Agilent Microplate Centrifuge ActiveX

Version 8.0.0

User Guide

Original Instructions

Agilent Technologies

Notices

User Guide Part Number

G5405-90003

Edition

Revision 00, month year

Contact Information

Agilent Technologies Inc.
Automation Solutions
5301 Stevens Creek Blvd.
Santa Clara, CA 95051
USA

Technical Support: 1.800.979.4811
or +1.408.345.8011
service.automation@agilent.com

Customer Service: 1.866.428.9811
or +1.408.345.8356
orders.automation@agilent.com

European Service: +44 (0)1763208826
euroservice.automation@agilent.com

Documentation feedback:
documentation.automation@agilent.com

Web:
www.agilent.com/lifesciences/automation

Acknowledgements

Microsoft® and Windows® are either registered trademarks or trademarks of the Microsoft Corporation in the United States and other countries.

Warranty

The material contained in this document is provided "as is," and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Agilent and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

Restricted Rights Legend

If software is for use in the performance of a U.S. Government prime contract or subcontract, Software is delivered and licensed as "Commercial computer software" as defined in DFAR 252.227-7014 (June 1995), or as a "commercial item" as defined in FAR 2.101(a) or as "Restricted computer software" as defined in FAR 52.227-19 (June 1987) or any equivalent agency regulation or contract clause. Use, duplication or disclosure of Software is subject to Agilent Technologies' standard commercial license terms, and non-DOD Departments and Agencies of the U.S. Government will receive no greater than Restricted Rights as defined in FAR 52.227-19(c)(1-2) (June 1987). U.S. Government users will receive no greater than Limited Rights as defined in FAR 52.227-14 (June1987) or DFAR 252.227-7015 (b)(2) (November 1995), as applicable in any technical data.

Safety Notices

A WARNING notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in personal injury or death. Do not proceed beyond a WARNING notice until the indicated conditions are fully understood and met.

A **CAUTION** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a **CAUTION** notice until the indicated conditions are fully understood and met.

Contents

Contents

Preface

This preface contains the following topics:

About this guide

What this guide covers

This guide describes the ActiveX controls for the Agilent Microplate Centrifuge.

This guide does not provide instructions for setting up and using the Agilent Microplate Centrifuge. For these details, see the *Microplate Centrifuge User Guide*.

Accessing Agilent Technologies Automation Solutions user guides

You can search the online knowledge base or download the latest version of any PDF file from the Agilent Technologies website at www.agilent.com/lifesciences/automation. Safety information for the devices appears in the corresponding device user guide. You can also search the knowledge base or the PDF files for safety information.

Related topics

For information about...	See...
How to set up and use the Agilent Microplate Centrifuge	*Microplate Centrifuge User Guide*
Reporting problems	"Reporting problems" on page iv

Reporting problems

Contacting Automation Solutions Technical Support

If you find a problem with the Microplate Centrifuge, contact Automation Solutions Technical Support. For contact information, see Notices on the back of the title page.

Reporting hardware problems

When contacting Agilent Technologies, make sure you have the serial number of the device ready.

Reporting software problems

When you contact Automation Solutions Technical Support, make sure you provide the following:

- Short description of the problem

- Relevant software version number (for example, automation control software, diagnostics software, ActiveX control software, and firmware)
- Error message text (or screen capture of the error message dialog box)
- Relevant files, such as log files

Reporting user guide problems

If you find a problem with this user guide or have suggestions for improvement, send your comments in an email to documentation.automation@agilent.com.

Related topics

For information about...	See...
How to set up and use the Microplate Centrifuge	*Microplate Centrifuge User Guide*
Accessing user information	"Accessing Agilent Technologies Automation Solutions user guides" on page iv

Centrifuge ActiveX control

This chapter gives integrators the ActiveX control information required to integrate the Microplate Centrifuge into another company's lab automation system. The ActiveX has been verified to work with both Visual C++ and Visual Basic .NET.

This chapter contains the following topics:

About ActiveX controls

What is the Centrifuge ActiveX control

The Centrifuge ActiveX control is the software component that allows third-party lab automation systems to interact with the Microplate Centrifuge.

How the Centrifuge ActiveX control is used

In an Agilent Technologies automation system that is running the VWorks software, ActiveX interfaces are not used to communicate with devices. However, some integrations, such as those with LIMS, require that a third-party application control the Microplate Centrifuge. The Centrifuge ActiveX control enables third-party applications to interface with the Microplate Centrifuge.

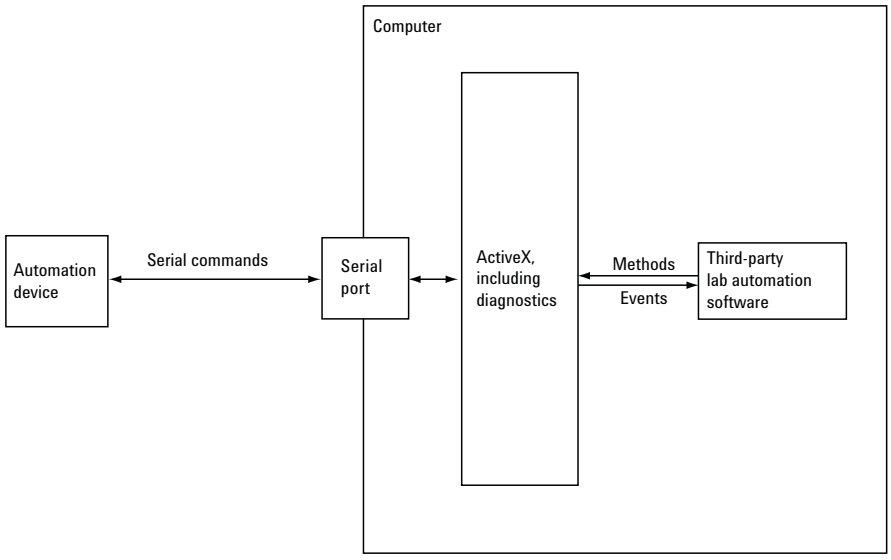
Each ActiveX control consists of a collection of the following:

- *Methods*. Functions that can be called to invoke individual operations
- *Properties*. Attributes or features of the ActiveX control
- *Events*. Notifications that methods have completed or resulted in errors

To ensure proper integration, you must know the available methods and properties for the ActiveX control.

The following diagram illustrates the use of the Centrifuge ActiveX control in a lab automation system environment. Actions you perform are conducted through ActiveX methods. System responses are relayed back through ActiveX events or through return values and variables passed to methods.

Note: Although the Centrifuge ActiveX control generates events, the third-party application must implement handlers for them.



Properties

Blocking

```
VARIANT_BOOL Blocking
```

Description

Determines whether methods should block until completion or return immediately for asynchronous operation.

Acceptable values

- VARIANT_TRUE (C++) or True (Visual Basic .NET). The ActiveX is forced to block or wait until a method completes before it returns control to the caller.
- VARIANT_FALSE (C++) or False (Visual Basic .NET). Returns control to the application immediately, and the caller should handle the events accordingly.

Default value

VARIANT_FALSE or False

Blocking affects some methods differently. See each method's description for the effect. Unless otherwise noted:

- In non-blocking mode (Block = False), a method:
  - Starts another thread of execution to perform the given method, returning control to the application immediately.
  - Returns 0 on launching new thread successfully; otherwise returns nonzero.
  - If the method is successful, an event indicating completion is fired; if unsuccessful, an Error event is fired.
- In blocking mode (Block = True), a method:
  - Is executed.
  - Returns 0 if it completes successfully; returns nonzero otherwise.
- Error message can be reviewed by calling GetLastError().

Visual C++ example

```
//set the Centrifuge in blocking mode
VARIANT_BOOL blocking=VARIANT_TRUE;
m_Centrifuge.PutBlocking(blocking);
//set the Centrifuge in non-blocking mode
blocking=VARIANT_FALSE;
m_Centrifuge.PutBlocking(blocking);
//returns the blocking value
blocking=m_Centrifuge.GetBlocking();
//user should handle events if non-blocking mode is
selected!
```

Visual Basic .NET example

```
'set Centrifuge in blocking mode
Centrifuge1.Blocking=True
'set Centrifuge in non-blocking mode
Centrifuge1.Blocking=False
'returns the blocking value
Dim bMode as Boolean
bMode= Centrifuge1.Blocking
'user should handle events if non-blocking mode is
selected!
```

ControlPicture

```
IPictureDisp*ControlPicture
```

Description

A read-only picture of the Microplate Centrifuge that can be used in the container's application.

Parameters

None

Visual C++ example

```
/*the CPicture class will be imported in to your project
When the ActiveX is installed*/
CButton button;
//create button
button.Create("Button", WS_CHILD | WS_VISIBLE | BS_BITMAP,
CRect(10, 10, 60, 60), pParentWnd/*pointer of parent
window*/, 1);
CPicture CentrifugePic;
//retrieve the picture
CentrifugePic=m_Centrifuge.GetControlPicture();
//paint the bitmap on to the button
button.SetBitmap((HBITMAP) CentrifugePic.GetHandle());
```

Visual Basic .NET example

```
Dim iPicture As System.Drawing.Image=
Centrifuge1.ControlPicture()
button.BackgroundImage=iPicture
```

Related topics

For information about...	See...
Microplate Centrifuge ActiveX methods	“Methods” on page 6
Microplate Centrifuge ActiveX events	“Events” on page 14
Overview of ActiveX controls	“About ActiveX controls” on page 2
Reporting problems	“Reporting problems” on page iv

Methods

Abort

```
LONG Abort( )
```

Description

Aborts a current task that is in the error state and clears the error.

Parameters

None

Returns

0 if successful; Other value if there was an error

Visual C++ example

```
LONG lres=m_Centrifuge.Abort();
```

Visual Basic .NET example

```
Dim ires as Integer
ires=Centrifuge1.Abort()
```

AboutBox

```
void AboutBox( )
```

Description

Displays the Microplate Centrifuge About dialog box that contains the ActiveX and hardware version numbers.

Parameters

None

Returns

None

Visual C++ example

```
m_Centrifuge.AboutBox();
```

Visual Basic .NET example

```
Centrifuge1.AboutBox()
```

Close

```
LONG Close( )
```

Description

Disconnects from the Microplate Centrifuge. After successful closing, the CloseComplete event is sent.

Parameters

None

Returns

0 if successful; Other value if there was an error

Visual C++ example

```
LONG lres=m_Centrifuge1.Close();
```

Visual Basic .NET example

```
Dim ires as Integer
ires=Centrifuge1.Close()
```

CloseDoor

```
LONG CloseDoor( )
```

Description

Closes the Microplate Centrifuge's door. After successful closing of the door, the CloseDoorComplete event is sent.

Parameters

None

Returns

0 if successful; Other value if there was an error

Visual C++ example

```
LONG lres=m_Centrifuge.CloseDoor();
```

Visual Basic .NET example

```
Dim ires as Integer
ires=Centrifuge1.CloseDoor()
```

EnumerateProfiles

```
VARIANT EnumerateProfiles( )
```

Description

Retrieves a list of defined profiles. The strings in this array are the profile names that should be used for the Initialize method.

Parameters

None

Returns

An array of profile names

Visual C++ example

```
VARIANT vProfiles=m_Centrifuge1.EnumerateProfiles();
SAFEARRAY *psa=vProfiles.parray;
BSTR *bstrArray;
if
(FAILED(SafeArrayAccessData(psa,reinterpret_cast<void**>(
&bstrArray))))
{
     VariantClear(&vProfiles);
     return;
}
for(ULONG i=0;i<psa->rgsabound[0].cElements;i++)
{
     MessageBox(CString(bstrArray[i]));
}
SafeArrayUnaccessData(psa);
VariantClear(&vProfiles);
```

Visual Basic .NET example

```
Dim i as Integer
Dim sProfiles() As String
sProfiles=Centrifuge1.EnumerateProfiles()
For i=0 To sProfiles.GetLength(0)-1
 MsgBox sProfiles(i)
Next
```

GetActiveXVersion

```
BSTR GetActiveXVersion( )
```

Description

Retrieves the Microplate Centrifuge's ActiveX version number.

Parameters

None

Returns

ActiveX version number (string)

Visual C++ example

```
CString ActiveXVer=m_Centrifuge1.GetActiveXVersion();
```

Visual Basic .NET example

```
Dim sVersion As String
sVersion=Centrifuge1.GetActiveXVersion()
```

GetHardwareVersion

```
BSTR GetHardwareVersion( )
```

Description

Retrieves the Microplate Centrifuge's hardware version number. The hardware version is only available after profile is initialized.

Parameters

None

Returns

Hardware version number (string)

Visual C++ example

```
CString HardwareVer=m_Centrifuge1.GetHardwareXVersion();
```

Visual Basic .NET example

```
Dim sHardwareVersion As String
sHardwareVersion= Centrifuge1.GetHardwareVersion()
```

GetLastError

```
BSTR GetLastError( )
```

Description

Retrieves the last known error condition.

Parameters

None

Returns

An error string.

Visual C++ example

```
CString str=m_Centrifuge1.GetLastError();
```

Visual Basic .NET example

```
Dim sError As String=""
sError=Centrifuge1.GetLastError()
```

Ignore

```
LONG Ignore( )
```

Description

Ignores the previously issued error and moves to the next step in the task. This is not a recommended course of action, as the errors are issued for a reason. However, ignoring some errors can be appropriate if the operator understands the implications.

Parameters

None

Returns

0 if successful; Other value if there was an error

Visual C++ example

```
LONG lres=m_Centrifuge1.Ignore();
```

Visual Basic .NET example

```
Dim ires as Integer
ires=Centrifuge1.Ignore()
```

Initialize

```
LONG Initialize(BSTR Profile)
```

Description

Initializes the profile and starts communication with the Microplate Centrifuge using the parameters set in the profile. The profile specifies the serial connection used to communicate with the Microplate Centrifuge. The parameters for each profile can be adjusted in the Diagnostics dialog box (by a call to the ShowDiagsDialog method) on the Profiles page.

Parameters

Name	Type	Description
Profile	BSTR	The name of the profile to be used for initialization

Returns

0 if successful, and initiates the InitializeComplete event; Other value if there was an error

Visual C++ example

```
//connect via serial connection specified in the profile
LONG lres=m_Centrifuge.Initialize(_bstr_t("Centrifuge
profile"));
```

Visual Basic .NET example

```
'connect via serial connection specified in the profile
Dim ires as Integer
ires=Centrifuge1.Initialize("Centrifuge profile")
```

OpenDoor

```
LONG OpenDoor(SHORT bucket_num)
```

Description

Opens the door to the bucket_num. After successful opening of the door, the OpenDoorComplete event is sent.

Parameters

Name	Type	Range	Description
bucket_num	SHORT	1-2	Presented bucket

Returns

0 if successful; Other value if there was an error

Visual C++ example

```
LONG lres=m_Centrifuge.OpenDoor(1);
```

Visual Basic .NET example

```
Dim ires as Integer
ires=Centrifuge1.OpenDoor(1)
```

Retry

```
LONG Retry( )
```

Description

Retries the last action after an error occurred.

Parameters

None

Returns

0 if successful; Other value if there was an error

Visual C++ example

```
LONG lres=m_Centrifuge1.Retry();
```

Visual Basic .NET example

```
Dim ires as Integer
ires=Centrifuge1.Retry()
```

ShowDiagsDialog

```
void ShowDiagsDialog (VARIANT_BOOL modal, SHORT
securityLevel)
```

Description

Displays the Diagnostics dialog box that allows the operator to troubleshoot and correct problems. This method can be called before the Initialize method to create a profile. Contents displayed are based on the operator's access level.

Parameters

Name	Type	Range	Description
modal	VARIANT_BOOL	VARIANT_TRUE/ VARIANT_FALSE	VARIANT_TRUE = display dialog box as modal (does not permit users to access the parent window) VARIANT_FALSE = display dialog box as modeless (permits users to access the parent window)

Name	Type	Range	Description
securityLevel	SHORT	-1, 0, 1, 2, 3	The security level the operator has in the dialog box: 0 = Administrator 1 = Technician 2 = Operator 3 = Guest -1 = No access

Returns

None

Visual C++ example

```
m_Centrifuge.ShowDiagsDialog(VARIANT_TRUE,0);
```

Visual Basic .NET example

```
Centrifuge1.ShowDiagsDialog(True,0)
```

SpinCycle

```
LONG SpinCycle(DOUBLE vel_percent, DOUBLE accel_percent,
DOUBLE decel_percent, SHORT timer_mode, LONG time, SHORT
bucket_num);
```

Description

Commands Microplate Centrifuge to perform a spin cycle. After successful spinning cycle, the SpinCycleComplete event is sent.

Parameters

Name	Type	Range	Description
vel_percent	DOUBLE	1-100	Percentage of maximum velocity to run
accel_percent	DOUBLE	1-100	Percentage of maximum acceleration to
decel_percent	DOUBLE	1-100	Percentage of max deceleration to run
timer_mode	SHORT	0-2	• When TIMER_MODE_TIME_TOTAL=0, the entire Centrifuge cycle takes *time* seconds. • When TIMER_MODE_TIME_AT_SPEED=1, the Microplate Centrifuge spends *time* seconds at full speed. • When TIMER_MODE_CONTINUOUS__SPIN =2, the Microplate Centrifuge will spin continuously until StopSpinCycle method is called.
time	LONG	1-86400	Number of seconds to spin

Name	Type	Range	Description
bucket_num	SHORT	1-2	Which bucket to present after spin cycle finishes

Returns

0 if successful; Other value if there was an error

Visual C++ example

```
/*Spin Centrifuge at 55% velocity, 60% acceleration, 90% braking for 10 seconds at desired speed. Present bucket #1 upon completion. */
LONG lres=m_Centrifuge1.SpinCycle(55.0,60.0,90.0,1,10,1);
```

Visual Basic .NET example

```
Dim ires as Integer
'Spin Centrifuge at 55% velocity, 60% acceleration, 90%
'braking for 10 seconds at desired speed. Present bucket
'#1 upon completion.
ires=Centrifuge1.SpinCycle(55.0,60.0,90.0,1,10,1);
```

StopSpinCycle

```
LONG StopSpinCycle(SHORT bucket_num);
```

Description

Stops currently running spin cycle and opens the door to the bucket_num. After the spin cycle has been successfully stopped and the door is opened, the StopSpinCycleComplete event is sent.

Parameters

Name	Type	Range	Description
bucket_num	SHORT	1-2	Presented bucket

Returns

0 if successful; Other value if there was an error

Visual C++ example

```
LONG lres=m_Centrifuge.StopSpinCycle(1);
```

Visual Basic .NET example

```
Dim ires as Integer
ires=Centrifuge1.StopSpinCycle (1)
```

Related topics

Microplate Centrifuge ActiveX properties	"Properties" on page 3

Microplate Centrifuge ActiveX events	"Events" on page 14
Overview of ActiveX controls	"About ActiveX controls" on page 2
Reporting problems	"Reporting problems" on page iv

Events

Error

```
void Error(SHORT Number, BSTR* Description, LONG Scode, BSTR
Source, BSTR HelpFile, LONG HelpContext, VARIANT_BOOL*
CancelDisplay)
```

Description

This event is sent when an error occurs during any non-blocking method execution.

Parameters

Name	Type	Range	Description
Description	BSTR*		The description of the error.
CancelDisplay	VARIANT_BOOL*	VARIANT_TRUE/ VARIANT_FALSE	The option to hide the error message. Use VARIANT_TRUE for C++ Use True for Visual Basic .NET

Note: SHORT Number, LONG Scode, BSTR Source, BSTR HelpFile, and LONG HelpContext are not used.

Returns

None

InitializeComplete

```
void InitializeComplete( )
```

Description

This event occurs when the Initialize method is successful.

Parameters

None

Returns

None

CloseComplete

```
void CloseComplete()
```

Description

This event occurs when the Close method is successful.

Parameters

None

Returns

None

CloseDoorComplete

```
void CloseDoorComplete()
```

Description

This event occurs when the CloseDoor method is successful.

Parameters

None

Returns

None

OpenDoorComplete

```
void OpenDoorComplete(SHORT bucket_num)
```

Description

This event occurs when the OpenDoor method is successful.

Parameters

Name	Type	Range	Description
bucket_num	SHORT	1-2	Presented bucket

Returns

None

SpinCycleComplete

```
void SpinCycleComplete(SHORT bucket_num)
```

Description

This event occurs when the SpinCycle method is successful.

Parameters

Name	Type	Range	Description
bucket_num	SHORT	1-2	Presented bucket

Returns

None

StopSpinCycleComplete

```
void StopSpinCycleComplete(SHORT bucket_num)
```

Description

This event occurs when the StopSpinCycle method is successful.

Parameters

Name	Type	Range	Description
bucket_num	SHORT	1-2	Presented bucket

Returns

None

Related topics

Microplate Centrifuge ActiveX methods	"Methods" on page 6
Microplate Centrifuge ActiveX properties	"Properties" on page 3
Overview of ActiveX controls	"About ActiveX controls" on page 2
Reporting problems	"Reporting problems" on page iv

Agilent Technologies

User Guide

G5405-90003

Revision 00, April 2012