EPSON

EPSON RC+ 5.0 Option Ver.5.3

VB Guide 5.0

Rev.2 EM097R1853F

EPSON RC+ 5.0 Option (Ver.5.3)

VB Guide 5.0

Rev.2

Copyright © 2006-2009 SEIKO EPSON CORPORATION. All rights reserved.

FOREWORD

Thank you for purchasing our robot products.
This manual contains the information necessary for the correct use of the Manipulator.
Please carefully read this manual and other related manuals before installing the robot system.
Keep this manual handy for easy access at all times.

WARRANTY

The robot and its optional parts are shipped to our customers only after being subjected to the strictest quality controls, tests, and inspections to certify its compliance with our high performance standards.

Product malfunctions resulting from normal handling or operation will be repaired free of charge during the normal warranty period. (Please ask your Regional Sales Office for warranty period information.)

However, customers will be charged for repairs in the following cases (even if they occur during the warranty period):

1. Damage or malfunction caused by improper use which is not described in the manual, or careless use.
2. Malfunctions caused by customers' unauthorized disassembly.
3. Damage due to improper adjustments or unauthorized repair attempts.
4. Damage caused by natural disasters such as earthquake, flood, etc.

Warnings, Cautions, Usage:

1. If the robot or associated equipment is used outside of the usage conditions and product specifications described in the manuals, this warranty is void.
2. If you do not follow the WARNINGS and CAUTIONS in this manual, we cannot be responsible for any malfunction or accident, even if the result is injury or death.
3. We cannot foresee all possible dangers and consequences. Therefore, this manual cannot warn the user of all possible hazards.

TRADEMARKS

Microsoft, Windows, and Windows logo are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other brand and product names are trademarks or registered trademarks of the respective holders.

TRADEMARK NOTATION IN THIS MANUAL

Microsoft® Windows® XP Operating system
Microsoft® Windows® Vista Operating system
Throughout this manual, Windows XP, and Windows Vista refer to above respective operating systems. In some cases, Windows refers generically to Windows XP, and Windows Vista.

NOTICE

No part of this manual may be copied or reproduced without authorization.
The contents of this manual are subject to change without notice.
Please notify us if you should find any errors in this manual or if you have any comments regarding its contents.

INQUIRIES

Contact the following service center for robot repairs, inspections or adjustments.
If service center information is not indicated below, please contact the supplier office for your region.

Please prepare the following items before you contact us.

- -Your controller model and its serial number
- -Your manipulator model and its serial number
- -Software and its version in your robot system
- -A description of the problem

SERVICE CENTER

MANUFACTURER & SUPPLIER

Japan & Others

SEIKO EPSON CORPORATION
Suwa Minami Plant
Factory Automation Systems Dept.
1010 Fujimi, Fujimi-machi,
Suwa-gun, Nagano, 399-0295
JAPAN
TEL : +81-(0)266-61-1802
FAX : +81-(0)266-61-1846

SUPPLIERS

North & South America

EPSON AMERICA, INC.
Factory Automation/Robotics
18300 Central Avenue
Carson, CA 90746
USA
TEL : +1-562-290-5900
FAX : +1-562-290-5999
E-MAIL : info@robots.epson.com

Europe

EPSON DEUTSCHLAND GmbH
Factory Automation Division
Otto-Hahn-Str.4
D-40670 Meerbusch
Germany
TEL : +49-(0)-2159-538-1391
FAX : +49-(0)-2159-538-3170
E-MAIL : robot.infos@epson.de

TABLE OF CONTENTS

1. Introduction 1
   1.1 Features 1

2. Installation 2
   2.1 Step by step instructions 2
   2.2 What's installed 2

3. Getting Started 3
   3.1 Getting started using Visual Basic 3
   3.2 Getting started using Visual C# 4
   3.2 Getting started using Visual C++ 5

4. Environments 6
   4.1 Design-Time Environment 6
      4.1.1 Development Startup 6
      4.1.2 Spel Class Initialization 6
      4.1.3 Development Cycle 6
   4.2 Production Environment 7
   4.2.1 Opening EPSON RC+ 5.0 at Runtime 7
   4.2.2 Using EPSON RC+ 5.0 Dialogs and Windows 7
   4.2.3 Installation on Target System 7

5. Executing Programs 8
   5.1 Executing SPEL+ Programs 8
   5.2 Aborting all tasks 8

6. Events 9
   6.1 Overview 9
   6.2 System Events 9
   6.3 User Events from SPEL+ 9

7. Error Handling 10
   7.1 Errors for Spel methods 10

8. Handling Pause and Continue 11
8.1 Pause state 11
8.2 Catching the Pause event 11
8.3 Executing Pause 11
8.4 Continue after pause 12
8.5 Abort after pause 12

9. Handling Emergency Stop 13
9.1 Using system EStop events 13

10. EPSON RC+ Windows and Dialogs 14
10.1 Windows 14
10.2 Dialogs 14

11. Displaying Video 15

12. SpelNetLib Reference 17
12.1 Spel Class 17
12.2 Spel Class Properties 17
12.3 Spel Class Methods 17
12.4 Spel Class Events 18
12.5 Spel Error Numbers and Messages 18
12.6 Commands 18
12.6.1 Connect Method, Spel Class 18
12.6.2 Continue Method, Spel Class 19
12.6.3 Disconnect Method, Spel Class 20
12.6.4 ErrorCode Property, Spel Class 20
12.6.5 ErrorOn Property, Spel Class 21
12.6.6 EStopOn Method, Spel Class 21
12.6.7 EventReceived Event, Spel Class 22
12.6.8 GetControllerInfo Method, Spel Class 24
12.6.9 GetErrorMessage Method, Spel Class 25
12.6.10 GetVar Method, Spel Class 26
12.6.11 HideWindow Method, Spel Class 27
12.6.12 Initialize Method, Spel Class 27
12.6.13 NoProjectSync Property, Spel Class 28
12.6.14 OperationMode Property, Spel Class 29
12.6.15 Pause Method, Spel Class 30
12.6.16 PauseOn Property, Spel Class 30

12.6.17 Project Property, Spel Class....31
12.6.18 Reset Method, Spel Class....31
12.6.19 RunDialog Method, Spel Class....32
12.6.20 SafetyOn Property, Spel Class....32
12.6.21 SetVar Method, Spel Class....33
12.6.22 ShowWindow Method, Spel Class....33
12.6.23 SpelVideoControl Property, Spel Class....34
12.6.24 Start Method, Spel Class....35
12.6.25 Stop Method, Spel Class....36
12.6.26 TeachPoint Method, Spel Class....37
12.6.27 Version Property, Spel Class....38
12.6.28 WarningCode Property, Spel Class....38
12.6.29 WarningOn Property, Spel Class....39
12.6.31 SpelVideo Control....39
12.6.31 SpelVideo Control Properties....39
12.6.32 SpelVideo Control Events....39
12.6.33 Camera Property, SPELVideo Control....40
12.6.34 GraphicsEnabled Property, SPELVideo Control....41
12.6.35 VideoEnabled Property, SPELVideo Control....42

1. Introduction

The EPSON RC+ VB Guide 5.0 Option enables you to use Microsoft Visual Basic or any other language that supports .NET technology to run your robotic applications. This gives you the power to create sophisticated user interfaces, use databases, and use third party products designed for use with VB.

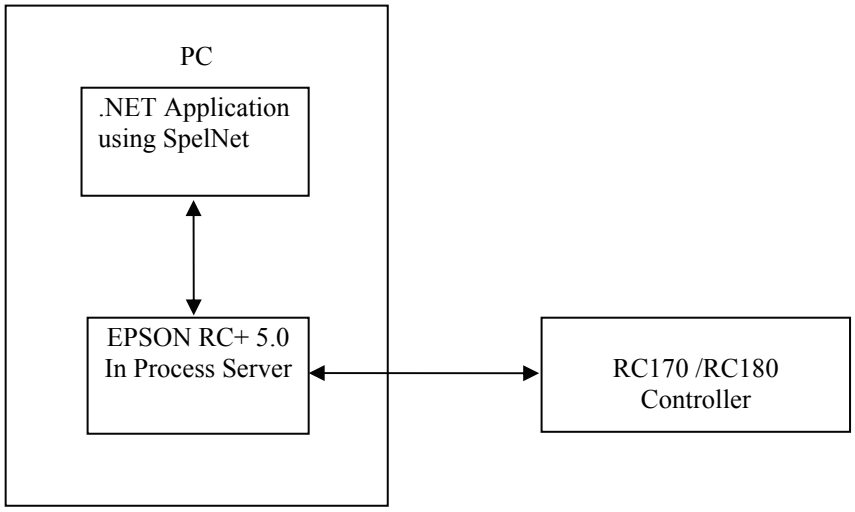
1.1 Features

The following features are supported in the VB Guide 5.0 package:

- SpelNetLib.dll library.
- Several EPSON RC+ windows and dialogs can be used by your VB application, including:
  - Robot Manager
  - Command window
  - IO monitor
  - Task manager
  - Controller Tools dialog
  - System configuration

During development, EPSON RC+ 5.0 can be run along with Visual Basic. At production time, EPSON RC+ 5.0 can be run invisibly in the background.

The figure below shows the basic structure of a system using VB Guide 5.0.



VB Guide 5.0 Basic Structure

EPSON RC+ 5.0 is an in-process server for the SpelNetLib library. Each instance of SpelNetLib loads an instance of EPSON RC+ 5.0 into the user application.

2. Installation

Please follow the instructions in this chapter to help ensure proper installation of the VB Guide 5.0 software.

Before starting, ensure that all Windows applications have been closed.

2.1 Step by step instructions

1. Install one of the Visual Studio 2005 Express versions, such as Visual Basic Express, or install Visual Studio 2005 .NET.
2. Install EPSON RC+ 5.0.
3. Ensure that the software key has been enabled for VB Guide 5.0 in the controller(s) you will be connecting to. Refer to the EPSON RC+ 5.0 User's Guide for information on how to enable options in the controller.

This completes the VB Guide 5.0 installation.

2.2 What's installed

The directories and files shown in the table below are installed on your PC during installation.

File	Description
\EPSONRC50\VBGUIDE\VB\DEMOS	Visual Basic .NET demonstrations
\EPSONRC50\VBGUIDE\VCS\DEMOS	Visual C# .NET demonstrations
\EPSONRC50\VBGUIDE\VC\DEMOS	Visual C++ .NET demonstrations
\EPSONRC50\PROJECTS\VBGuideDemos	EPSON RC+ 5.0 projects for demos
\EPSONRC50\EXE\SpelNetLib.dll	SpelNetLib Class library

3. Getting Started

This chapter contains information for getting started in the following development environments.

- Visual Basic .NET

- Visual C# .NET

- Visual C++ .NET

Demonstration programs are supplied with VB Guide 5.0. It is recommended that you go through the demonstrations to get familiar with the product.

3.1 Getting started using Visual Basic

To use SpelNetLib in a VB.NET project, declare a Spel Class object, as shown in the example below. *g_spel* can now be used in your project.

1. In Visual Studio .NET, select File | Project.
2. Create a Visual Basic project.
3. From the Project menu, select Add Reference.
4. In the NET Components tab, browse to the \EpsonRC50\Exe directory and select the SpelNetLib.dll file.
5. From the Project menu, create a new module and add the following code.

```
Module Module1

  Public WithEvents g_spel As SpelNetLib.Spel

  Public Sub InitApp()

    g_spel = New SpelNetLib.Spel

    With g_spel
      .Initialize
      .Project = "c:\EpsonRC50\projects\vbnet\vbnet.sprj"
    End With
  End Sub

  Public Sub EventReceived( _
          ByVal sender As Object, _
          ByVal e As SpelNetLib.SpelEventArgs) _
          Handles g_spel.EventReceived)

    MsgBox("received event " & e.Event)
  End Sub

End Module
```

3.2 Getting started using Visual C#

1. In Visual Studio .NET, select File | Project.
2. Create a C# project.
3. From the Project menu, select Add Reference.
4. Select the Browse tab and browse to the \EpsonRC50\Exe directory and select the SpelNetLib.dll file.
5. In the Form1 class, declare a Spel variable as shown below.

```
private SpelNetLib.Spel m_spel;
```

6. In the Form_Load event, add initialization code, as shown below.

```
private void Form1_Load(object sender, EventArgs e)
{
    m_spel = new SpelNetLib.Spel();
    m_spel.Initialize();

    m_spel.Project =
"c:\\EPSONRC50\\projects\\vcsnet\\vcsnet.sprj";

m_spel.EventReceived += new
   SpelNetLib.Spel.EventReceivedEventHandler(m_spel_EventR
eceived);
```

7. Add the event handler, as shown below.

```
public void m_spel_EventReceived(object sender,
   SpelNetLib.SpelEventArgs e)
{
}
```

3.3 Getting started using Visual C++

1. In Visual Studio .NET, select File | Project.
2. Create a C++ CLR Windows Forms Application project.
3. From the Project menu, select References
4. Click the Add New Reference button.
5. Select the Browse tab and browse to the \EpsonRC50\Exe directory and select the SpelNetLib.dll file.
6. In the Form1 class, declare a Spel variable as shown below.

```
private SpelNetLib::Spel^ m_spel;
```

7. In the Form_Load event, add initialization code, as shown below.

```
private System::Void Form1_Load(
    System::Object^ sender, System::EventArgs^ e)
{
   m_spel = gcnew SpelNetLib::Spel();
   m_spel->Initialize();
   m_spel->Project =
       "c:\\EPSONRC50\\projects\\vcnet\\vcnet.sprj";
   m_spel->EventReceived += gcnew
       SpelNetLib::Spel::EventReceivedEventHandler(
       this, &Form1::m_spel_EventReceived);
}
```

7. Add the event handler, as shown below.

```
private System::Void m_spel_EventReceived(
    System::Object^ sender, SpelNetLib::SpelEventArgs^ e)
{
   MessageBox::Show(e->Message);
}
```

4. Environments

4.1 Design-Time Environment

4.1.1 Development Startup

Typically, you would perform these steps to start development:

1. Declare a Spel class variable in a module in your VB project.
2. Start EPSON RC+ 5.0.
3. Configure EPSON RC+ 5.0 to communicate with one or more RC170/RC180 controllers.
4. Open the desired EPSON RC+ project or create a new EPSON RC+ project.
5. Build the EPSON RC+ project.
6. Add initialization code in VB for the SPEL class instance.
7. Run and debug the VB project.

4.1.2 Spel Class Initialization

After a new instance of the Spel class has been created, it needs to be initalized. When initialization occurs, the underlying EPSON RC+ 5.0 modules are loaded and initialized. Initialization is implicit with the first method call or property access. You can explicity initialize the class by calling the Initialize method.

```
m_spel.Initialize
```

If you only want to use the current project in the controller, you must set the NoProjectSync property to true.

```
m_spel.NoProjectSync = True
```

4.1.3 Development Cycle

Follow these basic steps to edit and run your VB code:

1. Stop the VB project.
2. Open EPSON RC+ 5.0.
3. Make changes in the EPSON RC+ 5.0 project.
4. Build the EPSON RC+ 5.0 project.
5. Switch to VB.
6. Run the VB project.

4.2 Production Environment

4.2.1 Opening EPSON RC+ 5.0 at Runtime

Decide if you want to allow the EPSON RC+ 5.0 environment to be opened from your application. This is especially useful for debugging. Set the **OperationMode** property to Program to put RC+ in Program Mode and open the RC+ GUI.

4.2.2 Using EPSON RC+ 5.0 Dialogs and Windows

At runtime, you can open and hide certain EPSON RC+ 5.0 windows from your VB application. You can also run certain EPSON RC+ 5.0 dialogs. See the chapter *EPSON RC+ Windows and Dialogs* for details.

4.2.3 Installation on Target System

You should make an installation program for your VB project by using a Visual Studio setup project. Then follow these steps to setup a target system for your VB application:

1. Install EPSON RC+ 5.0.
2. Install your EPSON RC+ 5.0 project. This is not needed if you plan to only run the project in the controller.
3. Install your VB application.

5. Executing Programs

5.1 Executing SPEL+ Programs

You can run any of the eight builtin main functions in the current controller project by using the *Start* method of the Spel class. The table below shows the program numbers and their corresponding function names in the SPEL+ project.

Program Number	SPEL+ Function Name
0	main
1	main1
2	main2
3	main3
4	main4
5	main5
6	main6
7	main7

Here is an example that starts function "main":

```
Sub btnStart_Click( _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnStart.Click)

    m_spel.Start(0) ' Starts function main
    btnStart.Enabled = False
    btnStop.Enabled = True
End Sub
```

5.2 Aborting all tasks

If you are running tasks and want to abort all tasks at once, you can use the *Stop* method of the Spel class.

For example:

```
Sub btnStop_Click( _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnStop.Click)
    m_spel.Stop
    btnStop.Enabled = False
    btnStart.Enabled = True
End Sub
```

6. Events

6.1 Overview

The Spel Class supports two types of events: system events and user events. System events are notifications of system status. User defined events are sent from any SPEL+ task to the VB application.

6.2 System Events

There are several system events that are sent to the VB application. Each system event indicates a change in status. There are events for Pause, Continue, Emergency Stop, etc. For complete details on all system events, see the description for *EventReceived*.

6.3 User Events from SPEL+

You can cause events to occur in your VB application from your SPEL+ programs. For example, you can let the VB application know information about a continuous cycle loop. This is a better method to use than polling for variable values from VB.

To fire an event to VB from SPEL+, use the SPELCom_Event command in a SPEL+ program statement. For example:

```
SPELCom_Event 1000, cycNum, lotNum, cycTime
```

The SPELCom_Event command is similar to a Print command. You can specify one or more pieces of data to be sent to VB. See EPSON RC+ 5.0 Help for details on SPELCom_Event.

Before you can receive events, you must declare your Spel class variable using the WithEvents clause.

```
Public WithEvents g_spel As SpelNetLib.Spel
```

Catch the event in the EventReceived routine for the Spel class instance. To edit this routine, in the module where the Spel class is declared select EventReceived from the procedure list.

Here is an example of code in the EventReceived routine that updates some labels when an event occurs.

```
Sub m_spel_EventReceived (ByVal sender As Object, _
            ByVal e As SpelNetLib.SpelEventArgs) _
            Handles m_spel.EventReceived
    Dim tokens() As String
    Select Case e.Event
        Case 2000
           tokens = e.Message.Split(New [Char]() {" "c}, _
             System.StringSplitOptions.RemoveEmptyEntries)
           lblCycCount.Text = tokens(0)
           lblLotNumber.Text = tokens(1)
           lblCycTime.Text = tokens(2)
    End Select
End Sub
```

7. Error Handling

7.1 Errors for Spel methods

When you execute a Spel class method, an exception is thrown if there are any errors.

When an error occurs, Spel throws it to the calling routine. You should use error handlers in your application to catch this error. In some cases, you will only want to display an error message. For example:

```
Sub btnStart_Click( _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnStart.Click)

  Try
    m_spel.Start(0)
  Catch ex As SpelNetLib.SpelException
    MsgBox(ex.Message)
  End Try
End Sub
```

You can examine the error number associated with the exeception by using the ErrorNumber property of SpelException.

```
  Try
    m_spel.Start(0)
  Catch ex As SpelNetLib.SpelException
    MsgBox(ex.ErrorNumber)
  End Try
```

8. Handling Pause and Continue

8.1 Pause state

When a pause occurs, SPEL+ tasks are in the pause state.

The controller is in the pause state after one of the following occurs while tasks are running:

- The Spel class Pause method was executed

- A SPEL+ task executed Pause.

- The safeguard was opened.

8.2 Catching the Pause event

The Spel class will signal your VB application that a pause has occurred.

You can catch the Pause event in the `EventReceived` event for the Spel class.

```
Sub m_spel_EventReceived (ByVal sender As Object, ByVal e As
SpelNetLib. SpelEventArgs) Handles m_spel.EventReceived
    Select Case EventNumber
        Case SpelNetLib.SpelEvents.Pause
            btnPause.Enabled = False
            btnContinue.Enabled = True
    End Select
End Sub
```

8.3 Executing Pause

The following routine shows how to issue a PAUSE from VB using the SPEL *Pause* method.

```
Sub btnPause_Click( _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnPause.Click)

    m_spel.Pause
    btnPause.Enabled = False
    btnContinue.Enabled = True
End Sub
```

8.4 Continue after pause

To continue after a pause has occurred, use the *Continue* method.

```
Sub btnContinue_Click( _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnContinue.Click)

    m_spel.Continue
    btnContinue.Enabled = False
    btnPause.Enabled = True
End Sub
```

8.5 Abort after pause

You can also execute the *Stop* method if you don't want to continue after a pause.

```
Sub btnStop_Click( _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnStop.Click)

    m_spel.Stop
    btnContinue.Enabled = False
    btnPause.Enabled = False
End Sub
```

9. Handling Emergency Stop

When an Emergency stop occurs, you may want to perform some specific action in your program, such as displaying a dialog, or a message box.

The Spel class issues two standard events for emergency stop status: EStopOn and EStopOff.

9.1 Using system EStop events

You can catch the system EStop events in the EventReceived handler in you VB application.

```
Imports SpelNetLib.Spel

Private Sub m_spel_EventReceived(ByVal sender As Object, ByVal
e As SpelEventArgs) Handles m_spel.EventReceived
    Select Case EventNumber
        Case SpelEvents.EstopOn
            MsgBox "E-Stop detected"
            gEStop = True
            lblEStop.BackColor = vbRed
            lblEStop.Text = "EStop ON"
        Case SpelEvents.EstopOn
            gEStop = False
            lblEStop.BackColor = vbGreen
            lblEStop.Text = "EStop OFF"
    End Select
End Sub
```

10. EPSON RC+ Windows and Dialogs

You can open certain EPSON RC+ windows and dialogs from VB using the ShowWindow and RunDialog methods of the Spel class.

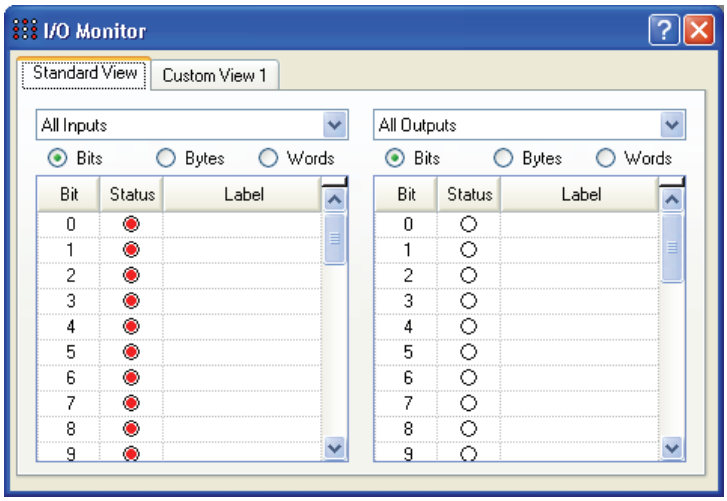
10.1 Windows

Windows are non-modal, meaning that they can remain open while other elements of your VB GUI can be used. You can show and hide EPSON RC+ windows from your VB program.

For example, to open and close the I/O Monitor window:

```
m_spel.ShowWindow(SpelNetLib.SpelWindows.IOMonitor)
m_spel.HideWindow(SpelNetLib.SpelWindows.IOMonitor)
```

WindowID	Window
SpelWindows.IOMonitor	IO Monitor
SpelWindows.TaskManager	Task Manager



I/O Monitor Window

10.2 Dialogs

Dialogs are modal: when a dialog is opened, other elements of your VB GUI cannot be used until the dialog is closed.

For example, to open the Robot Manager dialog:

```
m_spel.RunDialog(SpelNetLib.SpelDialogs.RobotManager)
```

Once a dialog has been opened, it must be closed by the operator. You cannot close a dialog from within your program. This is for safety reasons.

The following table shows the dialogs that can be opened.

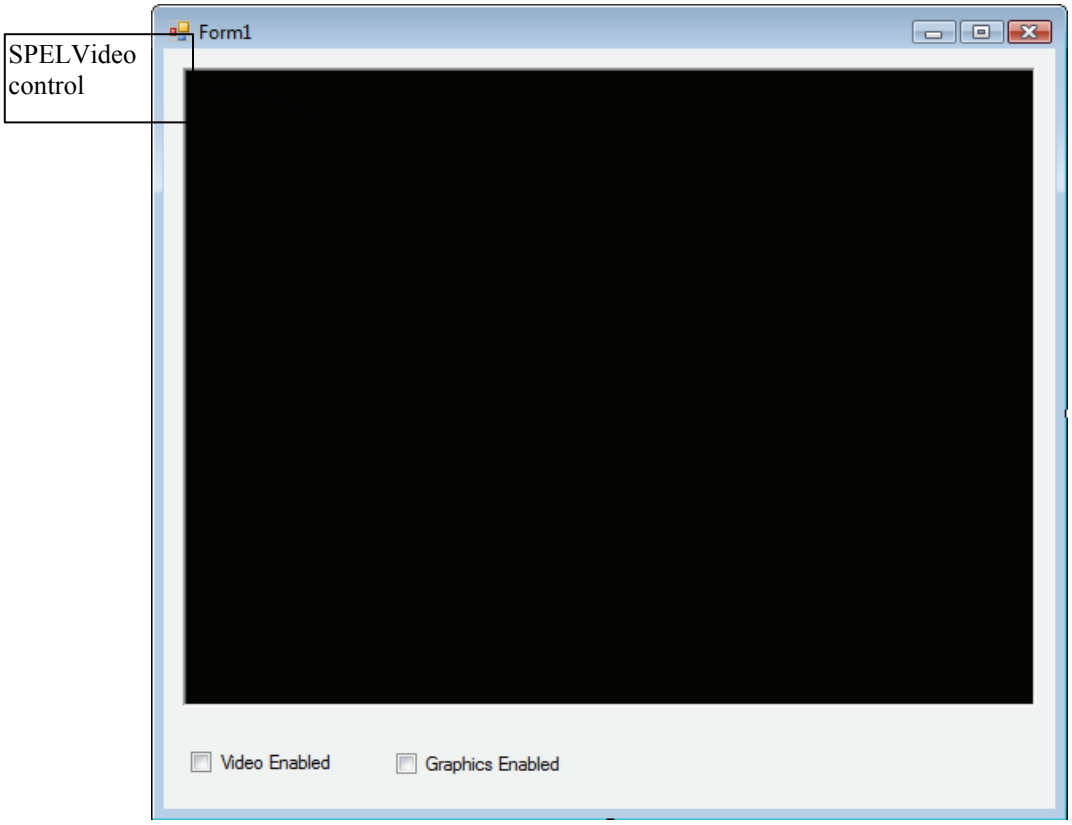
DialogID	Dialog
SpelDialogs.RobotManager	Robot Manager
SpelDialogs.ControllerTools	Controller Tools

11. Displaying Video

You can easily display a live vision window on a form in your application by using the SPELVideo control. When you run a vision sequence, the graphics can also be displayed on the window.

Perform the following steps to create a vision window:

1. Add the SpelVideo component to your project. To add the control to your VB toolbox, right click on the toolbox and select Choose Items. Select the Browse tab and browse to the \EpsonRC50\Exe directory and select the SpelNetLib.dll file. The SpelVideo control icon will be added to the toolbox.
2. Place a SpelVideo control on the form you want the video to be displayed. The control size can be changed up to the full size.
3. Set the VideoEnabled property to True.
4. Set the GraphicsEnabled property to True if you want to display vision graphics. You must also attach the SpelVideo control to a Spel class instance using the Spel class SpelVideoControl property.



SPELVideo control placed on a form

When the GraphicsEnabled property is True and the control is attached to a Spel class instance, then vision graphics will be displayed whenever the VRun method is executed on the controller connected to the Spel class instance.

If you use more than one SpelVideo control in your application, then you should disable the video on the non-active controls. There can only be one vision window displayed at one time.

Here is an example showing how to enable video and graphics on a VB form where a Spel class instance is used and a SpelVideo control have been placed:

```
Private Sub Form_Load(sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    m_spel = New Spel
    m_spel.Initialize()
    m_spel.Project = "c:\epsonrc50\projects\test\test.sprj"
    SpelVideo1.VideoEnabled = True
    SpelVideo1.GraphicsEnabled = True
    m_spel.SpelVideoControl = SPELVideo1
End Sub
```

12. SpelNetLib Reference

12.1 Spel Class

Description

This class allows you to execute commands and receive events from EPSON RC+ 5.0.

File Name

SpelNetLib.dll

12.2 Spel Class Properties

ErrorCode

ErrorOn

EstopOn

NoProjectSync

OperationMode

PauseOn

Project

Version

WarningCode

WarningOn

12.3 Spel Class Methods

Connect

Continue

Disconnect

GetControllerInfo

GetErrorMessage

HideWindow

Initialize

Pause

Reset

RunDialog

SafetyOn

ShowWindow

Start

Stop

TeachPoint

12.4 Spel Class Events

EventReceived

12.5 Spel Error Numbers and Messages

For error numbers and error messages, see the *SPEL+ Language Reference*.

12.6 Commands

12.6.1 Connect Method, Spel Class

Description
Connects the Spel class instance with a controller.

Syntax
Connect (*ConnectionNumber* As Integer)

Parameters

ConnectionNumber	Number representing a specific connection number as defined from the EPSON RC+ 5.0 Setup | Communications dialog. Connection #1 is always USB and cannot be changed.

Remarks

When a Spel class instance needs to communicate with a controller, it automatically connects to the last successful connection. If you want to connect to a specific controller via Ethernet, use the Connect method.

See Also
Disconnect, Initialize

Connect Example

```
Try
  m_spel.Connect(2)
Catch ex As SpelNetLib.SpelException
  MsgBox(ex.Message)
End Try
```

12.6.2 Continue Method, Spel Class

Description
Causes all tasks in the controller to resume if a pause has occurred.

Syntax
Continue

Remarks
Use **Continue** to resume all tasks that have been paused by the Pause method or by safeguard open.

When the safeguard is open while tasks are running, the robot will decelerate to a stop and the robotl motors will be turned off. After the safeguard has been closed, you can use **Continue** to resume the cycle.

See Also
Pause, Start, Stop

Continue Example

```
Sub btnContinue_Click( _
     ByVal sender As System.Object, _
     ByVal e As System.EventArgs) Handles btnContinue.Click)

    btnPause.Enabled = True
    btnContinue.Enabled = False
  Try
    m_spel.Continue
  Catch ex As SpelNetLib.SpelException
    MsgBox(ex.Message)
  End Try
End Sub
```

12.6.3 Disconnect Method, Spel Class

Description
Disconnects the Spel class instance from the current connection.

Syntax
Disconnect

Remarks

Use **Disconnect** to disconnect from the current controller connection.

See Also
Connect, Initialize

Disconnect Example

```
Try
  m_spel.Disconnect
Catch ex As SpelNetLib.SpelException
  MsgBox(ex.Message)
End Try
```

12.6.4 ErrorCode Property, Spel Class

Description

Returns the current controller error code.

Syntax
ErrorCode

Type

Integer

See Also

ErrorOn, EstopOn, SafetyOn, WarningOn

ErrorCode Example

```
If m_spel.ErrorOn Then
    lblErrorCode.Text = m_spel.ErrorCode.ToString()
Else
    lblErrorCode.Text = ""
End If
```

12.6.5 ErrorOn Property, Spel Class

Description

Returns status of the controller error state.

Syntax

ErrorOn

Type

Boolean

See Also

EstopOn, SafetyOn, WarningOn

ErrorOn Example

```
If m_spel.ErrorOn Then
    lblErrorStatus.Text = "ON"
Else
    lblErrorStatus.Text = "OFF"
End If
```

12.6.6 EStopOn Method, Spel Class

Description

Returns status of the controller's emergency stop status.

Syntax

EstopOn

Return Value

Boolean. True if the emergency stop is active, False if closed.

See Also

PauseOn, SafetyOn

EStopOn Example

```
If m_spel.EStopOn Then
    lblEStop.Text = "Emergency stop is active"
Else
    lblEStop.Text = ""
EndIf
```

12.6.7 EventReceived Event, Spel Class

Description

Occurs when EPSON RC+ 5.0 sends a system event or when a program running in SPEL$^+$ sends an event using a SPELCom_Event statement.

Syntax

EventReceived (ByVal *sender* As Object, ByVal *e* As SpelNetLib.SpelEventArgs)

Parameters

e.Event Number representing a specific user-defined event.

e.Message String containing event data.

Remarks

There are several system events that EPSON RC+ 5.0 issues. The following table describes them.

System Events

Event Number	Event Message	Constant	Description
1	"PAUSE"	SpelEvents.Pause	Occurs when tasks are paused.
2	"SAFE GUARD OPEN"	SpelEvents.SafeGuardOpen	Occurs when safe guard is open.
3	"SAFE GUARD CLOSE"	SpelEvents.SafeGuardClose	Occurs when safe guard is closed.
4	"ALL TASKS STOPPED"	SpelEvents.AllTasksStopped	Occurs when all tasks have been stopped.
5	"Error xxx!: mmm in task t at line yyy"	SpelEvents.Error	Occurs when a task is aborted due to an unhandled error or a system error is generated.
6	Text from print statement	SpelEvents.Print	Occurs when a Print statement executes from a SPEL$^+$ task.
7	"ESTOP ON"	SpelEvents.EStopOn	Occurs when emergency stop condition changes to ON.
8	"ESTOP OFF"	SpelEvents.EStopOff	Occurs when emergency stop condition changes to OFF.
9	"CONTINUE"	SpelEvents.Continue	Occurs after a Cont has been executed.

Handling Events

When **EventReceived** is called from the Spel class instance , it is waiting for the routine to finish. Therefore, you should never try to execute any VB Guide 5.0 commands from within the **EventReceived** routine. If you want to execute commands based on an event that occurred, set a flag in **EventReceived** that can be read by an event loop in your program.

For example, in your VB main form Load procedure, you can create an event loop that services events received from SPEL+. In the SPELCom_EventReceived routine, set global flags to indicate which events were received. Then, in your ServiceEvents routine, you can check which events occurred and issue the appropriate commands.

User Events

You can also send events from your SPEL+ programs to your Visual Basic application.

To display print event data

Add a TextBox to a form.

Each time the event is received, use the SelText property of the TextBox to append each message. See the example.

To send an event from SPEL+ to VB

Use the SPELCOM_EVENT command in a SPEL+ program statement. For example:

```
SPELCom_Event 10000, "Cycle aborted"
```

When this statement executes, the EventReceived routine will be called with the event number and message. See EPSON RC+ Help for details on SPELCOM_EVENT.

```
Sub m_spel_EventReceived ( _
         ByVal sender As Object, _
         ByVal e As SpelNetLib.SpelEventArgs) _
         Handles m_spel.EventReceived
    Redim tokens(0) As String
    Select Case e.Event
        Case 2000
            tokens = e.Message.Split(" ")
            lblCycCount = Val(tokens(0))
            lblLotNumber = Trim$(tokens(1))
            lblCycTime = Val(tokens(2))
    End Select
End Sub
```

12.6.8 GetControllerInfo Method, Spel Class

Description
Returns information about the current controller.

Syntax
GetControllerInfo

See Also
GetErrorMessage

Remarks
GetControllerInfo returns a new instance of the SpelControllerInfo class, which contains controller information properties.

GetControllerInfo Example

```
Dim info As SpelControllerInfo
Dim msg As String

info = m_spel.GetControllerInfo()
msg = "Project Name: " & info.ProjectName & vbCrLf _
    & "Project ID: " & info.ProjectID
MsgBox(msg)
```

12.6.9 GetErrorMessage Method, Spel Class

Description
Returns the error message for the specified error or warning code.

Syntax
GetErrorMessage (*ErrorCode* As Integer)

Parameters
ErrorCode The error code for which to return the associated error message.

See Also
GetControllerInfo

GetErrorMessage Example

```
Dim msg As String

If m_spel.ErrorOn Then
  msg = m_spel.GetErrorMessage(m_spel.ErrorCode)
  MsgBox(msg)
End If
```

12.6.10 GetVar Method, Spel Class

Description
Returns the value of a SPEL+ global preserve variable in the controller.

Syntax
GetVar(*VarName* As String)

Parameters

VarName The name of the SPEL+ global preserve variable.

Return Value

Returns the value whose data type is determined by the type of the SPEL+ variable.

Remarks

You can use GetVar to retrieve values of any global preserve variables in the controller's current project. Before you can retrieve values, the project must be sucessfully built.

See Also
SetVar

GetVar Example

In the SPEL+ project, the variable is declared:

```
Global Preserve Integer g_myIntVar
Global Preserve String g_myStringVar$
Function main
   ...
Fend
```

In the VB project:

Since g_myIntVar is declared as in integer, the VB variable used to retrieve the value of g_myInVar must be declared as an integer.

```
Dim myIntVar as Integer
Dim myStringVar as String

myIntVar = m_spel.GetVar("g_myIntVar")
myStringVar = m_spel.GetVar("g_myStringVar$")
```

12.6.11 HideWindow Method, Spel Class

Description
Hides an EPSON RC+ 5.0 window that was previously displayed with ShowWindow.

Syntax
HideWindow (*WindowID* As SpelWindows, *Parent* As Form)

Parameters

WindowID The ID of the EPSON RC+ 5.0 window to hide.

See Also
RunDialog, ShowWindow

HideWindow Example

```
Sub btnHideIOMonitor_Click _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnHideIOMonitor.Click)

    m_spel.HideWindow(SpelNetLib.SpelWindows.IOMonitor)
End Sub
```

12.6.12 Initialize Method, Spel Class

Description
Initializes the Spel class instance.

Syntax
Initialize

Remarks

Normally, the Spel class instance is automatically initialized when the first method has been executed. Initialization can take several seconds as EPSON RC+ loads into memory. So in some cases, you may want to call initialize first in your application during startup.

See Also
Connect, Disconnect

Initialize Example

```
m_spel.Initiialize
```

12.6.13 NoProjectSync Property, Spel Class

Description

Sets / returns whether the current project in the PC should be synchronized with the controller project.

Syntax

NoProjectSync

Type

Boolean

Default Value

False

Remarks

When NoProjectSync is set to False (default), then the Spel class ensures that the project on the PC is synchronized with the project on the controller.

When NoProjectSync is set to True, the Spel class does not check for any project on the PC and does not synchronize the PC project with the controller. This allows you to run programs in the controller without any project on the PC.

This property is not persistant. You must set it after creating a Spel class instance if you want to set it to True.

See Also

Start

Examples

```
m_spel.Initialize
m_spel.NoProjectSync = True
```

12.6.14 OperationMode Property, Spel Class

Description

Reads or sets the EPSON RC+ mode of operation.

Syntax

OperationMode

Type

SpelNetLib.SpelOperationMode

Remarks

When **OperationMode** is set to Program, the EPSON RC+ 5.0 GUI for the current instance of the Spel class is opened and the controller operation mode is set to Program. If the user closes the GUI, **OperationMode** is set to Auto. If **OperationMode** is set to Auto from VB, the GUI also closes.

OperationMode Example

```
Sub btnSpelProgramMode_Click _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnHideIOMonitor.Click)

  Try
    m_spel.OperationMode = _
                  SpelNetLib.SpelOperationMode.Program
    ' If you want to wait for the user to close the RC+
    ' GUI, you can wait here for OperationMode to change
    ' to Auto
    Do
      Application.DoEvents()
      System.Threading.Thread.Sleep(10)
    Loop Until m_spel.OperationMode = _
                   SpelNetLib.SpelOperationMode.Auto
  Catch ex As SpelNetLib.SpelException
    MsgBox(ex.Message)
  End Try
End If
```

12.6.15 Pause Method, Spel Class

Description
Causes all SPEL+ tasks in the controller to pause. If the robot is moving, it will immediately decelerate to a stop.

Syntax
Pause

See Also
Continue, EventReceived, Stop

Pause Example

```
Sub btnPause_Click()_
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnPause.Click)

    m_spel.Pause
    btnPause.Enabled = False
    btnContinue.Enabled = True
End Sub
```

12.6.16 PauseOn Property, Spel Class

Description
Returns status of the controller pause state.

Syntax
PauseOn

Type
Boolean

See Also
Continue, EstopOn, Pause, SafetyOn

PauseOn Example

```
If m_spel.PauseOn Then
    btnPause.Enabled = False
    btnContinue.Enabled = True
End If
```

12.6.17 Project Property, Spel Class

Description

Sets / returns the current EPSON RC+ 5.0 project.

Syntax

Project

Type

String

Default Value

Empty string.

Remarks

When setting the **Project**, you must supply the full path and name of the EPSON RC+ 5.0 project make file. The make file is the project name with a .SPRJ extension.

Project Example

```
m_spel.Project = "c:\epsonrc\projects\myapp\myapp.sprj"
```

12.6.18 Reset Method, Spel Class

Description

Resets the controller to the initialized state.

Syntax

Reset

Reset Example

```
m_spel.Reset
```

12.6.19 RunDialog Method, Spel Class

Description
Runs an EPSON RC+ 5.0 dialog.

Syntax
RunDialog (*DialogID* As SpelDialogs)

Parameters

DialogID The ID of the EPSON RC+ 5.0 dialog to run.

RunDialog Example

```
Sub btnRobotManager_Click( _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnRobotManager.Click)

    m_spel.RunDialog(SpelDialogs.RobotManager)
End Sub
```

12.6.20 SafetyOn Property, Spel Class

Description
Returns status of the controller's safeguard input.

Syntax
SafetyOn

Type
Boolean

See Also
ErrorOn, EstopOn, PauseOn, WarningOn

Remarks

Use the SafetyOn property to obtain the safeguard status when your application starts, then use the SafeguardOn Event to update the status.

SafetyOn Example

```
If m_spel.SafetyOn Then
    lblSafeguard.Text = "Safe guard is active"
Else
    lblSafeguard.Text = ""
EndIf
```

12.6.21 SetVar Method, Spel Class

Description
Sets the value of a SPEL+ global preserve variable.

Syntax
SetVar(*VarName* As String, *Value* As Object)

Parameters

VarName	The name of the SPEL+ global preserve variable.
Value	The new value.

See Also
GetVar

SetVar Example

```
m_spel.SetVar("g_myIntVar", 123)
```

12.6.22 ShowWindow Method, Spel Class

Description
Shows an EPSON RC+ 5.0 window.

Syntax
ShowWindow (*WindowID* As SpelWindows, *Parent* As Form)

Parameters

WindowID	The ID of the EPSON RC+ 5.0 window to show.

See Also
HideWindow, RunDialog

ShowWindow Example

```
Sub btnShowIOMonitor_Click( _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnShowIOMonitor.Click)

    m_spel.ShowWindow(SpelNetLib.SpelWindows.IOMonitor)
End Sub
```

12.6.23 SpelVideoControl Property, Spel Class

Description
Used to connect a SpelVideo control to the Spel class instance so that video and graphics can be displayed.

Syntax
SpelVideoControl

See Also
Graphics Enabled, VideoEnabled

SpelVideoControl Example

```
m_spel.SpelVideoControl = SpelVideo1
```

12.6.24 Start Method, Spel Class

Description
Start one SPEL+ program.

Syntax
Start (*ProgramNumber* As Integer)

Parameters

ProgramNumber The program number to start, corresponding to the eight built-in main functions in SPEL+ as shown in the table below. The range is 0 to 7.

Program Number	SPEL+ Function Name
0	main
1	main1
2	main2
3	main3
4	main4
5	main5
6	main6
7	main7

Remarks
When **Start** is executed, control will return immediately to the calling program. You cannot start a program that is already running.

See Also
Continue, Pause, Stop

Start Example

```
Sub btnStart_Click( _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnStart.Click)

    m_spel.Start(0)
End Sub
```

12.6.25 Stop Method, Spel Class

Description

Stops all SPEL+ tasks running in the controller.

Syntax

Stop

See Also

Continue, Pause, Start

Stop Example

```
Sub btnStop_Click( _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnStop.Click)

    m_spel.Stop
End Sub
```

12.6.26 TeachPoint Method, Spel Class

Description
Runs an a dialog that allows an operator to jog and teach one point.

Syntax
TeachPoint (*PointFile* As String, *PointNumber* As Integer, *Prompt* As String)

Parameters

PointFile	A string containing the name of the point file.
PointNumber	The point number to teach.
Prompt	A string containing the instructional text that is displayed on the bottom of the teach dialog.

Return Value

Boolean. Returns True if the operator clicked the Teach button, False if the operator clicked Cancel.

Remarks

Use TeachPoints to allow an operator to teach one robot point in the controller. When TeachPoints is executed, the point file is loaded from the controller. When the Teach button is clicked, the point is taught in the controller and the point file is saved on the controller.

TeachPoint Example

```
Sub btnTeachPick_Click( _
       ByVal sender As System.Object, _
       ByVal e As System.EventArgs) _
       Handles btnTeachPick.Click)

    Dim sts As Boolean
    Dim prompt As String

    prompt = "Jog to Pick position and click Teach"
    sts = m_spel.TeachPoint("points.pts", 1, prompt)

End Sub
```

12.6.27 Version Property, Spel Class

Description
Returns the current EPSON RC+ 5.0 software version.

Syntax
Version

Type
String

Version Example

```
' Get version of software
curVer = m_spel.Version
```

12.6.28 WarningCode Property, Spel Class

Description
Returns controller warning code.

Syntax
WarningCode

Type
Integer

See Also
ErrorCode, ErrorOn, EstopOn, SafetyOn, WarningOn

WarningCode Example

```
If m_spel.WarningOn Then
    lblWarningCode.Text = m_spel.WarningCode.ToString()
Else
    lblWarningCode.Text = ""
End If
```

12.6.29 WarningOn Property, Spel Class

Description

Returns status of the controller warning state.

Syntax

WarningOn

Type

Boolean

See Also

ErrorOn, EstopOn, PauseOn, SafetyOn

WarningOn Example

```
If m_spel.WarningOn Then
    lblWarningStatus.Text = "ON"
Else
    lblWarningStatus.Text = "OFF"
End If
```

12.6.30 SpelVideo Control

Description

This control allows you to display video from an EPSON Smart Camera.

File Name

SpelNetLib.dll

12.6.31 SpelVideo Control Properties

This control supports the properties listed below in addition to all standard .NET component properties. They are documented in this manual. See the Visual Basic on-line Help for documentation on the standard properties.

GraphicsEnabled

VideoEnabled

12.6.32 SpelVideo Control Events

All of the events for this control are standard .NET events. See the Visual Basic on-line Help for details

12.6.33 Camera Property, SPELVideo Control

Description

Sets/gets the camera number to display video from. This is useful when you want to display video during jogging operations, live video monitoring, etc. If you are using the control to display graphics for vision sequences, then when the sequence is run, the camera number for the sequence will be used instead of this property value.

Syntax

Camera

Type

Integer

Default Value

0 – any camera is displayed

See Also

VideoEnabled, GraphicsEnabled

Examples

```
SpelVideo1.Camera = 1
```

12.6.34 GraphicsEnabled Property, SPELVideo Control

Description

Determines whether vision graphics are displayed after a sequence is run. In order to see graphics, you must attach the control to a Spel class instance using the Spel VideoControl property. This property can be set "on the fly" so that graphics can be turned on/off while sequences are being run.

Syntax

GraphicsEnabled

Type

Boolean

Default Value

False

See Also

Camera, VideoEnabled

Examples

```
SpelVideo1.GraphicsEnabled = True
```

12.6.35 VideoEnabled Property, SPELVideo Control

Description

Determines whether video is displayed.

.

Syntax

VideoEnabled

Type

Boolean

Default Value

False

See Also

Camera, GraphicsEnabled

Examples

```
SpelVideo1.VideoEnabled = True
```