

# **SIEMENS**

**SIMATIC**

**S7ProSim V5.3 incl. SP1**

**ActiveX Control**

**User Manual**

Edition: 01/2005

**A5E00425523-01**

# 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<sup>®</sup> and SIMATIC<sup>®</sup> are registered trademarks of SIEMENS AG.

STEP 7<sup>™</sup> and S7<sup>™</sup> are trademarks of SIEMENS AG.

Microsoft<sup>®</sup>, Windows<sup>®</sup>, Windows 95<sup>®</sup>, Windows 98<sup>®</sup>, Windows NT<sup>®</sup>, Windows ME<sup>®</sup>, and Windows 2000<sup>®</sup> are registered trademarks of Microsoft Corporation. ActiveX<sup>™</sup> is a trademark of Microsoft Corporation.

## Copyright Siemens Energy & Automation, Inc. 2005

### **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.

Siemens Energy & Automation, PCbA  
One Internet Plaza  
Johnson City, TN 37602-4991, USA

## **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, Inc. 2005

Technical data subject to change.

# 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 V5.3 incl. SP1.

## 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

<b>S7ProSim Overview .....</b>	<b>1</b>
Inserting the S7ProSim Control into a Visual Basic Application .....	2
Accessing S7ProSim Control Properties in Visual Basic.....	3
Programming an Interface to S7-PLCSIM with S7ProSim.....	4
<b>Methods .....</b>	<b>5</b>
AboutBox .....	6
BeginScanNotify .....	7
Connect .....	8
Disconnect.....	9
EndScanNotify .....	10
ExecuteNmsScan .....	11
ExecuteNScans .....	12
ExecuteSingleScan.....	13
ReadOutputImage .....	14
ReadOutputPoint .....	15
WriteInputImage .....	16
WriteInputPoint .....	17
<b>Events .....</b>	<b>19</b>
ConnectionError.....	20
PLCSimStateChanged.....	21
ScanFinished.....	22
<b>Properties .....</b>	<b>23</b>
AutoConnect .....	24
ControlEngine .....	25
Enabled.....	26
ScanMode.....	27
<b>Type Definitions .....</b>	<b>29</b>
ImageDataTypeConstants .....	30
PointDataTypeConstants.....	31
ScanModeConstants .....	32
<b>Error return codes.....</b>	<b>33</b>
<b>Example Project Using S7ProSim ActiveX Control.....</b>	<b>35</b>
Code of the Example Project.....	36

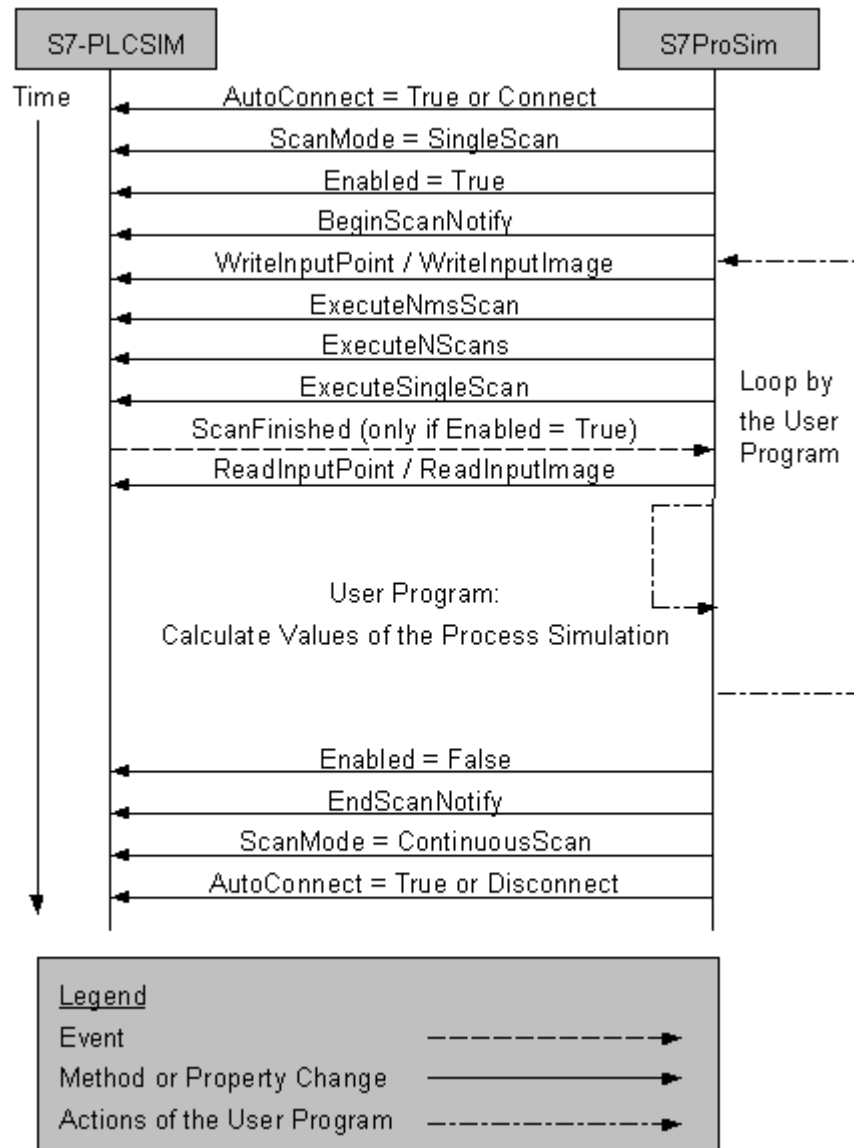


# S7ProSim Overview

S7ProSim provides an ActiveX™ Control that provides programmatic access to the process simulation interface of S7-PLCSIM. You can use S7ProSim in any application that can accept ActiveX controls 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. The example project demonstrates the use of S7ProSim methods and events.

The figure below shows the sequence chart for the different methods and events used in the example project.



## **Inserting the S7ProSim Control into a Visual Basic Application**

The S7ProSim ActiveX Control can be used in a variety of third-party containers. Use the following procedure to add the S7ProSim Control to a Visual Basic form:

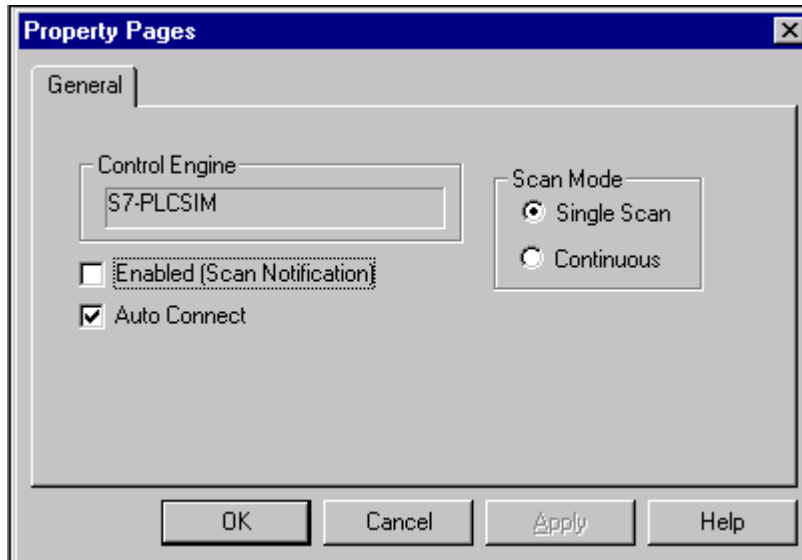
1. Select the **Project -> Components** menu command to display the Components dialog box.
2. From the list of controls, select "Siemens S7ProSim Control".
3. Click OK. An S7ProSim Control appears in the toolbox on the left of the Visual Basic form.
4. Select the S7ProSim Control in the toolbox and paste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

You can now access any of the S7ProSim Control properties, methods, and events from your Visual Basic program.



## Accessing S7ProSim Control Properties in Visual Basic

When you right-click the S7ProSim Control on the form and select **Properties** from the context menu, Visual Basic displays a Property Pages dialog:



The property window allows you to configure the following properties of the S7ProSim ActiveX 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.

## 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 Control in the project as a project component.
- Program event handlers for the S7ProSim events (optional). Within each event handler, you can insert any custom code for your application.

### Example: Visual Basic 6.0

```
Private Sub S7ProSim1_ScanFinished(ByVal ScanInfo As Variant)
    ...
End Sub

Private Sub S7ProSim1_PLCSimStateChanged(ByVal NewState As String)
    ...
End Sub

Private Sub S7ProSim1_ConnectionError(ByVal ControlEngine As String, ByVal error
As Long)
    MsgBox "Connection Error"
End Sub
```

- 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

- ◆ **AboutBox** Displays the AboutBox dialog.
- ◆ **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.
- ◆ **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.
- ◆ **ReadOutputImage** Reads elements from the peripheral output image (PQ memory area) of S7-PLCSIM.
- ◆ **ReadOutputPoint** 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).
- ◆ **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).

## AboutBox

```
void AboutBox()
```

## Visual Basic Usage

```
Sub AboutBox()
```

 **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

 **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

### **Return Value**

<b>Value</b>	<b>Meaning</b>
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

 **Disconnect**

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

 **Description**

Disconnects S7ProSim from S7-PLCSIM.

 **Parameters**

None

 **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

### 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.

### 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_PLCNORUNNING	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

### 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_PLCSIMNOTRUNNING	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

### Return Value

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

### Visual Basic Usage

Function **ExecuteSingleScan()** As Long





## 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.

### 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.

### 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)




## 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.

### 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




## 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.

### 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.
- ◆ **PLCSimStateChanged** Generated when a new PLC switch state is detected.
- ◆ **ScanFinished** Generated when single scan is done.



## ConnectionError

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



### Description

Generated when unable to connect to control engine.



### Visual Basic Usage

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

 **PLCSimStateChanged**

HRESULT **PLCSimStateChanged**(BSTR *NewState*)

 **Description**

Generated when a new PLC switch state is detected.

 **Visual Basic Usage**

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

## ScanFinished

HRESULT **ScanFinished**(VARIANT ScanInfo)





### Description

Generated when single scan is done.

### Visual Basic Usage

Event **ScanFinished**(*ScanInfo*)

# Properties

-  **AutoConnect** Determines whether the control is connected to S7-PLCSIM automatically at startup or at the change from design mode to run mode.
-  **ControlEngine** Defines the name of the control engine (read-only) to which the S7ProSim Control connects. The name is "S7-PLCSIM".
-  **Enabled** Toggles whether the control is registered for callbacks for ScanFinished and PLCSimStateChanged events
-  **ScanMode** Sets/returns the current control engine scan mode

## **AutoConnect**

boolean **AutoConnect**

### **Description**

Determines whether the control is connected to S7-PLCSIM automatically at startup or at the change from design mode to run mode.



## ControlEngine

BSTR ControlEngine

### Description

Defines the name of the control engine (read-only) to which the S7ProSim Control connects. The name is "S7-PLCSIM".

 **Enabled**

boolean **Enabled**

 **Description**

Toggles whether the control is registered for callbacks for ScanFinished and PLCSimStateChanged events.



## **ScanMode**




ScanModeConstants **ScanMode**

### **Description**

Sets/returns the current control engine scan mode to either SingleScan or ContinuousScan.



# Type Definitions

-  **ImageDataTypeConstants** Constants for the ReadOutputImage method
-  **PointDataTypeConstants** Constants for the ReadOutputPoint method
-  **ScanModeConstants** Constants for the scan mode

## ImageDataTypeConstants

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

### Description

Constants for the ReadOutputImage method

### Members

*S7Byte*  
*S7DoubleWord*  
*S7Word*

 **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*

## ScanModeConstants

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

### Description

Constants for the scan mode

### Members

*ContinuousScan*

*SingleScan*

## 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

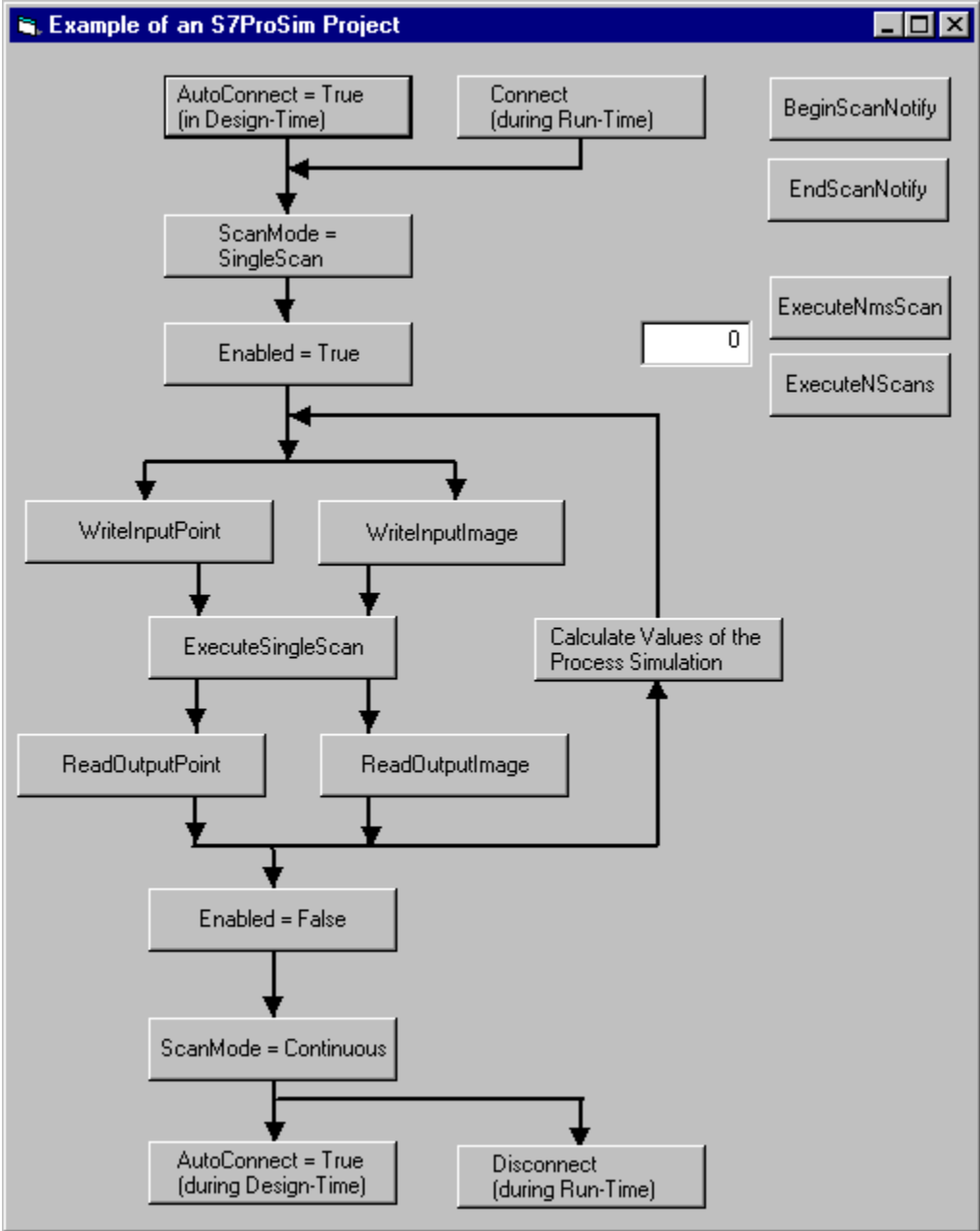




# Example Project Using S7ProSim ActiveX Control

This Visual Basic example shows the usage of all properties, methods and events of the S7ProSim ActiveX Control. You can see the code behind each of the command button handlers in the topic Example Code: Project Using S7ProSim ActiveX Control.

## 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 E_FAIL = &H80004005
Private Const PS_E_BADBYTENDX = &H80040201
Private Const PS_E_BADBYTECOUNT = &H80040202
Private Const PS_E_READFAILED = &H80040203
Private Const PS_E_WRITEFAILED = &H80040204
Private Const PS_E_BADBITNDX = &H80040205
Private Const PS_E_BADTYPE = &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_BADBYTENDX = _
"&H80040201: ByteIndex value out of Range"
Private Const MSG_BADBYTECOUNT = _
"&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_BADBITNDX = _
"&H80040205: BitIndex value out of range"
Private Const MSG_BADTYPE = "&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.ReadOuputImage(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

'***** 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.ReadOuputImage(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 cyle 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 E_FAIL
MsgBox MSG_FAIL, vbExclamation, "S7ProSim Example"
Case PS_E_BADBYTENDX
MsgBox MSG_BADBYTENDX, vbExclamation, "S7ProSim Example"
Case PS_E_BADBYTECOUNT
MsgBox MSG_BADBYTECOUNT, 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_BADBITNDX
MsgBox MSG_BADBITNDX, vbExclamation, "S7ProSim Example"
Case PS_E_BADTYPE
MsgBox MSG_BADTYPE, 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 ocured: &H" & Hex(ErrorNumber), _
vbExclamation, "S7ProSim Example"
End Select
End Sub
```

# Index

<b>A</b>	
AboutBox method.....	6
Adding S7ProSim to VB application.....	2
AutoConnect property .....	3, 24
<b>B</b>	
BeginScanNotify method.....	7
<b>C</b>	
Code, example project .....	36
Connect method .....	8
ConnectionError event .....	20
Constants .....	29
ImageDataTypeConstants.....	30
PointDataTypeConstants.....	31
ScanModeConstants .....	32
Continuous scan execution	
ScanMode .....	27
ScanModeConstants .....	32
Control engine name.....	25
ControlEngine property .....	3, 25
<b>D</b>	
Defined constants .....	29
ImageDataTypeConstants.....	30
PointDataTypeConstants.....	31
ScanModeConstants .....	32
Disconnect method .....	9
<b>E</b>	
Enabled property .....	3, 26
EndScanNotify method .....	10
Enumerated types .....	29
ImageDataTypeConstants.....	30
PointDataTypeConstants.....	31
ScanModeConstants .....	32
Error return codes .....	33
Event handlers .....	4
Events.....	19
ConnectionError.....	20
PLCSimStateChanged.....	21
ScanFinished .....	22
Example Project	
code .....	36
overview .....	35
ExecuteNmsScan Method.....	11
ExecuteNScans method.....	12
ExecuteSingleScan method .....	13
<b>I</b>	
ImageDataTypeConstants.....	30
Inserting S7ProSim into VB application.....	2
Introduction.....	1
<b>M</b>	
Methods .....	5
AboutBox .....	6
BeginScanNotify .....	7
Connect.....	8
Disconnect .....	9
EndScanNotify .....	10
ExecuteNmsScan .....	11
ExecuteNScans .....	12
ExecuteSingleScan.....	13
ReadOutputImage.....	14
ReadOutputPoint .....	15
WriteInputImage.....	16
WriteInputPoint .....	17
<b>O</b>	
Overview.....	1
<b>P</b>	
PLCSimStateChanged event.....	21
PointDataTypeConstants.....	31
Programming S7ProSim interface to S7-PLCSIM .....	4

Project Components, VB.....	2	Scan notification .....	7, 10, 26
Properties .....	3, 23	ScanFinished event .....	22
AutoConnect .....	24	ScanMode property .....	3, 27
ControlEngine .....	25	ScanModeConstants .....	32
Enabled .....	26	Siemens S7ProSim Control, adding to VB project .....	2
ScanMode .....	27	Single scan execution	
<b>R</b>			
Reading		ExecuteNmsScan .....	11
output image.....	14	ExecuteNScans .....	12
output point.....	15	ExecuteSingleScan.....	13
ReadOutputImage method.....	14	ScanMode .....	27
ReadOutputPoint method.....	15	ScanModeConstants.....	32
Return values .....	33	<b>T</b>	
<b>S</b>			
S7ProSim		Type definitions .....	29
adding to VB project .....	2	ImageDataTypeConstants .....	30
interface to S7-PLCSIM, programming .....	4	PointDataTypeConstants.....	31
overview .....	1	ScanModeConstants.....	32
properties.....	3	<b>V</b>	
Scan execution methods		Visual Basic project, adding S7ProSim .....	2
ExecuteNmsScan .....	11	<b>W</b>	
ExecuteNScans .....	12	WriteInputImage method .....	16
ExecuteSingleScan .....	13	WriteInputPoint method .....	17
Scan mode		Writing	
getting .....	27	input image .....	16
setting .....	27	input point .....	17



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  
One Internet Plaza  
Johnson City TN USA 37604

Include this information:

**From**

Name: \_\_\_\_\_  
Job Title: \_\_\_\_\_  
Company Name: \_\_\_\_\_  
Street: \_\_\_\_\_  
City and State: \_\_\_\_\_  
Country: \_\_\_\_\_  
Telephone: \_\_\_\_\_