XRC Controller

Playback Box

OCX Manual

Part Number 145012-1

July 25, 2000

MOTOMAN
805 Liberty Lane
West Carrollton, OH 45449
TEL: (937) 847-6200 FAX: (937) 847-6277
24-HOUR SERVICE HOTLINE: (937) 847-3200

The information contained within this document is the proprietary property of Motoman, Inc., and may not be copied, reproduced or transmitted to other parties without the expressed written authorization of Motoman, Inc.

‘2000 by MOTOMAN
All Rights Reserved

Because we are constantly improving our products, we reserve the right to change specifications without notice. MOTOMAN is a registered trademark of YASKAWA Electric Manufacturing.
# Contents

## Introduction

Chapter 1:

- About this Document ................................................................. 1-1
- System Requirements ................................................................. 1-1
- Reference to Other Documentation ............................................. 1-1
- Customer Service ......................................................................... 1-1

## Safety

Chapter 2:

- Standard Conventions ............................................................... 2-1
- General Safeguarding Tips ......................................................... 2-2
- Mechanical Safety Devices ......................................................... 2-2
- Installation Safety ........................................................................ 2-2
- Programming Safety ..................................................................... 2-3
- Operation Safety .......................................................................... 2-3
- Maintenance Safety ...................................................................... 2-4

## Operation & Usage

Chapter 3:

- PCXRC Control Panel ............................................................... 3-1
- Example Container Using
  - Visual Basic® 6.0 ...................................................................... 3-2
  - Adding PBBOX to a Form ......................................................... 3-2
  - Resizing the Control Panel ....................................................... 3-3
  - Method Syntax Example ........................................................... 3-3
- Example Container Using Visual C++® 6.0® ............................... 3-4
  - Adding PBBOX to a Visual C++® dialog .................................... 3-4
  - Resizing the Control Panel ....................................................... 3-5
  - Method Syntax Example ........................................................... 3-6
- Methods ....................................................................................... 3-8
- PBBOX Properties ....................................................................... 3-11
Chapter 1: Introduction

PBBOX control is an ActiveX® playback panel control for the Motoman PCXRC robot. This ActiveX® control contains the standard playback panel controls for the PCXRC robot. The PBBOX control enables local control of: servo power, alarm reset, operation mode, cycle, and job control from a Windows® based application. Local control is defined as PBBOX running on the PCXRC being controlled.

About this Document

This manual is intended as an introduction and overview for personnel who are familiar with the operation of their Motoman PCXRC robot model and Windows® based application development environments such as Visual Basic® and/or Visual C++® that incorporate ActiveX® controls. This manual contains the following chapters:

Chapter 1: Introduction
Provides general information about PBBOX control, a list of reference documents, and customer service information.

Chapter 2: Safety
Provides information regarding the safe use and operation of your robotic system.

Chapter 4: Operation
Provides instructions for usage of the PBBOX control with a Visual Basic® and/or Visual C++® Windows® application (Control Container). This includes syntax, use of methods and functions, and control resizing by dragging and properties.

System Requirements
- Motoman robot and PCXRC controller
- Visual Basic® (version 6 or higher), or Visual C++® (version 6 or higher)

Note: This manual assumes that the user is familiar with Windows® programming, robot programming, and ActiveX® COM-based technologies.

Reference to Other Documentation
For additional information, refer to the following:
- Manipulator Manual for your robot model
- Vendor manuals for system components not manufactured by Motoman

Customer Service
If you need technical assistance, contact the Motoman service staff at (937) 847-3200. Please have the following information ready before you call:
- System configuration (hard disk capacity, memory, software, etc.)
- Description of difficulty (make a note of any error message)
Chapter 2: Safety

It is the purchaser's responsibility to ensure that all local, county, state, and national codes, regulations, rules, or laws relating to safety and safe operating conditions for each installation are met and followed.

We suggest that you obtain and review a copy of the ANSI/RIA National Safety Standard for Industrial Robots and Robot Systems. This information can be obtained from the Robotic Industries Association by requesting ANSI/RIA R15.06. The address is as follows:

Robotic Industries Association
900 Victors Way
P.O. Box 3724
Ann Arbor, Michigan 48106
TEL: (734) 994-6088
FAX: (734) 994-3338

Ultimately, the best safeguard is trained personnel. The user is responsible for providing personnel who are adequately trained to operate, program, and maintain the robot cell. The robot must not be operated by personnel who have not been trained!

We recommend that all personnel who intend to operate, program, repair, or use the robot system be trained in an approved Motoman training course and become familiar with the proper operation of the system.

This safety section addresses the following:
- Standard Conventions
- General Safeguarding Tips
- Mechanical Safety Devices
- Installation Safety
- Programming Safety
- Operation Safety
- Maintenance Safety

Standard Conventions

This manual includes information essential to the safety of personnel and equipment. As you read through this manual, be alert to the four signal words:
- DANGER
- WARNING
- CAUTION
- NOTE

Pay particular attention to the information provided under these headings which are defined below (in descending order of severity).

⚠️ **DANGER!**
Information appearing under the DANGER! caption concerns the protection of personnel from the immediate and imminent hazards that, if not avoided, will result in immediate, serious personal injury or loss of life in addition to equipment damage.

⚠️ **WARNING!**
Information appearing under the WARNING! caption concerns the protection of personnel and equipment from potential hazards that can result in personal injury or loss of life in addition to equipment damage.
CAUTION!
Information appearing under the CAUTION! caption concerns the protection of personnel, equipment, software, and data from hazards that can result in minor personal injury or equipment damage.

NOTE: Information appearing in a NOTE caption provides additional information which is helpful in understanding the item being explained.

General Safeguarding Tips
All operators, programmers, plant and tooling engineers, maintenance personnel, supervisors, and anyone working near the robot must become familiar with the operation of this equipment. All personnel involved with the operation of the equipment must understand potential dangers of operation. General safeguarding tips are as follows:

- Improper operation can result in personal injury and/or damage to the equipment. Only trained personnel familiar with the operation of this robot, the operator’s manuals, the system equipment, and options and accessories should be permitted to operate this robot system.
- Do not enter the robot cell while it is in automatic operation. Programmers must have the teach pendant when they enter the robot cell.
- Improper connections can damage the robot. All connections must be made within the standard voltage and current ratings of the robot I/O (Inputs and Outputs).
- The robot must be placed in Emergency Stop (E-STOP) mode whenever it is not in use.

- In accordance with ANSI/RIA R15.06, section 6.13.4 and 6.13.5, use lockout/tagout procedures during equipment maintenance. Refer also to Section 1910.147 (29 CFR, Part 1910), Occupational Safety and Health Standards for General Industry (OSHA).

Mechanical Safety Devices
The safe operation of the robot, positioner, auxiliary equipment, and system is ultimately the user’s responsibility. The conditions under which the equipment will be operated safely should be reviewed by the user. The user must be aware of the various national codes, ANSI/RIA R15.06 safety standards, and other local codes that may pertain to the installation and use of industrial equipment. Additional safety measures for personnel and equipment may be required depending on system installation, operation, and/or location. The following safety measures are available:

- Safety fences and barriers
- Light curtains
- Door interlocks
- Safety mats
- Floor markings
- Warning lights

Check all safety equipment frequently for proper operation. Repair or replace any non-functioning safety equipment immediately.

Installation Safety
Safe installation is essential for protection of people and equipment. The following suggestions are intended to supplement, but not replace, existing federal, local, and state laws and regulations. Additional safety measures for personnel and equipment may be required depending on system installation, operation, and/or location. Installation tips are as follows:
• Be sure that only qualified personnel familiar with national codes, local codes, and ANSI/RIA R15.06 safety standards are permitted to install the equipment.
• Identify the work envelope of each robot with floor markings, signs, and barriers.
• Position all controllers outside the robot work envelope.
• Whenever possible, install safety fences to protect against unauthorized entry into the work envelope.
• Eliminate areas where personnel might get trapped between a moving robot and other equipment (pinch points).
• Provide sufficient room inside the workcell to permit safe teaching and maintenance procedures.

Programming Safety

All operators, programmers, plant and tooling engineers, maintenance personnel, supervisors, and anyone working near the robot must become familiar with the operation of this equipment. All personnel involved with the operation of the equipment must understand potential dangers of operation. Programming tips are as follows:

• Any modifications to PART 1 of the controller PLC can cause severe personal injury or death, as well as damage to the robot! Do not make any modifications to PART 1. Making any changes without the written permission of Motoman will VOID YOUR WARRANTY!
• Some operations require standard passwords and some require special passwords. Special passwords are for Motoman use only. YOUR WARRANTY WILL BE VOID if you use these special passwords.
• Back up all programs and jobs onto a floppy disk whenever program changes are made. To avoid loss of information, programs, or jobs, a backup must always be made before any service procedures are done and before any changes are made to options, accessories, or equipment.

• The concurrent I/O (Input and Output) function allows the customer to modify the internal ladder inputs and outputs for maximum robot performance. Great care must be taken when making these modifications. Double-check all modifications under every mode of robot operation to ensure that you have not created hazards or dangerous situations that may damage the robot or other parts of the system.
• Improper operation can result in personal injury and/or damage to the equipment. Only trained personnel familiar with the operation, manuals, electrical design, and equipment interconnections of this robot should be permitted to operate the system.
• Inspect the robot and work envelope to be sure no potentially hazardous conditions exist. Be sure the area is clean and free of water, oil, debris, etc.
• Be sure that all safeguards are in place.
• Check the E-STOP button on the teach pendant for proper operation before programming.
• Carry the teach pendant with you when you enter the workcell.
• Be sure that only the person holding the teach pendant enters the workcell.
• Test any new or modified program at low speed for at least one full cycle.

Operation Safety

All operators, programmers, plant and tooling engineers, maintenance personnel, supervisors, and anyone working near the robot must become familiar with the operation of this equipment. All personnel involved with the operation of the equipment must understand potential dangers of operation. Operation tips are as follows:

• Be sure that only trained personnel familiar with the operation of this robot, the operator's manuals, the system equipment, and options and accessories are permitted to operate this robot system.
• Check all safety equipment for proper operation. Repair or replace any non-functioning safety equipment immediately.
• Inspect the robot and work envelope to ensure no potentially hazardous conditions exist. Be sure the area is clean and free of water, oil, debris, etc.
• Ensure that all safeguards are in place.
• Improper operation can result in personal injury and/or damage to the equipment. Only trained personnel familiar with the operation, manuals, electrical design, and equipment interconnections of this robot should be permitted to operate the system.
• Do not enter the robot cell while it is in automatic operation. Programmers must have the teach pendant when they enter the cell.
• The robot must be placed in Emergency Stop (E-STOP) mode whenever it is not in use.
• This equipment has multiple sources of electrical supply. Electrical interconnections are made between the controller, external servo box, and other equipment. Disconnect and lockout/tagout all electrical circuits before making any modifications or connections.
• All modifications made to the controller will change the way the robot operates and can cause severe personal injury or death, as well as damage the robot. This includes controller parameters, ladder parts 1 and 2, and I/O (Input and Output) modifications. Check and test all changes at slow speed.

Maintenance Safety
All operators, programmers, plant and tooling engineers, maintenance personnel, supervisors, and anyone working near the robot must become familiar with the operation of this equipment. All personnel involved with the operation of the equipment must understand potential dangers of operation. Maintenance tips are as follows:

• Do not perform any maintenance procedures before reading and understanding the proper procedures in the appropriate manual.
• Check all safety equipment for proper operation. Repair or replace any non-functioning safety equipment immediately.
• Improper operation can result in personal injury and/or damage to the equipment. Only trained personnel familiar with the operation, manuals, electrical design, and equipment interconnections of this robot should be permitted to operate the system.
• Back up all your programs and jobs onto a floppy disk whenever program changes are made. A backup must always be made before any servicing or changes are made to options, accessories, or equipment to avoid loss of information, programs, or jobs.
• Do not enter the robot cell while it is in automatic operation. Programmers must have the teach pendant when they enter the cell.
• The robot must be placed in Emergency Stop (E-STOP) mode whenever it is not in use.
• Be sure all safeguards are in place.
• Use proper replacement parts.
• This equipment has multiple sources of electrical supply. Electrical interconnections are made between the controller, external servo box, and other equipment. Disconnect and lockout/tagout all electrical circuits before making any modifications or connections.
• All modifications made to the controller will change the way the robot operates and can cause severe personal injury or death, as well as damage the robot. This includes controller parameters, ladder parts 1 and 2, and I/O (Input and Output) modifications. Check and test all changes at slow speed.
• Improper connections can damage the robot. All connections must be made within the standard voltage and current ratings of the robot I/O (Inputs and Outputs).
Chapter 3: Operation & Usage

This chapter provides information for using PBBOX control with Visual Basic® 6.0, or Visual C++® 6.0. Instructions for basic usage of the PBBOX control, including syntax, use of methods and functions, and resizing by dragging and properties are also included.

PCXRC Control Panel

The PBBOX ActiveX® control contains the primary system controls for the PCXRC robot. PBBOX enables control of your PCXRC robot from a Windows® based application (control container). Areas of control include: servo power, alarm reset, operation mode, cycle, and job control.

Note: To use PBBOX control, the PCXRC must be placed in “Remote” mode from the physical playback panel on the front door of controller, or from a hard wired signal into the “Remote” input.

Servo Power
The Servo Power OFF/ON buttons turn servo power ON or OFF while in PLAY mode only. These buttons do not operate in TEACH mode.

⚠️ CAUTION!

Never use the OFF button in place of a physically hardwired E-Stop button.

Alarm Reset
The ALARM RESET button is used to reset minor alarms and error messages.

Mode
The Mode buttons (TEACH and PLAY) set the robots mode of operation.

Note: Changing modes from PLAY to TEACH, during playback, causes the program to cease execution and servo power to turn off.

Cycle
The Cycle buttons (STEP, 1CYCLE, and AUTO) set the operating method for playback operations.

Hold
The HOLD button halts the operation of the manipulator. A decelerated stop is performed and a hold condition is maintained until the HOLD button is pressed again.

Start
Pressing the START button while in PLAY mode with servo power on, causes playback execution of the current job if there are no Alarm or Error conditions.
Example Container Using Visual Basic® 6.0

The following procedure demonstrates general usage of the PBBOX control and methods within a Visual Basic® standard project. This example illustrates how to use the PBBOX control and the proper syntax for using its methods.

Note: Visual Basic® 6.0 is used in these examples. However, these general steps can be used to add PBBOX control to your specific application.

Adding PBBOX to a Form
2. Start a new Standard EXE project.
3. From the Project Explorer window, select the [view object] icon. Form 1 is displayed.
4. From the main project window, select Components...
5. From the Components window, scroll down and check PBbox ActiveX® Control module. Select OK. The PBBOX icon appears in the toolbox.
6. Select the PBBOX icon from the toolbox.
7. Position the mouse on the form at the location you want to draw the control.
8. Hold down the mouse button and draw the outline of the PBBOX control on the form.
9. When you release the mouse button, the new PBBOX control appears at the location specified.

Resizing the Control Panel

The height and width of the PBBOX control panel can be modified using the Properties function, or by dragging the grips on the edges of the control itself.

Resizing using Properties

Note: Custom property methods can also be used to adjust size and shape of control panel during Run time (See "Methods" on page 8.)

1. Right click the PBBOX control panel and select Properties. The Property Pages window appears.
2. Enter the desired height and width and select OK.

Resizing by Dragging

1. Select the control panel. Eight grip points appear.
2. Click on any of the grips and while holding down the mouse button, drag the anchor to the desired size.
3. When you release the mouse button, the PBBOX control panel appears at the new size.

Method Syntax Example

1. Select the Command Button icon from the toolbox.
2. Position the mouse on the form at the location you want to draw the new button.
3. Hold down the mouse button and draw the outline of the button on the form.
4. When you release the mouse button, the new button appears at the location specified.
5. In the Properties window, modify the Caption name to: “Invoke Set Play Method.”
6. Double click the new “Invoke Set Play Method” button. The Project1 - Form1 (Code) dialog box appears containing the handler routine.
7. Type “Pbbox1.” at the cursor. The PBBOX methods list appears.
8. Scroll down and double click SetPlayMode. The new “Invoke Set Play Method” button is now fully operational and can be used to set the mode of operation for the PCXRC to PLAY.
9. Build and execute.

Example Container Using Visual C++ 6.0®

The following procedure demonstrates general usage of PBBOX control and methods within a Visual C++® Dialog based application. This example illustrates how to use the PBBOX control and the proper syntax for using its methods.

Note: Visual C++® 6.0 is used in these examples. However, these general steps can be used to add PBBOX control to your specific application.

Adding PBBOX to a Visual C++® dialog
4. Type a name for your project and click OK. MFC AppWizard - Step 1 appears.
5. Select Dialog Based and click Next.
6. In the Step 2 window, make certain ActiveX® Controls support is checked.
7. Step through the remaining windows and click Finish and OK.

**Inserting PBBOX into dialog**

1. From the main menu, select Project → Add to Project → Components and Controls... The Components and Controls Gallery appears.

2. Double click Registered ActiveX® Controls.

3. Select Pbbox Control and press Insert. Select OK at the "Insert this component?" dialog box.

4. The Confirm Classes dialog appears. Select OK. The PBBOX control icon appears in the Toolbox.

5. Select PBBOX icon from toolbox.

6. Position cursor on dialog and, holding down the left mouse button, draw the outline of the PBBOX control on the dialog.

7. When you release the mouse button, the new PBBOX control appears at the location specified.

**Resizing the Control Panel**

The height and width of the PBBOX control panel can be modified using the Properties function, or by dragging.

**Resizing using Properties**

*Note: Custom property methods can also be used to adjust size and shape of control panel during Run time (See “Methods” on page 8.)*

1. Right click the PBBOX ActiveX® control and select Properties.
2. Select the Control tab.
3. Enter desired height and width and select OK.
4. When you release the mouse button, the PBBOX control panel appears at the new size.

Resizing by Dragging
1. Select the control panel. Eight grip points appear.
2. Click on any of the grips and while holding down the mouse button, drag the anchor to the desired size.
3. When you release the mouse button, the PBBOX control panel appears at the new size.

Method Syntax Example
1. Select Command Button icon from toolbox.
2. Position the mouse on the form at the location you want to draw the new button.
3. Hold down the mouse button and draw the outline of the button on the form.
4. When you release the mouse button, the new button appears at the location specified.
5. Right click on the new button → Properties. The Push Button Properties window appears.
6. Rename caption to “Invoke Set Play Method” and close this window.
7. Right click PBBOX control panel and select Properties. Pbox Control Properties appears.
8. Take note of the identifier: IDC_PBBXCTRL1, and close this window.
9. From main menu, select View → ClassWizard. MFC ClassWizard appears.
10. Select the Member Variables tab.
11. Select the Control ID: IDC_PBBXCTRL1 and click the Add Variable... button. The Add Member Variable window appears.
12. Add the member variable “m_MyPbboxCtrl1” as a control variable.
13. Click OK.
14. Double click new Invoke Set Play Method button. Add Member Function window appears.

15. Click OK. The button handler code appears.

```
void CSchmopBoxStuffDlg::OnButton1()
{
    // TODO: Add your control notification handler code here.
}
```

16. Type `m_pBbboxCtrl1` at the cursor. The ocx methods list appears.

```
void SCbboxStuffDlg::OnButton1()
{
    // TODO: Add your control notification handler code here.
    m_pBbboxCtrl1;
}
```

17. Scroll down and select SetPlayMode.

18. Type `();` after SetPlayMode. The new “Invoke Set Play Method” button is now fully operational and can be used to set the mode of operation for the PCXRC to PLAY.

Methods

The PBBOX ActiveX® control includes several custom methods. These can be used to control the actions and appearance of the PBBOX control as well as extract information from the PCXRC.

**Table 3-1  Pbbox Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>long GetCtlWidth()</td>
<td>Returns current control width in screen units</td>
</tr>
<tr>
<td>void SetCtlWidth(long nNewValue)</td>
<td>Sets control width in screen units</td>
</tr>
<tr>
<td>long GetCtlHeight()</td>
<td>Returns current control height in screen units</td>
</tr>
<tr>
<td>void SetCtlHeight(long nNewValue)</td>
<td>Sets control height in screen units</td>
</tr>
<tr>
<td>BSTR GetPlayCaption()</td>
<td>Returns current Play button caption as a String data type</td>
</tr>
<tr>
<td>void SetPlayCaption(LPCTSTR lpszNewValue)</td>
<td>Takes String data type as an argument and sets Play button caption</td>
</tr>
<tr>
<td>BSTR GetTeachCaption()</td>
<td>Returns current Teach button caption as a String data type</td>
</tr>
<tr>
<td>void SetTeachCaption(LPCTSTR lpszNewValue)</td>
<td>Takes String data type as an argument and sets Teach button caption</td>
</tr>
<tr>
<td>BSTR GetServoOnCaption()</td>
<td>Returns current Servo On button caption as a String data type</td>
</tr>
<tr>
<td>void SetServoOnCaption(LPCTSTR lpszNewValue)</td>
<td>Takes a String data type as an argument and sets the Servo On button caption</td>
</tr>
<tr>
<td>BSTR GetServoOffCaption()</td>
<td>Returns current Servo Off button caption as a String data type</td>
</tr>
<tr>
<td>void SetServoOffCaption(LPCTSTR lpszNewValue)</td>
<td>Takes a String data type as an argument and sets the Servo Off button caption</td>
</tr>
<tr>
<td>BSTR GetAlarmResetCaption()</td>
<td>Returns current AlarmReset button caption as a String datatype</td>
</tr>
<tr>
<td>void SetAlarmResetCaption(LPCTSTR lpszNewValue)</td>
<td>Takes a String data type as an argument and sets AlarmReset button caption</td>
</tr>
<tr>
<td>BSTR GetStepCaption()</td>
<td>Returns the current Step button caption as a String data type</td>
</tr>
</tbody>
</table>
### Table 3-1 Pbbox Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void SetStepCaption(LPCTSTR lpszNewValue)</code></td>
<td>Takes a String data type as an argument and sets the Step button caption</td>
</tr>
<tr>
<td><code>BSTR GetOneCycleCaption()</code></td>
<td>Returns the current 1 Cycle button caption as a String data type</td>
</tr>
<tr>
<td><code>void SetOneCycleCaption(LPCTSTR lpszNewValue)</code></td>
<td>Takes a String data type as an argument and sets the 1 Cycle button caption</td>
</tr>
<tr>
<td><code>BSTR GetAutoCaption()</code></td>
<td>Returns the current Auto button caption as a String data type</td>
</tr>
<tr>
<td><code>void SetAutoCaption(LPCTSTR lpszNewValue)</code></td>
<td>Takes a String data type as an argument and sets the Auto button caption</td>
</tr>
<tr>
<td><code>BSTR GetHoldCaption()</code></td>
<td>Returns the current Hold button caption as a String data type</td>
</tr>
<tr>
<td><code>void SetHoldCaption(LPCTSTR lpszNewValue)</code></td>
<td>Takes a String data type as an argument and sets the Hold button caption</td>
</tr>
<tr>
<td><code>BSTR GetStartCaption()</code></td>
<td>Returns the current Start button caption as a String data type</td>
</tr>
<tr>
<td><code>void SetStartCaption(LPCTSTR lpszNewValue)</code></td>
<td>Takes a String data type as an argument and sets the Start button caption</td>
</tr>
<tr>
<td><code>long SetPlayMode()</code></td>
<td>Sets the PCXRC to PLAY mode and returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td><code>long SetTeachMode()</code></td>
<td>Sets the PCXRC to TEACH mode and returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td><code>long SetServoOn()</code></td>
<td>Turns Servo Power On and returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td><code>long SetServoOff()</code></td>
<td>Turns Servo Power Off and returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td><code>long SetHoldOn()</code></td>
<td>Turns HOLD On and returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td><code>long SetHoldOff()</code></td>
<td>Turns HOLD Off and returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
</tbody>
</table>
### Table 3-1 Pbbox Methods

<table>
<thead>
<tr>
<th>Method Call</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>long SetStepMode()</td>
<td>Places PCXRC in STEP mode and returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td>long Set1CycleMode()</td>
<td>Places PCXRC in 1CYCLE mode and returns HRESULT from PCXRC API (Refer to API manual for information on error codes)</td>
</tr>
<tr>
<td>long SetAutoMode()</td>
<td>Places PCXRC in AUTO mode and returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td>long SetStartJobOn()</td>
<td>Starts current robot job and returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td>long SetResetOn()</td>
<td>Resets the current minor Alarm or Error and returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td>long GetPlayModeStatus(long FAR* pRetValue)</td>
<td>Returns 1 in pRetValue if in PLAY mode and 0 if not. Method returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td>long GetTeachModeStatus(long FAR* pRetValue)</td>
<td>Returns 1 in pRetValue if in TEACH mode and 0 if not. Method returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td>long GetHoldOnStatus(long FAR* pRetValue)</td>
<td>Returns 1 in pRetValue if in HOLD mode and 0 if not. Method returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td>long GetHoldOffStatus(long FAR* pRetValue)</td>
<td>Returns 1 in pRetValue if not in HOLD mode and 0 if in HOLD mode. Method returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td>long GetStepModeStatus(long FAR* pRetValue)</td>
<td>Returns 1 in pRetValue if in STEP mode and 0 if not. Method returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
<tr>
<td>long Get1CycleModeStatus(long FAR* pRetValue)</td>
<td>Returns 1 in pRetValue if in 1CYCLE mode and 0 if not. Method returns HRESULT from PCXRC API (refer to API manual for information on error codes)</td>
</tr>
</tbody>
</table>
PBBOX Properties

The following items can be manipulated through the properties dialogs of the particular development environment that is being used to create the Control Container:

Table 3-2  PBBOX Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CtlWidth</td>
<td>Sets the PBBOX width in screen units</td>
</tr>
<tr>
<td>CtlHeight</td>
<td>Sets the PBBOX height in screen units</td>
</tr>
<tr>
<td>PlayCaption</td>
<td>Sets the PLAY button caption</td>
</tr>
<tr>
<td>TeachCaption</td>
<td>Sets the TEACH button caption</td>
</tr>
<tr>
<td>OneCycleCaption</td>
<td>Sets the 1CYCLE button caption</td>
</tr>
<tr>
<td>AlarmResetCaption</td>
<td>Sets the ALARM RESET button caption</td>
</tr>
<tr>
<td>AutoCaption</td>
<td>Sets the AUTO button caption</td>
</tr>
<tr>
<td>HoldCaption</td>
<td>Sets the HOLD button caption</td>
</tr>
<tr>
<td>ServoOffCaption</td>
<td>Sets the OFF button caption</td>
</tr>
<tr>
<td>ServoOnCaption</td>
<td>Sets the ON button caption</td>
</tr>
<tr>
<td>StartCaption</td>
<td>Sets the START button caption</td>
</tr>
<tr>
<td>StepCaption</td>
<td>Sets the STEP button caption</td>
</tr>
</tbody>
</table>

Note: The PBBOX control does implement error handling of the codes returned from the API. It is up to the Control Container to determine how these codes should be handled.
## Index

**A**
- About this Manual 1-1
- Adding PBBOX to a Visual Basic Form 3-2
- Adding PBBOX to a Visual C++ dialog 3-4
- Alarm Reset 3-1
- ANSI/RIA 2-1

**C**
- Customer Service 1-1
- Cycle 3-1

**E**
- Example Container Using Visual Basic 6 3-2
- Example Container Using Visual C++ 6 3-4

**G**
- General Safeguarding Tips 2-2

**H**
- Hold 3-1

**I**
- Inserting PBBOX into dialog 3-5
- Installation Safety 2-2

**M**
- Maintenance Safety 2-4
- Mechanical Safety Devices 2-2
- Methods 3-8
- Mode 3-1

**O**
- Operation Safety 2-3

**P**
- PCXRC Control Panel 3-1
- Programming Safety 2-3

**R**
- Reference to Other Documentation 1-1
- Resizing by Dragging 3-3, 3-6
- Resizing the Control Panel 3-3, 3-5
- Resizing using Properties 3-3, 3-5
- Robotic Industries Association 2-1

**S**
- Servo Power 3-1
- Standard Conventions 2-1
- Start 3-1
- System Requirements 1-1