLf
    cMessage = cMessage & _
        "minimum cycle time " & vArrayInfo(1) & vbCrLf
    cMessage = cMessage & _
        "Largest Execution time took " & vArrayInfo(2) & vbCrLf
    cMessage = cMessage & _
        "Average scan took " & vArrayInfo(3)
    MsgBox cMessage, vbInformation, "S7ProSim Example"
End Sub
Private Sub cmdBeginScanNotify_Click()
    S7ProSim1.BeginScanNotify
End Sub
Private Sub cmdEndScanNotify_Click()
    S7ProSim1.EndScanNotify
End Sub
Private Sub cmdExecuteNmsScan_Click()
    Dim ReturnValue As Long
    ReturnValue = S7ProSim1.ExecuteNmsScan(Int(txtScanNumber.Text))
    If ReturnValue <> 0 Then
        MsgBox "Failed!", vbOKOnly
    End If
End Sub

```

```

Private Sub cmdExecuteNScan_Click()
    Dim ReturnValue As Long
    ReturnValue = S7ProSim1.ExecuteNScans(Int(txtScanNumber.Text))
    If ReturnValue <> 0 Then
        MsgBox "Failed!", vbOKOnly
    End If
End Sub
Private Sub Form_Unload(cancel As Integer)
    Dim errDisconnect As Long
    errDisconnect = S7ProSim1.Disconnect
    If errDisconnect = S_OK Then
        MsgBox MSG_OK, vbInformation, "S7ProSim Example"
    Else
        ShowError errDisconnect
    End If
End Sub
'=====
'PRIVATE SUBS
'=====

'ShowError
'-----
Private Sub ShowError(ErrorNumber)
    Select Case ErrorNumber
        Case PS_E_FAIL
            MsgBox MSG_FAIL, vbExclamation, "S7ProSim Example"
        Case PS_E_INVBYTENDX
            MsgBox MSG_INVBYTENDX, vbExclamation, "S7ProSim Example"
        Case PS_E_INVBYTECOUNT
            MsgBox MSG_INVBYTECOUNT, vbExclamation, "S7ProSim Example"
        Case PS_E_READFAILED
            MsgBox MSG_READFAILED, vbExclamation, "S7ProSim Example"
        Case PS_E_WRITEFAILED
            MsgBox MSG_WRITEFAILED, vbExclamation, "S7ProSim Example"
        Case PS_E_INVBITNDX
            MsgBox MSG_INVBITNDX, vbExclamation, "S7ProSim Example"
        Case PS_E_INVTYPE
            MsgBox MSG_INVTYPE, vbExclamation, "S7ProSim Example"
        Case PS_E_NOTREGISTERED
            MsgBox MSG_NOTREGISTERED, vbExclamation, "S7ProSim Example"
        Case PS_E_NOTSINGLESCAN
            MsgBox MSG_NOTSINGLESCAN, vbExclamation, "S7ProSim Example"
        Case PS_E_NOTIFICATION_EXIST
            MsgBox MSG_NOTIFICATION_EXIST, vbExclamation, _
                "S7ProSim Example"
        Case PS_E_PLCSIMNOTRUNNING
            MsgBox MSG_PLCSIMNOTRUNNING, vbExclamation, _
                "S7ProSim Example"
        Case PS_S_ALLREADSNOTPOSSIBLE
            MsgBox MSG_ALLREADSNOTPOSSIBLE, vbExclamation, _
                "S7ProSim Example"
        Case PS_S_ALLWRITESNOTPOSSIBLE
            MsgBox MSG_ALLWRITESNOTPOSSIBLE, vbExclamation, _
                "S7ProSim Example"
        Case PS_E_NOTCONNECTED
            MsgBox MSG_NOTCONNECTED, vbExclamation, "S7ProSim Example"
        Case PS_E_POWEROFF
            MsgBox MSG_POWEROFF, vbExclamation, "S7ProSim Example"
        Case Else
            MsgBox "System Error occured: &H" & Hex(ErrorNumber), _
                vbExclamation, "S7ProSim Example"
    End Select
End Sub

```

Index

A

AboutBox method, 8
Accessing S7ProSim control properties in Visual Basic, 4
AutoConnect property, 4, 6

B

BeginScanNotify method, 9
BitIndex, 17, 19
ByteIndex, 17, 19

C

Code of the example project, 26
Components, 3
Connect method, 6, 11
ConnectionError event, 22
Constants, 23
ContinuousScan mode, 4, 6
ControlEngine, 22
ControlEngine property, 4, 6

D

Data parameter, 18, 19
DataType parameter, 16, 17
Disconnect method, 6, 12

E

ElementsToRead, 16
Enabled property, 4, 6
EndScanNotify method, 10
Events of the S7ProSim control, 21
Example project, 25, 26
ExecuteNmsScan method, 13
ExecuteNScans method, 14
ExecuteSingleScan method, 15

I

Inserting the S7ProSim control into a Visual Basic application, 3

M

Methods, 7, 8
Methods of the S7ProSim control, 7
MsNumber, 13

N

NewState, 22
NScanNumber, 14

O

Overview, 1

P

pData, 16, 17
PLCSIMStateChanged event, 6, 10, 22
Predefined constants, 23
Properties of the S7ProSim control, 5
Property Pages dialog, 3
PS_E_FAIL, 9, 11, 13, 14, 15, 16, 17, 18, 19
PS_E_INVBITNDX, 17, 19
PS_E_INVBYTECOUNT, 16, 17, 18, 19
PS_E_INVBYTENDX, 16, 17, 18, 19
PS_E_INVTYPE, 16, 18, 19
PS_E_NOTCONNECTED, 9, 11, 13, 14, 15, 16, 17, 18, 19
PS_E_NOTSINGLESCAN, 13, 14, 15
PS_E_PLCSIMNOTRUNNING, 13, 14, 15
PS_E_POWEROFF, 9, 11, 16, 17, 18, 19
PS_E_READFAILED, 16, 17
PS_E_WRITEFAILED, 18, 19
PS_S_ALLREADSNOTPOSSIBLE, 16
PS_S_ALLWRITESNOTPOSSIBLE, 18

R

ReadOutputImage method, 16
ReadOutputPoint method, 17
Return codes of the BeginScanNotify method, 9
Return codes of the Connect method, 11
Return codes of the Disconnect method, 12
Return codes of the EndScanNotify method, 10
Return codes of the ExecuteNmsScan method, 13
Return codes of the ExecuteNScans method, 14
Return codes of the ExecuteSingleScan method, 15
Return codes of the ReadOutputImage method, 16
Return codes of the ReadOutputPoint method, 17

Return codes of the WriteInputImage
method, 18

Return codes of the WriteInputPoint method,
19

S

S7ProSim, 1

Constants, 23

Events, 21

Methods, 7

Properties, 5

S7ProSim control, 3

S7ProSim control properties, 4

S7ProSim overview, 1

ScanFinished event, 6, 10, 22

ScanInfo, 22

ScanMode property, 4, 6

Sequence chart, 1

SingleScan mode, 4, 6

StartIndex, 16, 18

V

Visual Basic, 3, 4

W

WriteInputImage method, 18

WriteInputPoint method, 19

Please check any industry that applies to you:

- Automotive
- Chemical
- Electrical Machinery
- Food
- Instrument and Control
- Non electrical Machinery
- Petrochemical
- Pharmaceutical
- Plastic
- Pulp and Paper
- Textiles
- Transportation
- Other _____

Mail your response to:

SIEMENS ENERGY & AUTOMATION, INC
ATTN: TECHNICAL COMMUNICATIONS M/S 519
3000 BILL GARLAND ROAD
PO BOX 1255
JOHNSON CITY TN USA 37605-1255

Include this information:

From

Name: _____
Job Title: _____
Company Name _____
Street: _____
City and State: _____
Country: _____
Telephone: _____