

# ActiveX Tool Kit

Manual Rev. 3.1







**By Galil Motion Control, Inc.**









*Galil Motion Control, Inc.  
3750 Atherton Road  
Rocklin, California 95765  
Phone: 916-626-0101  
Fax: 916-626-0102  
Internet Address: [galil@galilmc.com](mailto:galil@galilmc.com)  
URL: [www.galilmc.com](http://www.galilmc.com)*





*Rev 02/2004*



# Contents

<b>CONTENTS .....</b>	<b>I</b>
<b>CHAPTER 1 INTRODUCTION .....</b>	<b>1</b>
<i>Installing the ActiveX Tool Kit.....</i>	<i>1</i>
<i>Adding the ActiveX controls to your Visual Basic Project .....</i>	<i>2</i>
<i>Linking the ActiveX controls to the Galil Controller .....</i>	<i>2</i>
<i>Using the ActiveX Help Files .....</i>	<i>4</i>
<i>Receiving Unsolicited Responses from the Controller.....</i>	<i>4</i>
<i>Adding the DMCCOM40.BAS Module to Your Project (Optional).....</i>	<i>5</i>
 <b>CHAPTER 2 – DMCSHELL CONTROL.....</b>	<b>6</b>
<i>Galil Properties, Methods, and Events.....</i>	<i>6</i>
<i>DMCShell Property Descriptions.....</i>	<i>7</i>
<i>DMCShell Event Descriptions .....</i>	<i>18</i>
<i>DMCShell Method Descriptions .....</i>	<i>20</i>
<i>DMCShell Examples.....</i>	<i>24</i>
 <b>CHAPTER 3 – DMCPOLL CONTROL .....</b>	<b>30</b>
<i>Galil Properties and Events.....</i>	<i>30</i>
<i>DMCPoll Property Descriptions.....</i>	<i>30</i>
<i>DMCPoll Events .....</i>	<i>43</i>
<i>DMCPoll Examples.....</i>	<i>44</i>
 <b>CHAPTER 4 – DMCTERMINAL CONTROL .....</b>	<b>48</b>
<i>Galil Properties, Methods, and Events.....</i>	<i>48</i>
<i>DMCTerminal Property Descriptions.....</i>	<i>48</i>
<i>DMCTerminal Event Descriptions .....</i>	<i>54</i>
<i>DMCTerminal Method Descriptions .....</i>	<i>55</i>
<i>DMCTerminal Examples .....</i>	<i>56</i>
 <b>CHAPTER 5 - DMCTERMINAL2 CONTROL.....</b>	<b>58</b>
<i>Galil Properties, Methods, and Events.....</i>	<i>59</i>
<i>DMCTerminal2 Property Descriptions.....</i>	<i>59</i>
<i>DMCTerminal2 Method Descriptions .....</i>	<i>67</i>
<i>DMCTerminal2 Event Descriptions .....</i>	<i>70</i>
<i>DMCTerminal2 Example .....</i>	<i>72</i>
 <b>CHAPTER 6 – DMCSEND CONTROL.....</b>	<b>76</b>
<i>Galil Properties, Methods, and Events.....</i>	<i>76</i>
<i>DMCSend Property Descriptions .....</i>	<i>77</i>
<i>DMCSend Event Descriptions.....</i>	<i>88</i>
<i>DMCSend Method Descriptions.....</i>	<i>90</i>
<i>DMCSend Button Descriptions.....</i>	<i>94</i>
<i>DMCSend Examples .....</i>	<i>94</i>
 <b>CHAPTER 7 DMCScope CONTROL .....</b>	<b>98</b>

<i>Galil Properties and Events</i> .....	98
<i>DMCScope Property Descriptions</i> .....	99
<i>Galil Events</i> .....	115
<i>DMCScope Event Descriptions</i> .....	115
<i>DMCScope Examples</i> .....	116
 <b>CHAPTER 8 – DMCDIAGNOSTICS CONTROL</b> .....	<b>119</b>
<i>Galil Properties</i> .....	119
<i>Galil Events</i> .....	119
 <b>CHAPTER 9 – DMCDIAGSNONAXIS CONTROL</b> .....	<b>121</b>
<i>Galil Properties</i> .....	122
<i>Galil Events</i> .....	122
 <b>CHAPTER 10 – DMCSETUP CONTROL</b> .....	<b>123</b>
<i>Galil Properties</i> .....	124
<i>Galil Events</i> .....	124
 <b>CHAPTER 11 – DMCARRAY CONTROL</b> .....	<b>125</b>
<i>Galil Properties</i> .....	125
<i>DMCArray Property Descriptions</i> .....	125
<i>Galil Methods</i> .....	139
<i>DMCArray Method Description</i> .....	139
<i>Galil Events</i> .....	140
<i>DMCArray Event Descriptions</i> .....	140
<i>DMCArray Examples</i> .....	142
 <b>CHAPTER 12 – DMCPPLAYBACK CONTROL</b> .....	<b>145</b>
<i>Galil Properties</i> .....	145
<i>DMCPlayback Property Descriptions</i> .....	145
<i>Galil Events</i> .....	154
<i>DMCPlayback Event Descriptions</i> .....	154
<i>DMCPlayback Examples</i> .....	155
 <b>CHAPTER 13 – DMCCOMPRESS CONTROL</b> .....	<b>159</b>
<i>Galil Properties</i> .....	159
<i>DMCCompress Property Descriptions</i> .....	159
<i>Galil Events</i> .....	162
<i>DMCCompress Event Descriptions</i> .....	162
 <b>CHAPTER 14 – DMCMOVE CONTROL</b> .....	<b>165</b>
<i>Galil Properties</i> .....	165
<i>DMCMove Property Descriptions</i> .....	166
<i>Galil Events</i> .....	175
<i>DMCMove Event Descriptions</i> .....	175
<i>Galil Methods</i> .....	176
<i>DMCMove Method Descriptions</i> .....	176
 <b>CHAPTER 15 – DMCREGISTER CONTROL</b> .....	<b>179</b>
<i>Galil Properties and Methods</i> .....	179
<i>DMCRegister Property Descriptions</i> .....	179

<i>DMCRegister Method Descriptions</i> .....	189
<i>DMCRegister Examples</i> .....	192
 <b>CHAPTER 16 – DMCI/O CONTROL</b> .....	<b>193</b>
<i>Galil Properties</i> .....	193
<i>DMCIO Property Descriptions</i> .....	194
<i>Galil Events</i> .....	204
<i>DMCIO Event Descriptions</i> .....	204
 <b>CHAPTER 17 – DMCVECTOR CONTROL</b> .....	<b>207</b>
<i>Galil Properties</i> .....	207
<i>DMCVector Property Descriptions</i> .....	207
<i>Galil Events</i> .....	217
<i>DMCVector Event Descriptions</i> .....	217
<i>Galil Methods</i> .....	218
<i>DMCVector Method Descriptions</i> .....	218
<i>DMCVector Examples</i> .....	219
 <b>CHAPTER 18 – DMCANALOG CONTROL</b> .....	<b>221</b>
<i>Galil Properties</i> .....	221
<i>DMCAnalog Property Descriptions</i> .....	221
<i>Galil Events</i> .....	231
<i>DMCAnalog Event Descriptions</i> .....	231
<i>DMCAnalog Examples</i> .....	233
 <b>CHAPTER 19 – DMCDIAGNOSTICTESTS CONTROL</b> .....	<b>234</b>
<i>Galil Properties</i> .....	234
<i>DMCDiagnosticTests Property Descriptions</i> .....	234
<i>Galil Events</i> .....	239
<i>DMCDiagnosticTests Event Descriptions</i> .....	239
<i>Galil Methods</i> .....	241
<i>DMCPrintReport Method Descriptions</i> .....	241
<i>Test Parameters</i> .....	241
<b>APPENDIX 1 - PROGRAM EXAMPLES</b> .....	<b>244</b>
<i>Experiment 1- Display Motor Position</i> .....	244
<i>Experiment 2- Interface to Start/Stop Motor</i> .....	245
<i>Experiment 3- Using Sliders</i> .....	246
<i>Experiment 4- User Inputs Gear Ratio</i> .....	247
<i>Experiment 5- Polling Windows</i> .....	249
<i>Experiment 6- Terminal and Scope Objects</i> .....	250
<i>Experiment 7- Move Tool</i> .....	251
<i>Experiment 8- Teach and Playback</i> .....	253
<b>APPENDIX 2 – DATA RECORD, ERROR CODE, REGISTRY ENTRY, AND OTHER GLOBAL CONSTANTS</b> .....	<b>256</b>
<i>Data Record Constants</i> .....	256
<i>Error Code Constants</i> .....	258
<i>Registry Entry Constants</i> .....	258
<i>Global Constants (and Public Constants)</i> .....	259
<b>APPENDIX 3 - DISTRIBUTING YOUR APPLICATION</b> .....	<b>259</b>
<b>APPENDIX 4 - TROUBLESHOOTING</b> .....	<b>261</b>

<i>General Tips and Tricks</i> .....	261
<b>APPENDIX 5 - USING THE ACTIVEX TOOL KIT WITH LABVIEW</b> .....	<b>263</b>
<b>APPENDIX 6 - USING THE ACTIVEX TOOLKIT WITH VB .NET</b> .....	<b>267</b>







# Chapter 1 Introduction

The Galil ActiveX Tool Kit provides a great increase in productivity for programming with Visual Basic (5,6, and .NET), Delphi, Visual C++, LabVIEW, or any other ActiveX compatible development system in Microsoft Windows 32-bit environments. These Microsoft operating systems include Windows 98SE, Millennium Edition, 2000, NT4.0, and XP.

Communication with Galil controllers is simplified so that only a few lines of programming statements are necessary to perform real work. In addition, the Galil ActiveX Tool Kit includes a set of pre-built objects such as a polling window and a terminal, which can be incorporated into an application by simply dragging the appropriate tool icon onto a Visual Basic form. For this manual, Visual Basic 6.0 is used as the foundation for the syntax, arguments, and examples. Some examples and tips are also given for VB.NET and LabVIEW.

## Installing the ActiveX Tool Kit

It is recommended that the ActiveX Tool Kit is the last Galil software package installed. It is also recommended that a programming environment is installed prior to installing the ActiveX Tool Kit. This will insure that the ActiveX control licenses will get installed in the correct directory, and that the controls will be properly registered with Windows.

Using the Galil Software products CD-Rom, run the file DMCOCX32.EXE in the DMCOCX directory. Password protected software requires a password to begin the installation. The password should be included on the cover of the CD-Rom case. When the installation dialog prompts for the password, enter the password as one word, lowercase. When the installation is complete and the computer is re-booted, the ActiveX controls will be available for installation into the Visual Basic toolbox.

The ActiveX Tool Kit installation file can also be downloaded from the Galil website located at [www.galilmc.com](http://www.galilmc.com).

### **Note: for Windows NT 4.0 / 2000/ XP users**

Since Windows NT 4.0 / 2000/ XP allows different users to log onto the same machine certain considerations must be made if the user logs in as someone other than the 'Administrator'. When the ActiveX Tool Kit is installed it registers its files in the current users configuration. If another user logs in they cannot place the tools in their project. Each user must register the files for their own user configuration by reinstalling the ActiveX Tool Kit after they log in.

### **Registering ActiveX Controls**

Normally, it is not necessary to register the ActiveX controls; the installation procedure previously described automatically takes care of registration. However, the controls can fail to be recognized by Visual Basic under certain conditions. A common source of this problem is when another piece of Galil software such as WSDK is installed on top of the ActiveX Tool Kit. If you receive a message from Visual Basic such as : "License information for this component not found..." when trying to execute a program, you need to re-register the controls. To re-register the controls, go to the Windows Start menu and select Programs->Galil->Register Galil OCX's. You will see a menu that allows you to select the controls to be registered. After selecting the necessary controls, click the Register button to register them.

## Adding the ActiveX controls to your Visual Basic Project

### Using Visual Basic 5/6

Once the tool kit is installed go to the 'Components' menu selection under the 'Project' main menu and select the tools you need from the list. Figure-1 below shows the components dialog box as it appears in VB6. Select the desired controls and click "OK". The controls will then appear in the toolbox on the left side of the VB design screen.

Note: Only one of the terminal controls may be selected (either DMCTerminal or DMCTerminal 2). See Chapter 4 and Chapter 5 for details for each terminal.

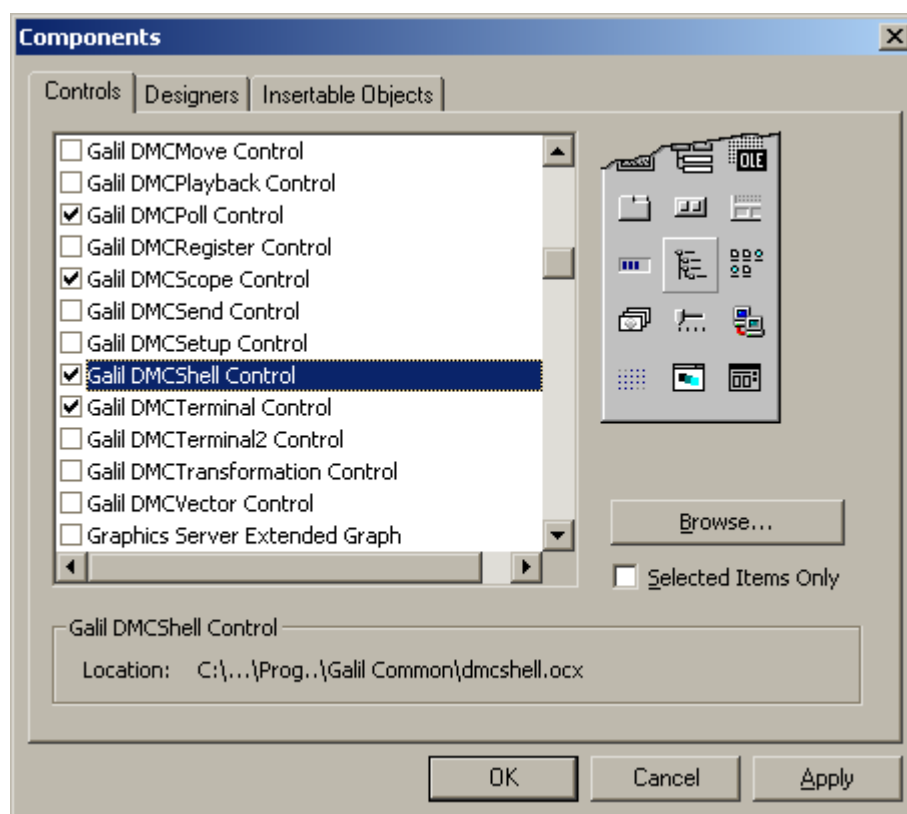


Figure 1 – Components dialog box in VB6  
(Note: Shown with only one DMCTerminal control selected)

## Linking the ActiveX controls to the Galil Controller

Each controller in the system must be added to the Galil registry before they can be accessed by 'registry aware' ActiveX controls such as DMCTerminal and DMCTerminal 2. Once the controllers in the system are registered with the Galil registry, then the Active-X controls only need to address them by their controller

number in the registry. This eliminates any need to keep track of the different communication formats. Use the DMCTERM.EXE program included with the tool kit to add a controller to the registry. Or follow the installation procedure outlined in the specific user manual for your controller. Figure-2 displays two different controllers as Controller1 and Controller2 in the Galil registry.

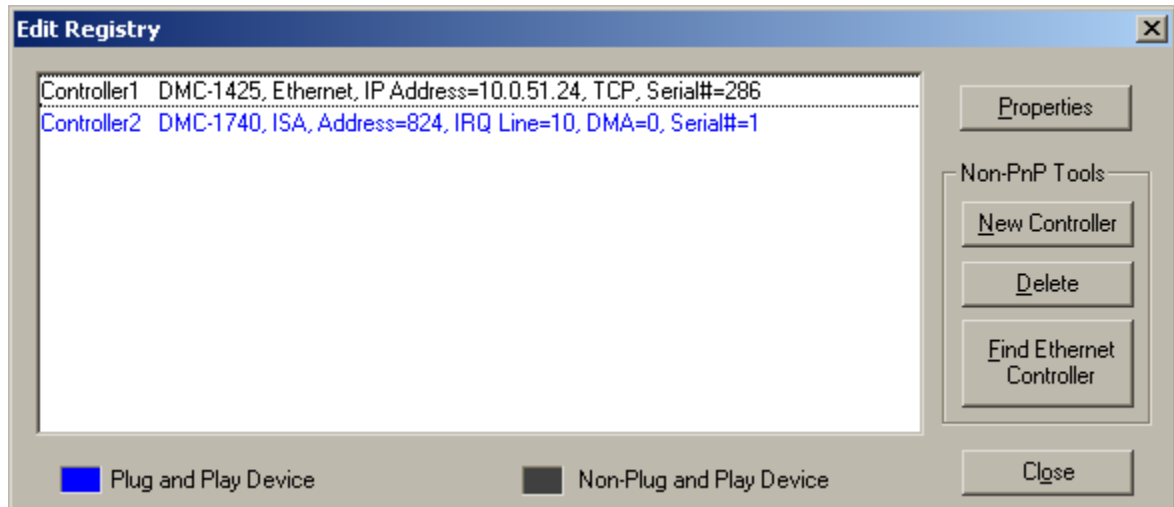


Figure 2 – Galil Registry

#### **Adding a controller to the registry (Ethernet or serial controllers only)**

Running the Galil program "DMCTerm", select the "Register Controller..." option under the "File" menu. In the form that appears click "New Controller", then select the controller type and specific information as required by the controller communications settings (i.e. IP address, Baud rate, etc.). Figure-3 shows The "Select Model..." dialog box.

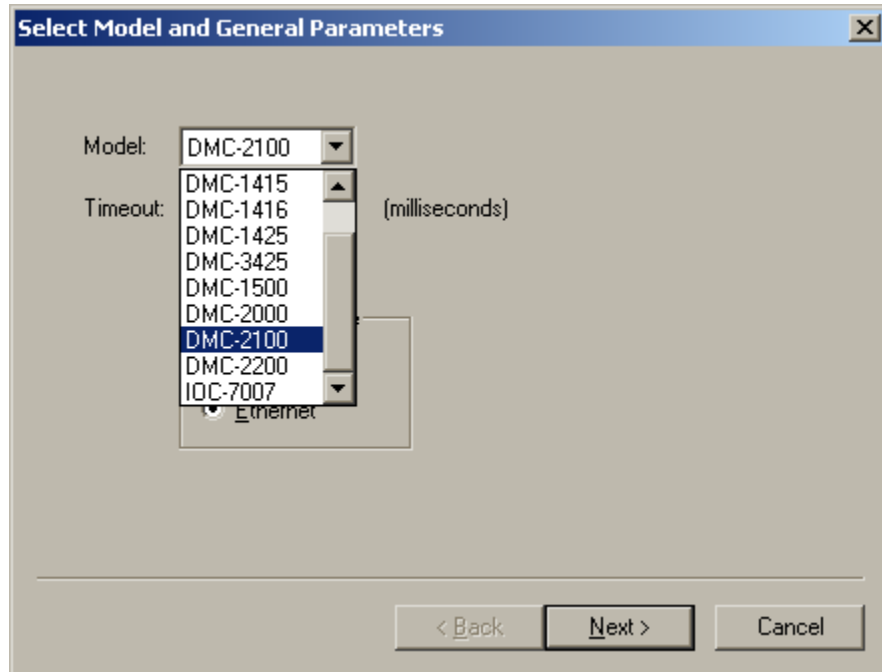


Figure 3 – Select Model and General Parameters

Another method to register controllers is with the DMCRegister OCX. This tool allows modification of the Galil registry from within a Visual Basic program. Providing access to the registry is a simple way to let users change the address of controllers or create a registry on a computer, but is only available at run time. See the complete detailed description of the DMCRegister control in the main section of this manual.

## Using the ActiveX Help Files

Each ActiveX control has an associated help file. The help files are located in the \DMCOCX\OCX directory and have the extension .HLP. The files may be viewed by double-clicking on the file from a file manager such as Windows Explorer. The help file also can be opened from within Visual Basic by selecting a control located on a form and pressing the F1 key.

The help files are an excellent resource, providing the most up-to-date descriptions of all of the Properties, Events and Methods for each control. In addition, extensive sample code is provided.

## Receiving Unsolicited Responses from the Controller

Unsolicited responses are message outputs from the Galil motion controller, which results from executing a Galil DMC application program from within the controller. These messages are mainly generated by using MG statements in the program, but can also come from using interrogation commands such as TP and TE within the program.

The DMCShell control receives unsolicited messages through the DMCResponse event procedure. The event procedure executes when unsolicited message are posted to the DMCShell control, however, the posting method and timing depends on the controller communications method being used (ISA/PCI Bus, Ethernet, serial, or USB).

For serial and USB communications, the DMCSHELL control must poll for the unsolicited messages. The DMCPollController property must be set to **True** after the connection is established with DMCCONNECT. The rate at which the DMCSHELL polls for the unsolicited messages is set by the DMCPollInterval property.

For ISA and PCI bus controllers using version 7 drivers and greater and that have the “Communications Interrupt” method enabled, the DMCPollController property is not required. Unsolicited messages will automatically generate an “interrupt” on the bus and immediately post the message to the DMCSHELL. However, if older drivers are used or if “Stall” or “Delay” methods are enabled, then the DMCSHELL must use the DMCPollController property. The rate at which the DMCSHELL polls for the unsolicited messages is set by the DMCPollInterval property.

For Ethernet communications, there are two possible scenarios depending on how the controller is configured to receive unsolicited messages within the Galil registry. The first is the default case where the controller receives unsolicited message from the same handle as the main communications (only one handle open to the controller upon connection). In this case the DMCPollController property must be used. The rate at which the DMCSHELL polls for the unsolicited messages is set by the DMCPollInterval property. The second case is when the controller is configured to open a second handle to receive unsolicited messages (two handles are open to the controller upon connection). In this case, the DMCPollController is not required and any unsolicited messages are immediately posted to the DMCSHELL.

To configure the Ethernet controller for the desired unsolicited messages method, highlight the controller in the Galil registry and select the “Properties” button. When the dialog appears, select the “Ethernet Parameters” tab which then displays the “Unsolicited Messages” options.

## Adding the DMCCOM40.BAS Module to Your Project (Optional)

The DMCCOM40.BAS module can be used in a VB application to allow the program to utilize the Galil API function calls as described in the DMCWin32 Galil API toolkit manual. The DMCCOM40.BAS module is available with the **DMCWIN32** programmer’s utility (available for free download from <http://www.galilmc.com/support/download.html>).

This module also contains public declarations for all the constants used for the DataRecord access feature associated with the DMCSHELL.ocx. This module can be added to your project to allow the DMCSHELL methods GetDataRecordById or GetDataRecordItemById2 to access the data record using data ID identifiers. See Appendix 2 for details of the data record ID identifiers included in the DMCCOM40.BAS file.

To add this file, select ‘Add Module’ from the ‘Project’ menu in VB5/6. Select the DMCCOM40.BAS file from the default directory (C:\Program Files\Galil\DMCWIN\VB) which is present after DMCWin32 has been installed.



# Chapter 2 – DMCShell Control

The base communications shell object provides all the necessary functions to send commands and receive responses from the controller. It also provides functions to download, upload, and send applications programs, reset, master reset, and clear the controller, respond to interrupts, and poll the controller for background messages. The base communications shell object must be added to all projects. One shell is required for each controller in a multi-controller application.

NOTE: If a project has multiple forms use just one shell object for the entire project. As long as the form containing the shell is in memory ( has not been unloaded ) the shell will be active. To access a shell located on form1 from form2, first reference the form, then the shell. See the examples at the end of this chapter or in the appendix, for further details on using the shell object.

## Galil Properties, Methods, and Events

### Properties

DMCClear  
DMCCCommand  
DMCConnect  
DMCController  
DMCDataRecordRevision  
DMCDelay  
DMCDiagnostics  
DMCFileBuffer  
DMCFileOperation  
DMCMasterReset  
DMCMotionCompleteAxes  
DMCPollController  
DMCPollInterval  
DMCProgramLabel  
DMCReset  
DMCResponse  
DMCTimeout  
DMCVersion

### Methods

ConvertCommandAsciiToBinary  
ConvertCommandBinaryToAscii  
ConvertFileAsciiToBinary  
ConvertFileBinaryToAscii  
GetDataRecordItem  
GetDataRecordItemById  
RefreshDataRecord

### Events

DMCError  
DMCInterrogate  
DMCInterrupt  
DMCMotionComplete  
DMCResponse

## DMCShell Property Descriptions

### DMCBinaryCommand

#### Description

Used to send Galil language commands in binary format to the Galil motion controller.

#### Usage

[form.][control.]**DMCBinaryCommand**[= command]

#### Setting

The setting for the DMCBinaryCommand property may be any valid Galil language command which has been converted from ASCII to binary. Galil language commands may be converted from ASCII to binary using the ConvertCommandAsciiToBinary method.

#### Remarks

Once the DMCBinaryCommand property has been set, the Galil language command is sent to the Galil motion controller. If the Galil language command has a response, the DMCResponse property will be set by the DMCShell control. If there is an error, the application program will be notified through the DMCError event. A question mark ("?",) in the DMCResponse property also denotes an unrecognized command or a command error. Once the command has completed, the DMCBinaryCommand property is set to a NULL string ("") by the DMCShell control.

For a list of all the Galil language commands and their syntax, see the Command Reference Guide for the Galil motion controller installed.

Although the DMCBinaryCommand property is of data type variant, you must coerce the value to be of data type string (it must not be a fixed-length string). For C++ programmers, the variant type must be set to VT\_BSTR and the variant value must be a BSTR. A BSTR is equivalent to a Visual Basic string. The reason for this is that a binary command can contain embedded null characters which are permitted in the Visual Basic string or BSTR data type.

#### Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

#### Data Type

Variant

---

### DMCClear

#### Description

Used to clear the Galil motion controller input and output buffers.

#### Usage

[form.][control.]**DMCClear**[= {True | False}]

#### Setting

The DMCClear property settings are:

Setting Description	
True	Clear the Galil motion controller.
False	Does nothing. This is the Default setting.

## Remarks

Once the Galil motion controller input and output buffers have been cleared, the DMCClear property is reset to False by the DMCSHELL control.

## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

## Data Type

Integer (Boolean)

## DMCCOMMAND

## Description

Used to send Galil language commands to the Galil motion controller.

## Usage

[form.][control.]DMCCOMMAND[= command]

## Setting

The setting for the DMCCOMMAND property may be any valid Galil language command. Only one Galil language command may be used per instance. That is, the DMCCOMMAND property will ignore any Galil language commands after the first semi-colon (;). Note that a Galil language command is a string.

## Remarks

Once the DMCCOMMAND property has been set, the Galil language command is sent to the Galil motion controller. If the Galil language command has a response, the DMCRSPONSE property will be set by the DMCSHELL control. If there is an error, the application program will be notified through the DMCEVENT event. A question mark ("?",) in the DMCRSPONSE property also denotes an unrecognized command or a command error. Once the command has completed, the DMCCOMMAND property is set to a NULL string ("") by the DMCSHELL control.

For a list of all the Galil language commands and their syntax, see the Command Reference Guide for the Galil motion controller installed.

## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

## Data Type

String

---

## DMCCONNECT

## Description

Used to open or close a communication session with the Galil motion controller.

## Usage

[form.][control.]DMCCONNECT[= {True | False}]

## Setting

The DMCCONNECT property settings are:

### Setting Description

True	Open a connection to the Galil motion controller.
False	Close a connection to the Galil motion controller.



## Remarks

The DMCCConnect property uses the property DMCCController when opening a communication session with the Galil motion controller. This property must be set correctly in order for a connection to be established. If a connection can not be established, the DMCCConnect property is set to False by the DMCSHELL control.

## Note

This property may be used only at run time.

## Data Type

Integer (Boolean)

## DMCCController

## Description

The number corresponding to the Galil motion controller registered in the Windows registry database.

## Usage

[form.][control.]DMCCController[= controller]

## Setting

The setting for the DMCCController property must be an integer between 1 and 16. The default setting is 1.

## Remarks

The Galil controller must first be registered using the register tool under Visual Basic, the DMCREG.EXE program included with the Visual Basic Tool Kit, or the terminal program ( DMCTermxx.exe ).

## Note

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

## Data Type

Long

---

## DMCDataRecordRevision

## Description

The data record access revision number. If the controller does not support the data record access feature, the value will be zero.

## Usage

revision = [form.][control.]DMCDataRecordRevision

## Setting

None. The DMCDataRecordRevision property is read-only.

## Remarks

The DMCDataRecordRevision property is important if you are using the GetDataRecordItem method. This method uses predefined offsets into the data record in order to retrieve specific items from the data record. These offsets can change with the revision of the data record.

## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established. This property is read-only.

## Data Type

Integer

---

## **DMCDelay**

### **Description**

The amount of delay to add between sending a command to the Galil motion controller and attempting to receive a response back.

### **Usage**

[*form.*][*control.*]DMCDelay[= *delay*]

### **Setting**

The DMCDelay property setting must be a long integer within the range 0 through 2,000,000. The default setting is 5.

### **Remarks**

The DMCDelay property is a short pause (measured in milliseconds) between a write to and read from the Galil motion controller. A delay may be required for host PCs with very fast CPUs. You should add a delay if all data from commands is not received, or if time-out errors (-1) occur even though the DMCTimeout property is relatively large (1000 or more milliseconds). This most affects serial communications controllers. When adjusting the DMCDelay property, start with a small number such as 5 or 10. In most cases, it is safe to set the DMCDelay property to 0.

### **Note**

This property may be used at any time.

### **Data Type**

Long

---

## **DMCDiagnostics**

### **Description**

Used to collect diagnostic information on the communications traffic between the Galil motion controller and the application program. It is also useful for determining the cause of basic communications problems.

### **Usage**

[*form.*][*control.*]DMCDiagnostics[= {True | False}]

### **Setting**

The DMCDiagnostics property settings are:

<b>Setting Description</b>	
True	Open the diagnostics file and start collecting diagnostics.
False	Stop collecting diagnostics and close the diagnostics file.

### **Remarks**

Diagnostic messages are saved in a file named DIAGS.TXT in the current directory. The file is created every time the DMCDiagnostics property is set to True, erasing the previous contents of the file if any. To close the file, set the DMCDiagnostics property back to False.

### **Note**

This property may be used at any time.

### **Data Type**

Integer (Boolean)

---

## **DMCFileBuffer**

### **Description**

Used to buffer the Galil language application program for the file operations download from buffer and upload to buffer. These file operations are activated by the DMCFileOperation property.

### **Usage**

[form.][control.]DMCFileBuffer[= application program]

### **Setting**

The setting for the DMCFileBuffer property may be any valid Galil language application program. Program lines are separated by carriage return (CR) and line feed (LF) characters, decimal 13 and 10 respectively.

### **Remarks**

The DMCFileBuffer property must be set before the download from buffer file operation can commence.

### **Note**

This property may be used at any time.

### **Data Type**

String

---

## **DMCFileName**

### **Description**

Used to set the file name for a file operation. The available file operations upload, download, and send are activated by the DMCFileOperation property.

### **Usage**

[form.][control.]DMCFileName[= file name]

### **Setting**

The setting for the DMCFileName property may be any valid DOS file name, including the path. Note that a file name is a string.

### **Remarks**

The DMCFileName property must be set before any file operation can commence.

For the download and send file operations, the file must already exist. For the upload file operation, if the file exists, it will be overwritten, else it will be created.

Once a file operation has completed, the DMCFileName property is set to a NULL string ("" ) by the DMCSHELL control.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

String

---

## **DMCFileOperation Property**

### **Description**

Used to perform the available Galil motion controller file operations: download file, upload file, send file, download from buffer, upload to buffer, send binary file.

### **Usage**

[form.][[control.]]**DMCFileOperation**[= file operation]

### **Setting**

The DMCFileOperation property settings are:

<b>FileOperationNone</b>	Does nothing. This is the default setting.
<b>FileOperationDownload</b>	Download a DMC file to the Galil motion controller.
<b>FileOperationDownloadFromBuffer</b>	Download an application program from a buffer (DMCFileBuffer property).
<b>FileOperationUpload</b>	Upload a file (app program) from the Galil motion controller.
<b>FileOperationUploadToBuffer</b>	Upload an application program to a buffer (DMCFileBuffer property).
<b>FileOperationSend</b>	Send a file consisting of Galil language commands in ASCII format to the Galil motion controller.
<b>FileOperationSendBinary</b>	Send a file consisting of Galil language commands in binary format to the Galil motion controller.

### **Remarks**

File download transmits the contents of a file to the Galil motion controller's program memory. The file must consist of valid Galil language commands.

File upload transmits the contents of the Galil motion controller's program memory to a file.

The send file and send binary file operations transmit each line in a file to the Galil motion controller as a separate Galil language command; each Galil language command is interpreted immediately. After the send file or send binary file operation, the contents of the file are not retained in the Galil motion controller's program memory. The file must consist of valid Galil language commands.

For file operations 1,2,3, and 6 the DMCFileName property must be set before the file operation can commence.

For the download file, send file, and send binary file operations, the file must already exist. For the upload file operation, if the file exists, it will be overwritten, else it will be created.

When downloading or uploading a DMC application program using a buffer (the DMCFileBuffer property), each program line is separated by a CR/LF pair. In Visual Basic terms, this is Chr(13) + Chr(10) added to each line.

Once a file operation has completed, the DMCFileOperation property is set back to 0 by the DMCSHELL control.

**Note**

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

**Data Type**

Long

---

**DMCInterrogate****Description**

Used to send Galil language commands at regular intervals to the Galil motion controller. Normally, the Galil language commands chosen are ones which return values such as position or error. In this way, the Galil motion controller can be interrogated for many attributes including the state of any variables being used.

**Usage**

*[form.][control.]DMCInterrogate(index) [= command]*

**Setting**

The setting for the DMCInterrogate property may be any valid Galil language command. Note that a Galil language command is a string. The DMCInterrogate property is actually an property array with 10 elements. Therefore, up to 10 interrogation commands may be set. To cancel an interrogation command, set the appropriate DMCInterrogate property (determined by the index) to a NULL string ("");

**Remarks**

Once an element of the DMCInterrogate property has been set, the Galil language command is sent to the Galil motion controller at regular intervals as determined by the DMCPollInterval property. If the Galil language command has a response, the application program will be notified through the DMCInterrogate event. The first argument of the DMCInterrogate event (which is Index) determines which interrogation command the response is for.

If there is an error, the application program will be notified through the DMSError event. A question mark ("?",) in the Response argument of the DMCInterrogate event also denotes an unrecognized command or a command error.

For a list of all the Galil language commands and their syntax, see the Command Reference Guide for the installed Galil motion controller.

**Note**

Interrogation will occur during normal operation of the controller. It will not work when the controller is waiting after a trippoint command, like After Motion ( AM ) has been sent to the card or if the controller is waiting for an input in response to the IN command.

Do not use the interrogate function if the variable needs to be displayed. Use the polling window tool included in this tool kit instead.

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

**Data Type**

Array of Strings

---

**DMCMasterReset****Description**

Used to reset the Galil motion controller to its factory default settings and erases the EEPROM.

## Usage

[form.][control.]DMCMasterReset[= {True | False}]

## Setting

The DMCMasterReset property settings are:

Setting	Description
True	Master reset the Galil motion controller.
False	Does nothing. This is the Default setting.

## Remarks

Once the Galil motion controller has been reset, the DMCMasterReset property is reset to False by the DMCSHLL control. Setting the DMCMasterReset property to True has the same effect as the Galil language command "<control>R<control>S".

## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

## Data Type

Integer (Boolean)

---

## DMCMotionCompleteAxes

### Description

Used to set a wait for a motion complete condition. When motion is complete on the specified axes, the DMCMotionComplete event will fire.

### Usage

[form.][control.]DMCMotionCompleteAxes[= axis list]

### Setting

The setting for the DMCMotionCompleteAxes property must be a string consisting of a list of axes in upper case. For example, the string "X" will mean that the DMCMotionComplete event will fire when motion is complete on the X axis. The string "ABCDEF" will mean that the DMCMotionComplete event will fire when motion is complete on the A, B, C, D, E, and F axes. The axes S and T can be used to wait for motion complete for coordinated motion.

### Remarks

The DMCMotionCompleteAxes property is set to an empty string once the DMCMotionComplete event fires. You must set the DMCMotionCompleteAxes property each time you want the DMCMotionComplete event to fire upon a motion complete condition.

*Note: This property may not be referenced at runtime until a connection to the Galil motion controller has been established.*

### Data Type

String

---

## DMCPollController

### Description

Used to poll the Galil motion controller for unsolicited responses.

### Usage

[form.][control.]DMCPollController[= {True | False}]

## Setting

The DMCPollController property settings are:

### Setting Description

True	Begins polling the Galil motion controller.
False	Ends polling the Galil motion controller.

## Remarks

If the DMCPollController property is set to True, the DMCSHELL control will poll the Galil motion controller for unsolicited responses every *n* milliseconds where *n* is equal to the setting for the DMCPollInterval property.

If there is an unsolicited response from the Galil motion controller, the application program will be notified through the DMCRResponse event.

Unsolicited responses are responses (output) from the Galil motion controller which results from commands executing within a Galil DMC application program. Commands that cause unsolicited responses include MG, IN, TP or any command in a DMC application program that causes data to be sent out from the controller. In most cases the MG could be used to notify Visual Basic of a change of condition or that the program reached a certain point

### Notes:

*This property may not be referenced at runtime until **after** a connection to the Galil motion controller has been established with DMCCConnect.*

*Also, DMCPollController is not necessary to obtain unsolicited messages when using certain communication methods. These include the "Communications Interrupt" method associated with PCI and ISA controllers and the "second handle" method used with Ethernet controllers.*

## Data Type

Integer (Boolean)

---

## DMCPollInterval

### Description

The number of milliseconds to wait between checking the Galil motion controller for unsolicited responses.

### Usage

*[form.][control.]DMCPollInterval[= interval]*

### Setting

The DMCPollInterval property setting must be an integer within the range 75 through 10,000. The default setting is 500.

### Remarks

If the DMCPollController property is set to True, the DMCSHELL control will poll the Galil motion controller for unsolicited responses every *n* milliseconds where *n* is equal to the setting for the DMCPollInterval property.

Unsolicited responses are responses (output) from the Galil motion controller which result from executing a Galil language application program. Polling the Galil motion controller is not necessary to obtain responses during normal interactive communication. For this, there is the DMCRResponse property. However, if you intend to monitor output which results from executing a Galil language application program from within your application program, you must use polling.

If there is an unsolicited response from the Galil motion controller, the application program will be notified through the DMCRResponse event.

**Note**

This property may be used at any time. Setting the DMCPollInterval to a very small number (50 ms or less) will cause the system to skip interrogation commands.

**Data Type**

Long

---

**DMCProgramLabel****Description**

Used to set the program label for a download file operation.

**Usage**

`[form.][control.]DMCProgramLabel[= program label]`

**Setting**

The setting for the DMCProgramLabel property must be a valid Galil language program label. Note that a program label is a string.

**Remarks**

The DMCProgramLabel property is optional for file downloads to the Galil motion controller. However, if you wish to download a Galil language program to a specific label, the DMCProgramLabel property must be set before the download file operation.

**Note**

This property may be used at any time. This property is saved if set at design time.

**Data Type**

String

---

**DMCReset****Description**

Used to reset the state of the Galil motion controller to its power-on condition. The previously saved state of the controller, along with parameter values and saved sequences, are restored.

**Usage**

`[form.][control.]DMCReset[= {True | False}]`

**Setting**

The DMCReset property settings are:

**Setting Description**

True	Reset the Galil motion controller.
False	Does nothing. This is the Default setting.

**Remarks**

Once the Galil motion controller has been reset, the DMCReset property is reset to False by the DMCSHELL control. Setting the DMCReset property to True has the same effect as the Galil language command "RS".

**Note**

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

**Data Type**

Integer (Boolean)

---

**DMCResponse**



## Description

Used to access the response to a Galil language command (sent via the DMCCCommand property) from the Galil motion controller.

## Usage

*response*= [*form.*][*control.*]DMCResponse

## Setting

None. The DMCResponse property is read-only.

## Remarks

When the DMCCCommand property has been set, a Galil language command is sent to the Galil motion controller. If the Galil language command has a response, the DMCResponse property will be set by the DMCSHLL control. If there is an error, the application program will be notified through the DMCError event. A question mark ("?",) in the DMCResponse property also denotes an unrecognized command or a command error.

The DMCResponse property is overwritten each time a command is sent to the Galil motion controller.

## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

## Data Type

String

---

## DMCTimeout

## Description

The number of milliseconds to wait for a response to a command sent to the Galil motion controller.

## Usage

[*form.*][*control.*]DMCTimeout[= *time-out*]

## Setting

The DMCTimeout property setting must be a long integer within the range 1 through 8,6400,000. The default setting is 1,000.

## Remarks

The DMCTimeout property should be set high enough so that the Galil motion controller has an adequate amount of time to respond to a given command or file operation. However, it should not be so high that any communication problem will cause excessive waiting on the part of the software. If a time-out occurs, the event DMCError will be fired with the ErrorNumber set to -1.

## Note

This property may be used at any time.

Time-outs may occur during a file download or when a trippoint command has been sent to the controller.

## Data Type

Long

---

## **DMCVersion**

### **Description**

Used to identify the model and version of the installed Galil motion controller.

### **Usage**

*version*= [*form.*][*control.*]**DMCVersion**

### **Setting**

None. The DMCVersion property is read-only.

### **Remarks**

The DMCVersion property is set by the control after the DMCCConnect property has been set to True, that is, after a connection to the Galil motion controller has been established.

### **Note**

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

### **Data Type**

String

---

## **DMCShell Event Descriptions**

### **Galil Events**

DMCError  
DMCInterrogate  
DMCInterrupt  
DMCMotionComplete  
DMCResponse

## **DMCError**

### **Description**

Used by the DMCShell control to notify the application program of an error.

### **Syntax**

Sub *ctlname*\_**DMCError**(*ErrorNumber*As Long, *ErrorMessage*As String)

### **Remarks**

**The two most frequent errors are:**

<b>ErrorNumber</b>	<b>Description</b>
-1	A time-out occurred.
-2	Galil language command error.

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.

2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

For a list of all the Galil language commands and their syntax, see the Command Reference Guide for the installed Galil motion controller.

Time-outs can occur for two reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCSHELL control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.

## **DMCInterrogate**

### **Description**

Used by the DMCSHELL control to notify the application program of a response to an interrogation command, set by using the DMCInterrogation property.

### **Syntax**

Sub *ctlname*\_DMCInterrogate(*IndexAs* Integer, *ResponseAs* String)

### **Remarks**

The first argument, *Index*, identifies which interrogation command (that is, the Galil language command referred to by DMCInterrogation(*Index*)) the response is applicable to. This is important when more than one interrogation command is currently active.

The second argument, *Response*, is the same response which would result if the Galil language command had been entered interactively, with the following exception: the terminating carriage return/line feed and colon are stripped from the response

---

## **DMCInterrupt**

### **Description**

Used by the DMCSHELL control to notify the application program of a bus level interrupt. This event is only sent for bus controllers that have been properly configured to be used with interrupts.

### **Syntax**

Sub *ctlname*\_DMCInterrupt(*StatusByteAs* Integer)

### **Remarks**

The *StatusByte* indicates which interrupt occurred. For more information on Galil motion controller interrupts and their return status bytes, see the Technical Reference Guide for the installed Galil motion controller.

## Note

The controller must have the interrupt selector jumper placed on the card before an IRQ can be sent (ISA cards). Refer to the User manual for setting up IRQ.

---

### **DMCMotionComplete**

#### **Description**

Used by the DMCSHELL control to notify the application program of a motion complete condition from the Galil motion controller. Setting the DMCMotionCompleteAxes property sets up the motion complete condition.

#### **Syntax**

Sub ctlname\_**DMCMotionComplete**()

#### **Remarks**

The DMCMotionComplete event will only fire if the DMCMotionCompleteAxes property has been previously set.

---

### **DMCResponse**

#### **Description**

Used by the DMCSHELL control to notify the application program of an unsolicited response from the Galil motion controller. The rate at which the responses from the Galil motion controller are checked is set with the DMCPollController property.

#### **Syntax**

Sub ctlname\_**DMCResponse**(ResponseAs String)

#### **Remarks**

Response contains the unsolicited response from the Galil motion controller. Unsolicited responses can occur when the Galil motion controller is executing a Galil language program and the Galil language program uses Galil language commands that evoke responses.

---

## **DMCSHELL Method Descriptions**

### **Galil Methods**

ConvertCommandAsciiToBinary  
ConvertCommandBinaryToAscii  
ConvertFileAsciiToBinary  
ConvertFileBinaryToAscii  
GetDataRecordItem  
GetDataRecordItemById  
RefreshDataRecord

## **ConvertCommandAsciiToBinary**

### **Description**

Used to convert a Galil language (DMC) command from ASCII to binary.

### **Usage**

RC = [form.][control.]ConvertCommandAsciiToBinary(AsciiCommand As String, BinaryCommand As String)

RC is a variable of data type long. Zero indicates success. If RC is non-zero, the method has encountered an error. Error codes are documented in the file DMCCOM40.BAS.

### **Remarks**

The AsciiCommand and BinaryCommand arguments are both Visual Basic strings. For C++ programmers this means they are of data type BSTR.

### **Note**

This method may only be used at run time.

---

## **ConvertCommandToBinaryAscii**

### **Description**

Used to convert a Galil language (DMC) command from binary to ASCII.

### **Usage**

RC = [form.][control.]ConvertCommandToBinaryAscii(BinaryCommand As String, AsciiCommand As String)

RC is a variable of data type long. Zero indicates success. If RC is non-zero, the method has encountered an error. Error codes are documented in the file DMCCOM40.BAS.

### **Remarks**

The BinaryCommand and AsciiCommand arguments are both Visual Basic strings. For C++ programmers this means they are of data type BSTR.

### **Note**

This method may only be used at run time.

---

## **ConvertFileAsciiToBinary**

### **Description**

Used to convert a file consisting of Galil language (DMC) commands from ASCII format to binary format.

### **Usage**

RC = [form.][control.]ConvertFileAsciiToBinary(AsciiFileName As String, BinaryFileName As String)

RC is a variable of data type long. Zero indicates success. If RC is non-zero, the method has encountered an error. Error codes are documented in the file DMCCOM40.BAS.

### **Remarks**

The AsciiFileName and BinaryFileName arguments are both Visual Basic strings. For C++ programmers this means they are of data type BSTR.

## Note

This method may only be used at run time.

---

## **ConvertFileBinaryToAscii**

### **Description**

Used to convert a file consisting of Galil language (DMC) commands from binary format to ASCII format.

### **Usage**

RC = [form.][control.]ConvertFileBinaryToAscii(BinaryFileName As String, AsciiFileName As String)

RC is a variable of data type long. Zero indicates success. If RC is non-zero, the method has encountered an error. Error codes are documented in the file DMCCOM40.BAS.

### **Remarks**

The BinaryFileName and AsciiFileName arguments are both Visual Basic strings. For C++ programmers this means they are of data type BSTR.

## Note

This method may only be used at run time.

---

## **GetDataRecordItem**

### **Description**

Used to retrieve a single data item from the data record used for Data Record Access.

### **Usage**

RC = [form.][control.]GetDataRecordItem(GeneralDataOffset As Integer, AxisDataOffset As Integer, DataItem As Variant)

RC is a variable of data type long. Zero indicates success. If RC is non-zero, the method has encountered an error. Error codes are documented in the file DMCCOM40.BAS.

### **Remarks**

Data Record Access refers to a feature, available on some Galil motion controllers such as the DMC-1700, which provides a data record consisting of general and axis specific status information such as input and output states, motor position, position error, and torque. The data record is physically refreshed by the controller at a specified interval such as 2ms, 4ms, etc.. The Data Record Access feature uses a different channel of communication other than the primary FIFO, so that the data record is always available. That is, it is not affected by other communication taking place between the host PC and the controller. The Data Record Access feature for the controller must be enabled in the Windows registry.

You can retrieve a specific data item from the data record by using the GeneralDataOffset and AxisDataOffset arguments as offsets into the data record. The allowable values for GeneralDataOffset and AxisDataOffset are documented in the file DMCCOM40.BAS. The global constants prefixed with "DRGenOffsets" refer to the GeneralDataOffset value and the global constants prefixed with "DRAxisOffsets" refer to the AxisDataOffset value.

## Note

This method may only be used at run time.

---

## **GetDataRecordItemById**

## **GetDataRecordItemById2**

### **Description**

Used to retrieve a single data item from the data record used for Data Record Access. Accesses the data record through item Ids. This method comes in two versions: one returns the DataItem as Variant, the other as Long.

### **Usage**

RC = [form.][control.]GetDataRecordItemById(ItemId As Integer, AxisId As Integer, DataItem As Variant)

RC = [form.][control.]GetDataRecordItemById(ItemId As Integer, AxisId As Integer, DataItem As Long)

RC is a variable of data type long. Zero indicates success. If RC is non-zero, the method has encountered an error. Error codes are documented in the file DMCCOM40.BAS.

### **Remarks**

Data Record Access refers to a feature, available on some Galil motion controllers such as the DMC-1700, which provides a data record consisting of general and axis specific status information such as input and output states, motor position, position error, and torque. The data record is physically refreshed by the controller at a specified interval such as 2ms, 4ms, etc.. The Data Record Access feature uses a different channel of communication other than the primary FIFO, so that the data record is always available. That is, it is not affected by other communication taking place between the host PC and the controller. The Data Record Access feature for the controller must be enabled in the Windows registry.

You can retrieve a specific data item from the data record by using the ItemId and AxisId arguments to identify the data item. The ItemId argument is used for every data item to be retrieved from the data record. The AxisId argument is used when the item refers to axis specific data, otherwise zero can be used for the AxisId. The allowable values for ItemId and AxisId are documented in the file DMCCOM40.BAS. The global constants prefixed with "DRId" refer to the ItemId and AxisId values.

One advantage the method GetDataRecordItemById has over GetDataRecordItem. is that the item and axis Ids are not firmware release dependent. That is, these values will never change. GetDataRecordItem uses offsets and these can change from release to release. The disadvantage of the method GetDataRecordItemById is that it is marginally slower than GetDataRecordItem.

### **Note**

This method may only be used at run time.

---

## **RefreshDataRecord**

### **Description**

Used to refresh the data record used for Data Record Access.

### **Usage**

RC = [form.][control.]RefreshDataRecord

RC is a variable of data type long. Zero indicates success. If RC is non-zero, the method has encountered an error. Error codes are documented in the file DMCCOM40.BAS.

### **Remarks**

Data Record Access refers to a feature, available on some Galil motion controllers such as the DMC-1700, which provides a data record consisting of general and axis specific status information such as input and output states, motor position, position error, and torque. The data record is physically refreshed by the

controller at a specified interval such as 2ms, 4ms, etc.. The Data Record Access feature uses a different channel of communication other than the primary FIFO, so that the data record is always available. That is, it is not affected by other communication taking place between the host PC and the controller. The Data Record Access feature for the controller must be enabled in the Windows registry.

The data record remains constant, regardless of the specified refresh rate (the physical refresh rate that is), until the RefreshDataRecord method is invoked. This ensures that when you query data record items from the data record using the GetDataRecordItem method they will be consistent. The Galil "DR" command controls the physical refresh rate for the data record.

### Note

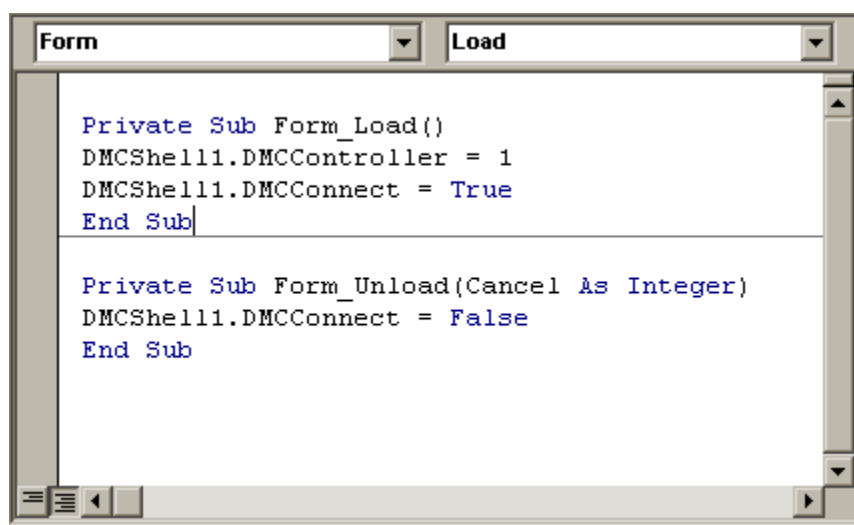
This method may only be used at run time.

---

## DMCShell Examples

### Example: Starting the Shell and Connecting to the Controller

After adding the shell object to Visual Basic place the object on any form where communication to the DMC controller is desired. Setting the 'DMCController' property selects the controller type from the DMC Registry as described above. If only one controller is defined in that registry the shell will select it by default. It is good practice to select the controller even if only one is registered. The only code needed to start the shell is shown below in a form load procedure:



*Note: **ALWAYS** disconnect the controller using the command DMCShell1.dmcconnect=False. Placing this command in the form unload procedure will ensure that the connection is turned off when the program terminates. This is especially vital when using a DMC-1500 controller. Serial port communication can be restored only by restarting Windows if the VB program ends without disconnecting. To ensure the connection is terminated place a button on the form with the disconnect command and the 'End' command on the next line. Name this button 'Exit' and use it to stop the program.*

### Example: Clearing and resetting the controller

The following example will clear the input and output buffers:

```
DMCShell1.DMCClear = True
```



The following example will reset the Galil motion controller:

```
DMCShell1.DMCReset = True
```

The following example will master reset the Galil motion controller:

```
DMCShell1.DCMasterReset = True
```

### **Example: Turning on Diagnostics**

The following example will open a diagnostics file named DIAGS.TXT:

```
DMCShell1.DMCDiagnostics = True
```

The following example will close the above diagnostics file:

```
DMCShell1.DMCDiagnostics = False
```

### **Example: Sending a command to the controller**

In this example, a command will be sent to a Galil motion controller and its response will be checked. The command is input by the user from a text control used as an input field and the response is output to the user to a text control used as an output field.

```
' Send the command to the Galil motion controller as input by
' the user
DMCShell1.DMCCommand = TextIn.text
' Place the response from the Galil motion controller in the
' output text control
TextOut.text = DMCShell1.DMCResponse
```

### **Example: Download, Upload, and Send Files**

**In this example, a file will be downloaded to the Galil motion controller.**

```
' Use the common file dialog to select a file name (Microsoft Common Dialog
Control 6.0)
CommonDialog1.DialogTitle = "Open DMC File"
CommonDialog1.DefaultExt = "*.DMC"
CommonDialog1.Filter = "DMC Files (*.DMC)|*.DMC"
CommonDialog1.FilterIndex = 1
CommonDialog1.ShowOpen
If CommonDialog1.CancelError = False Then
    If CommonDialog1.FileName <> "" Then
        ' Set the DMCProgramLabel property to #
        ' to download the file to the end of memory
        DMCShell1.DMCProgramLabel = "#"
        ' Set the DMCFileName property
        DMCShell1.DMCFileName = CommonDialog1.FileName
        ' Set the DMCFileOperation property
        DMCShell1.DMCFileOperation = FileOperationDownload
        ' Clear the DMCProgramLabel property
        DMCShell1.DMCProgramLabel = ""
    End If
End If
```

**In this example, a file will be uploaded from the Galil motion controller.**

```
' Use the common file dialog to select a file name
CommonDialog1.DialogTitle = "Save DMC File As"
CommonDialog1.DefaultExt = "*.DMC"
```

```

CommonDialog1.Filter = "DMC Files (*.DMC)|*.DMC"
CommonDialog1.FilterIndex = 1
CommonDialog1.ShowSave
If CommonDialog1.CancelError = False Then
    If CommonDialog1.FileName <> "" Then
        ' Set the DMCFileName property
        DMCSHELL11.DMCFileName = CommonDialog1.FileName
        ' Set the DMCFileOperation property
        DMCSHELL11.DMCFileOperation = FileOperationUpload
    End If
End If

```

**In this example, a file will be sent to the Galil motion controller.**

```

' Use the common file dialog to select a file name
CommonDialog1.DialogTitle = "Open DMC File"
CommonDialog1.DefaultExt = "*.DMC"
CommonDialog1.Filter = "DMC Files (*.DMC)|*.DMC"
CommonDialog1.FilterIndex = 1
CommonDialog1.ShowOpen
If CommonDialog1.CancelError = False Then
    If CommonDialog1.FileName <> "" Then
        ' Set the DMCFileName property
        DMCSHELL11.DMCFileName = CommonDialog1.FileName
        ' Set the DMCFileOperation property
        DMCSHELL11.DMCFileOperation = FileOperationSend
    End If
End If

```

**Example: Responding to the DMCError Event**

In this example, the application program reports on errors as sent by the Galil motion controller.

```

Sub DMCSHELL11_DMCError (ErrorNumber As Long, ErrorMessage As String)
    Dim RC As Integer
    RC = MsgBox(ErrorMessage, MB_ICONEXCLAMATION, "DMC Error")
End Sub

```

**Example: Interrogate Position and Error**

In this example, the application program will set interrogation commands for position and error of X and Y.

```

' Get the position of X
DMCSHELL11.DMCInterrogate(0) = "TPX;"
' Get the position of Y
DMCSHELL11.DMCInterrogate(1) = "TPY;"
' Get the error of X
DMCSHELL11.DMCInterrogate(2) = "TEX;"
' Get the error of Y
DMCSHELL11.DMCInterrogate(3) = "TEY;"

```

**Example: Responding to the DMCInterrogate Event**

In this example, four interrogation commands have been activated using the DMCInterrogate property. There are four text controls on the current form which correspond to the responses from the interrogation commands.

```

Sub DMCSHELL11_DMCInterrogate (Index As Integer, Response As String)
    If Index = 0 Then
        ' DMCSHELL11.DMCInterrogate(0) = "TPX;"
        PosX.Text = Response
    ElseIf Index = 1 Then

```

```

        ' DMCSHell11.DMCInterrogate(1) = "TPY;"
        PosY.Text = Response
    ElseIf Index = 2 Then
        ' DMCSHell11.DMCInterrogate(2) = "TEX;"
        ErrX.Text = Response
    ElseIf Index = 3 Then
        ' DMCSHell11.DMCInterrogate(3) = "TEY;"
        ErrY.Text = Response
    End If
End Sub

```

### Example: Responding to the Response Event

In this example, the DMCPollController property has been set to True, and the application program is placing the Galil motion controller unsolicited responses in a multi-line text control named ResponseOutput.

```

Sub DMCSHell11_DMCResponse (Response As String)
    ' Concatenate the response
    ResponseOutput.Text = ResponseOutput.Text + Response
    ' Set the cursor to the last line
    ResponseOutput.SelStart = Len(ResponseOutput.Text)
    ResponseOutput.SelLength = 0
End Sub

```

### Example: Responding to the Interrupt Event

In this example, the application program is responding to an interrupt as produced by the Galil motion controller.

```

Sub DMCSHell11_DMCInterrupt (StatusByte As Integer)
    Dim Msg As String
    Msg = "An Interrupt has occurred. Status = " + Format(StatusByte) + "."
    MsgBox Msg, MB_ICONEXCLAMATION, "DMC Interrupt"
End Sub

```

### Example: Sending a Binary Command

In this example, a binary command will be sent to a Galil motion controller and its response will be checked. The command is input by the user from a text control used as an input field and the response is output to the user to a text control used as an output field.

```

Dim RC As Long
Dim AsciiCommand As String
Dim BinaryCommand As String

' Assign the ASCII command as input by the user to a string variable
AsciiCommand = TextIn.text

' Convert the ASCII command to a binary command
RC = DMCSHell11.ConvertCommandAsciiToBinary(AsciiCommand, BinaryCommand)

' Send the binary command to the Galil motion controller
DMCSHell11.DMCBinaryCommand = BinaryCommand

' Place the response from the Galil motion controller in the output text control
TextOut.text = DMCSHell11.DMCResponse

```

### Example: Data Record Access

In this example, the Data Record Access feature will be used to retrieve information on controller status and axis position.

```

Dim RC As Long
Dim DataItem As Variant
Dim Status As Integer
Dim Position As Long
Dim PositionError As Long

' Refresh the logical data record
RC = DMCSHELL1.RefreshDataRecord

' Get the controller status
RC = DMCSHELL1.GetDataRecordItem(DRGenOffsetsGeneralStatus, DRAxisOffsetsNoAxis,
DataItem)
Status = DataItem

' Get the x axis position
RC = DMCSHELL1.GetDataRecordItem(DRGenOffsetsAxis1,
DRAxisOffsetsAxisMotorPosition, DataItem)
Position = DataItem

' Get the x axis error
RC = DMCSHELL1.GetDataRecordItem(DRGenOffsetsAxis1,
DRAxisOffsetsAxisPositionError, DataItem)
PositionError = DataItem

```

**THIS PAGE LEFT BLANK INTENTIONALLY**



## Chapter 3 – DMCPoll Control



The polling window object provides a mechanism for interrogating the controller at regular time intervals for information such as position and speed. The information can then be displayed in a form window with a variety of formats.

### Galil Properties and Events

#### Properties

DMCBevelInner  
DMCBevelOuter  
DMCBevelWidth  
DMCBorderWidth  
DMCCaptionPosition  
DMCCCommand  
DMCController  
DMCDataRecordItem  
DMCFormat  
DMCOffset  
DMCOutputPosition  
DMCPollController  
DMCPollInterval  
DMCPollMethod  
DMCPrefix  
DMCRefreshDataRecord  
DMCScale  
DMCSuffix  
DMCUseSystemTimer

#### Events

DMCError  
DMCResponse

### DMCPoll Property Descriptions

#### **DMCBevelInner**

#### **Description**

Used with or without the DMCBevelOuter property to give the DMCPoll control a 3D appearance.

## Usage

[form.][control.]**DMCBevelInner**[= setting]

## Setting

The DMCBevelInner property settings are:

<b>BevelInnerNone</b>	None.
<b>BevelInnerInset</b>	Inset. This is the default setting.
<b>BevelInnerRaised</b>	Raised.

## Remarks

If either the DMCBevelInner or DMCBevelOuter properties are set to any value other than 0 (none), the ForeColor and BackColor properties will have no effect. If a 3D appearance is not desired or if you wish to change either the foreground or background colors of the DMCPoll control, both the DMCBevelInner or DMCBevelOuter properties should be set to 0 (none). This property may be used with or without the BorderStyle property.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCBevelOuter**

## Description

Used with or without the DMCBevelInner property to give the DMCPoll control a 3D appearance.

## Usage

[form.][control.]**DMCBevelOuter**[= setting]

## Setting

The DMCBevelOuter property settings are:

<b>BevelOuterNone</b>	None.
<b>BevelOuterInset</b>	Inset. This is the default setting.
<b>BevelOuterRaised</b>	Raised.

## Remarks

The DMCBevelOuter property is closely related to the [DMCBevelWidth](#) property. That is, the [DMCBevelWidth](#) property determines the width of the outer bevel.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

## **DMCBevelWidth**

### **Description**

Used to set the width of the bevel used by the DMCBevelInner property to give the DMCPoll control a 3D appearance.

### **Usage**

*[form.][control.]DMCBevelWidth[= width]*

### **Setting**

The DMCBevelWidth property setting must be an integer between 1 and 30 inclusive.

### **Remarks**

The DMCBevelWidth property is closely related to the [DMCBevelOuter](#) property. That is, the DMCBevelWidth property determines the width of the outer bevel. The [DMCBevelOuter](#) property must be set to non-zero in order for the DMCBevelWidth property to have any effect.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCBorderWidth**

### **Description**

Used to set the width of the bevel used by the DMCBevelInner property to give the DMCPoll control a 3D appearance.

### **Usage**

*[form.][control.]DMCBorderWidth[= width]*

### **Setting**

The DMCBorderWidth property setting must be an integer between 1 and 30 inclusive.

### **Remarks**

The DMCBorderWidth property is closely related to the [DMCBevelInner](#) property. That is, the DMCBorderWidth property determines the width of the inner bevel. The [DMCBevelInner](#) property must be set to non-zero in order for the DMCBevelWidth property to have any effect.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long



---

## **DMCCaptionPosition**

### **Description**

Used to position the display of the caption in the DMCPoll control.

### **Usage**

[*form.*][*control.*]**DMCCaptionPosition**[= *position*]

### **Setting**

The DMCCaptionPosition property settings are:

<b>CaptionPositionNone</b>	None.
<b>CaptionPositionTop</b>	Top. This is the default setting.
<b>CaptionPositionLeft</b>	Left.
<b>CaptionPositionBottom</b>	Bottom.
<b>CaptionPositionRight</b>	Right.

### **Remarks**

If the Caption property has not been set, the DMCCaptionPosition property will have no effect. To disable the display of the caption, set the DMCCaptionPosition property to **CaptionPositionNone**

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCCommand**

### **Description**

Used by the DMCPoll control to poll or interrogate the Galil motion controller. The DMCCommand property is sent to the Galil motion controller at a specified interval (see the [DMCPollInterval](#) property) and the result is displayed by the DMCPoll control.

### **Usage**

[*form.*][*control.*]**DMCCommand**[= *command*]

### **Setting**

The setting for the DMCCommand property may be any valid Galil language command that returns a response. Note that a Galil language command is a string.

### **Remarks**

If there is an error resulting from sending the specified Galil language command, either because of a syntax error or a communications time-out, the application program will be notified through the [DMCError](#) event. Once an error has been registered, the DMCPoll control will no longer poll the Galil motion controller (that is, the [DMCPollController](#) property will be set to False) unless the [DMCIgnoreError](#) property is set to True.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the model Galil motion controller installed.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

String

---

## **DMCController**

### Description

The number corresponding to the Galil motion controller registered in the Windows registry database.

### Usage

[form.][[control.]]**DMCController**[= controller]

### Setting

The setting for the DMCController property must be an integer between 1 and 16. The default setting is 1.

### Remarks

This property sets the poll window to communicate to the specific controller number in the Galil registry. Note: the controller must also be connected to a shell control.

### Note

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

### Data Type

Long

---

## **DMCDataRecordItem**

### Description

Used to select which data item from the data record to use for polling when the DMCPollMethod property is set to DataRecordAccess.

### Usage

[form.][[control.]]**DMCDataRecordItem**[= setting]

### Setting

The DMCDataRecordItem property settings are:

**DataRecordItemAnalogInput1**  
**DataRecordItemAnalogInput2**  
**DataRecordItemAnalogInput3**  
**DataRecordItemAnalogInput4**  
**DataRecordItemAnalogInput5**  
**DataRecordItemAnalogInput6**  
**DataRecordItemAnalogInput7**  
**DataRecordItemAnalogInput8**

DataRecordItemCoordinatedMoveDistance  
DataRecordItemCoordinatedMoveDistanceT  
DataRecordItemCoordinatedMoveStatus  
DataRecordItemCoordinatedMoveStatusT  
DataRecordItemEAxisAuxillaryPosition  
DataRecordItemEAxisMotorPosition  
DataRecordItemEAxisPositionError  
DataRecordItemEAxisReferencePosition  
DataRecordItemEAxisStatus  
DataRecordItemEAxisStopCode  
DataRecordItemEAxisSwitches  
DataRecordItemEAxisTorque  
DataRecordItemEAxisVelocity  
DataRecordItemErrorCode  
DataRecordItemFAxisAuxillaryPosition  
DataRecordItemFAxisMotorPosition  
DataRecordItemFAxisPositionError  
DataRecordItemFAxisReferencePosition  
DataRecordItemFAxisStatus  
DataRecordItemFAxisStopCode  
DataRecordItemFAxisSwitches  
DataRecordItemFAxisTorque  
DataRecordItemFAxisVelocity  
DataRecordItemGAxisAuxillaryPosition  
DataRecordItemGAxisMotorPosition  
DataRecordItemGAxisPositionError  
DataRecordItemGAxisReferencePosition  
DataRecordItemGAxisStatus  
DataRecordItemGAxisStopCode  
DataRecordItemGAxisSwitches  
DataRecordItemGAxisTorque  
DataRecordItemGAxisVelocity  
DataRecordItemGeneralInput0  
DataRecordItemGeneralInput1  
DataRecordItemGeneralInput2  
DataRecordItemGeneralInput3  
DataRecordItemGeneralInput4  
DataRecordItemGeneralInput5  
DataRecordItemGeneralInput6  
DataRecordItemGeneralInput7  
DataRecordItemGeneralInput8  
DataRecordItemGeneralInput9  
DataRecordItemGeneralOutput0  
DataRecordItemGeneralOutput1  
DataRecordItemGeneralOutput2  
DataRecordItemGeneralOutput3  
DataRecordItemGeneralOutput4  
DataRecordItemGeneralOutput5  
DataRecordItemGeneralOutput6  
DataRecordItemGeneralOutput7  
DataRecordItemGeneralOutput8  
DataRecordItemGeneralOutput9  
DataRecordItemGeneralStatus  
DataRecordItemHAxisAuxillaryPosition  
DataRecordItemHAxisMotorPosition

DataRecordItemHAxisPositionError	
DataRecordItemHAxisReferencePosition	
DataRecordItemHAxisStatus	
DataRecordItemHAxisStopCode	
DataRecordItemHAxisSwitches	
DataRecordItemHAxisTorque	
DataRecordItemHAxisVelocity	
DataRecordItemSampleNumber	SampleNumber. This is the default setting
DataRecordItemSegmentCount	
DataRecordItemSegmentCountT	
DataRecordItemHAxisAuxillaryPosition	
DataRecordItemHAxisMotorPosition	
DataRecordItemHAxisPositionError	
DataRecordItemHAxisReferencePosition	
DataRecordItemHAxisStatus	
DataRecordItemHAxisStopCode	
DataRecordItemHAxisSwitches	
DataRecordItemHAxisTorque	
DataRecordItemHAxisVelocity	
DataRecordItemWAxisAuxillaryPosition	
DataRecordItemWAxisMotorPosition	
DataRecordItemWAxisPositionError	
DataRecordItemWAxisReferencePosition	
DataRecordItemWAxisStatus	
DataRecordItemWAxisStopCode	
DataRecordItemWAxisSwitches	
DataRecordItemWAxisTorque	
DataRecordItemWAxisVelocity	
DataRecordItemWAxisAuxillaryPosition	
DataRecordItemXAxisMotorPosition	
DataRecordItemXAxisPositionError	
DataRecordItemXAxisReferencePosition	
DataRecordItemXAxisStatus	
DataRecordItemXAxisStopCode	
DataRecordItemXAxisSwitches	
DataRecordItemXAxisTorque	
DataRecordItemXAxisVelocity	
DataRecordItemYAxisAuxillaryPosition	
DataRecordItemYAxisMotorPosition	
DataRecordItemYAxisPositionError	
DataRecordItemYAxisReferencePosition	
DataRecordItemYAxisStatus	
DataRecordItemYAxisStopCode	
DataRecordItemYAxisSwitches	
DataRecordItemYAxisTorque	
DataRecordItemYAxisVelocity	

## Remarks

Data Record Access refers to a feature, available on some Galil motion controllers such as the DMC-1700, which provides a data record consisting of general and axis specific status information such as input and output states, motor position, position error, and torque. The data record is physically refreshed by the controller at a specified interval such as 2ms, 4ms, etc.. The Data Record Access feature uses a different channel of communication other than the primary FIFO, so that the data record is always available. That is, it is not affected by other communications between the host PC and the controller. The data record refresh rate

is controlled by the Galil language command DR. The Data Record Access feature for the controller must be enabled in the Windows registry.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCFormat**

### Description

Used to format the display of numeric values in the DMCPoll control.

### Usage

*[form.][control.]DMCFormat*[= *format string*]

### Setting

The DMCFormat property is a string, which must be in the following format:

*[0]<number of digits left of decimal point>.<number of digits right of decimal point>*

The leading zero (0) in the format string indicates that the number should be formatted with leading zeros. The number of digits to the left of the decimal point must be less than or equal to 12. The number of digits to the right of the decimal point must be less than or equal to 4.

The default setting is 010.0, which is interpreted as 10 digits to the left of the decimal point, no digits to the right of the decimal point and display with leading zeros.

### Remarks

Numeric data is rounded to the nearest decimal point if it has a fractional value. Numeric data is never truncated even if the format string does not provide enough digits to the left of the decimal point to accommodate it. That is, the value of the data will override the specified number of digits to the left of the decimal point. Negative numeric data is prefixed with a minus sign (-).

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

String

---

## **DMCIgnoreError**

### Description

Used to control the notification of errors.

### Usage

*[form.][control.]DMCIgnoreError*[= {True | False}]

### Setting

The DMCIgnoreError property settings are:

Setting	Description
True	Ignore errors.
False	Do not ignore errors. This is the default setting.

### Remarks

If the DMCIgnoreError property is set to False and an error occurs, the application is notified through the [DMCError](#) event and the [DMCPollController](#) property will be set to False, effectively shutting down communications between the DMCPoll control and the Galil motion controller.

If the DMCIgnoreError property is set to True, the DMCPoll control will not notify the application of any errors which may occur. Likewise, polling is not suspended on recognition of an error.

This property should be used with some caution as important information regarding an error condition may not be sent to the application program. The DMCIgnoreError property is best used when the application uses many DMCPoll or related controls and it is possible or even likely that the DMCPoll control may experience a communications time-out while waiting for another control to complete its activity with the Galil motion controller.

### Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

### Data Type

Integer (Boolean)

---

## **DMCOffset**

### Description

Used to add a constant value to the display of numeric data in the DMCPoll control.

### Usage

*[form.][control.]DMCOffset[= constant value]*

### Setting

The setting may be any fixed decimal number. The default setting is zero (0).

### Remarks

The DMCOffset property can be used to change the display of current position information to reflect a relative position without actually redefining the absolute position. If the [DMCScale](#) property is also being used, the order of precedence is (response \* scale) + offset.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Single or Float

---

## **DMCOutputPosition**

### **Description**

Used to position the display of the output in the DMCPoll control.

### **Usage**

*[form.][control.]DMCOutputPosition[= position]*

### **Setting**

The DMCOutputPosition property settings are:

<b>OutputPositionLeft</b>	Left
<b>OutputPositionRight</b>	Right.
<b>OutputPositionCenter</b>	Center. This is the default setting.

### **Remarks**

The output from most Galil language commands used for polling is numeric. This should be taken into consideration when positioning the display of the output in the DMCPoll control.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCPollController**

### **Description**

Used to start or stop polling the Galil motion controller.

### **Usage**

*[form.][control.]DMCPollController[= {True | False}]*

### **Setting**

The DMCPollController property settings are:

<b>True</b>	Begins polling the Galil motion controller.
<b>False</b>	Ends polling the Galil motion controller. This is the default setting.

### **Remarks**

When the DMCPollController property is set to True, the DMCPoll control will poll the Galil motion controller with the Galil language command specified in the [DMCCommand](#) property every n milliseconds where n is equal to the setting for the [DMCPollInterval](#) property.

### **Note**

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

## Data Type

Integer (Boolean)

---

## **DMCPollInterval**

### Description

The number of milliseconds between submitting Galil language commands to the Galil motion controller.

### Usage

[form.][control.]**DMCPollInterval**[= interval]

### Setting

The DMCPollInterval property setting must be an integer within the range 50 through 10,000. The default setting is 500.

### Remarks

The DMCPollInterval property should be set to the highest value as is practical for the application. The faster the polling (the lower the setting), the more resources are consumed.

### Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCPollMethod**

### Description

Used to control how the DMCPoll control requests data from the Galil motion controller.

### Usage

[form.][control.]**DMCPollMethod**[= method]

### Setting

The DMCPollMethod property settings are:

<b>PollMethodCommand</b>	Use the primary communication channel for polling. This is the default setting.
<b>PollMethodDataRecordItem</b>	Use the secondary communication channel for polling.

### Remarks

Data Record Access refers to a feature, available on some Galil motion controllers such as the DMC-1700, which provides a data record consisting of general and axis specific status information such as input and output states, motor position, position error, and torque. The data record is physically refreshed by the controller at a specified interval such as 2ms, 4ms, etc.. The Data Record Access feature uses a different channel of communication other than the primary FIFO, so that the data record is always available. That is, it is not affected by other communications between the host PC and the controller. The data record refresh rate is controlled by the Galil language command DR. The Data Record Access feature for the controller must be enabled in the Windows registry.



## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCPrefix**

### Description

Used to add a prefix to the display of numeric or string data in the DMCPoll control.

### Usage

[form.][control.]**DMCPrefix**[= *prefix*]

### Setting

The setting for the DMCPrefix property may be any string. The default setting is no prefix.

### Remarks

The DMCPrefix property may be used with or without the [DMCSuffix](#) property. The width of the DMCPoll control must be large enough to accommodate the prefix plus the suffix, if used, plus the data returned from the Galil language command.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

String

---

## **DMCRefreshDataRecord**

### Description

Used to control the refresh of the Galil motion controller's Data Record when the DMCPollMethod property is set to "DataRecordAccess".

### Usage

[form.][control.]**DMCRefreshDataRecord**[= {True | False}]

### Setting

The DMCIgnoreError property settings are:

<b>True</b>	Refresh the data record before retrieving the data record item. This is the default setting.
<b>False</b>	Do not refresh the data record before retrieving the data record item.

### Remarks

If there is only one DMCPoll control in your project, the DMCRefreshDataRecord property should always be set to True. However, if there are more than one DMCPoll controls in your project, only one should have this property set to True. Setting this property to True for more than one DMCPoll control will add unnecessary processing overhead to your project and your run-time performance will be severely impacted.

The DMCPoll control with the smallest DMCPollInterval should be the one with the DMCTrefreshDataRecord property set to True.

This property is only used if the DMCPollMethod property is set to "DataRecordAccess".

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Integer (Boolean)

---

## **DMCScale**

### Description

Used to scale the display of numeric data in the DMCPoll control.

### Usage

[form.][control.]DMCScale[= command]

### Setting

The setting may be any fixed decimal number. The default setting is zero (0).

### Remarks

The DMCScale property is usually used to convert position or error data from encoder counts to user units such as inches or millimeters. If the [DMCOffset](#) property is also being used, the order of precedence is (response \* scale) + offset.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

String

---

## **DMCSuffix**

### Description

Used to add a suffix to the display of numeric or string in the DMCPoll control.

### Usage

[form.][control.]DMCSuffix[= suffix]

### Setting

The setting for the DMCSuffix property may be any string. The default setting is no suffix.

### Remarks

The DMCSuffix property may be used with or without the [DMCPrefix](#) property. The width of the DMCPoll control must be large enough to accommodate the suffix plus the prefix, if used, plus the data returned from the Galil language command.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

String

---

## **DMCUseSystemTimer**

## Description

Used to determine which type of timer to use for polling.

## Usage

[form.][control.]**DMCUseSystemTimer**[= {True | False}]

## Setting

The DMCUseSystemTimer property settings are:

<b>True</b>	Use a high-resolution system timer for polling.
<b>False</b>	Use a standard windows timer for polling. This is the default setting.

## Remarks

The DMCPoll control by default uses a standard Windows timer to poll the Galil motion controller. The standard Windows timer is not very accurate, but it does not consume much in the way of resources. As an option, the DMCPoll control can use a high-resolution system timer to poll the controller. The high-resolution timer is accurate to within +/- 5ms and will allow you use much faster poll intervals. The trade-off is that this type of timer consumes much more resources. It is not recommended that more than four DMCPoll controls be used simultaneously with the DMCUseSystemTimer property set to True. Most Windows versions will allow up to eight.

## Note

This property may only be used when the DMCPollController property is set to False. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

## DMCPoll Events

DMCError

DMCResponse

## **DMCError**

## Description

Used by the DMCPoll control to notify the application program of an error.

## Syntax

Sub *ctlname*\_**DMCError**(*ErrorNumber*As Long, *ErrorMessage*As String)

## Remarks

The two most frequent errors are:

ErrorNumber	Description
-1	A time-out occurred.
-2	Galil language command error.

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCPoll control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.
3. The Galil motion controller is waiting for a trippoint, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trippoint is cleared. This is not an error.

---

## **DMCResponse**

### **Description**

Used by the DMCPoll control to notify the application program of a response from the Galil motion controller.

### **Syntax**

Sub *ctlname*\_**DMCResponse**(*ResponseAs* String)

### **Remarks**

Response contains the actual response from the Galil motion controller. It does not use any of the formatting properties such as [DMCFormat](#), [DMCOffset](#), [DMCPrefix](#), [DMCScale](#), or [DMCSuffix](#).

## **DMCPoll Examples**

### **Example Setting the DMCCommand Property**

In this example, The DMCCommand property will be set to "TPX", which is tell position of the X axis.

```

` Poll for the position of the x axis
DMCPoll11.DMCCCommand = "TPX"

` Set the polling interval to 100 milliseconds
DMCPoll11.DMCPollInterval = 100

` Start polling
DMCPoll11.DMCPollController = True

```

### Example: Using the Formatting Properties

In this example, the various formatting properties will be used to customize the display of the data in the DMCPoll control.

```

` Set the format
DMCPoll11.DMCFormat = "8.2"

` Set the offset
DMCPoll11.DMCOffset = 1000

` Set the prefix
DMCPoll11.DMCPrefix = "Speed: "

` Set the scale
DMCPoll11.DMCScale = .015

` Set the suffix
DMCPoll11.DMCSuffix = "RPM"

```

### Example: Responding to the DMCError Event

In this example, the application program reports on errors as sent by the Galil motion controller.

```

Sub DMCPoll11_DMCError (ErrorNumber As Long, ErrorMessage As String)
    Dim RC As Integer
    RC = MsgBox(ErrorMessage, MB_ICONEXCLAMATION, "DMC Error")
End Sub

```

### Example: Responding to the Response Event

In this example, the DMCPollController property has been set to True, and the application program is placing the responses from the Galil motion controller in a text control named ResponseOutput.

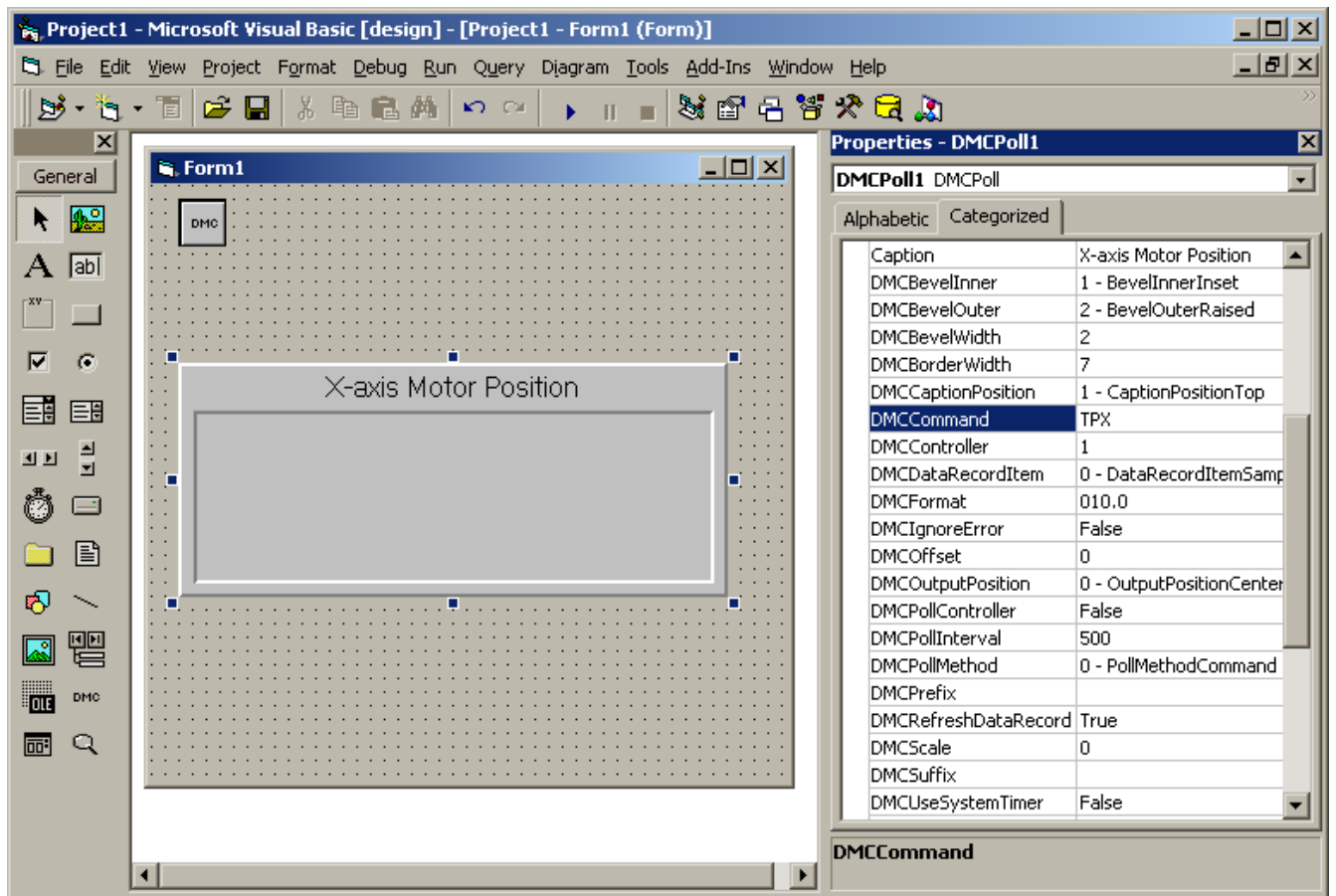
```

Sub DMCPoll11_DMCRresponse (Response As String)
    ResponseOutput.Text = Response
End Sub

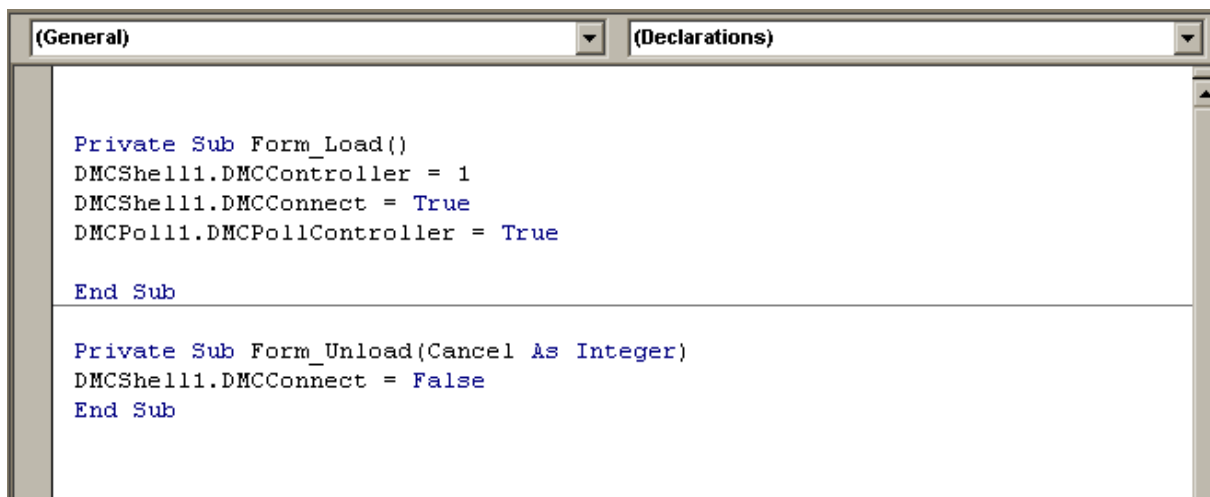
```

## Program Example

The polling object should be used for updating displays and monitoring flag variables. Although the timing property can be set to a certain value the actual time between updates might be larger. Some variation may occur depending on other activity within Windows. To setup a polling window just draw it on the form, set the property to define the data to be displayed, then activate the polling window in the form load procedure. This example shows a polling window set up to display X axis position with a 500 ms interval:



The DMCController property must be set to the same controller as the DMCSHELL. Set the property "DMCPoll1.DMCPollController = True" after the DMCSHELL is connected.

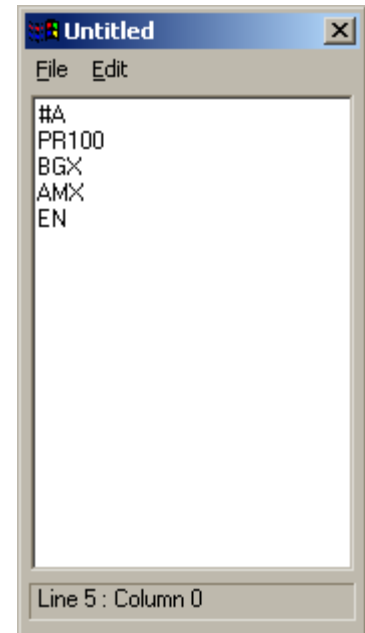
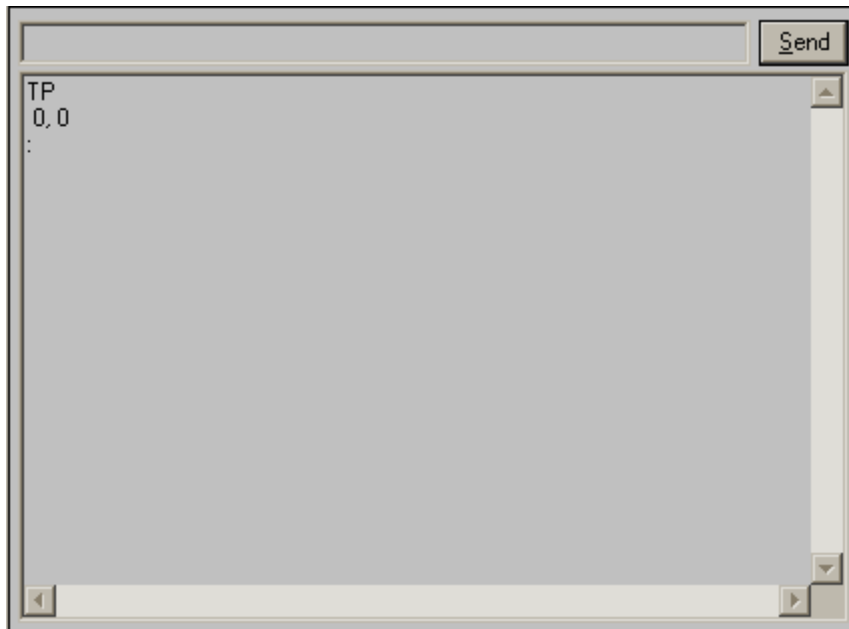


**THIS PAGE LEFT BLANK INTENTIONALLY**



# Chapter 4 – DMCTerminal Control

The terminal object provides a convenient method of communicating with the controller through typing in commands. It includes a full-screen editor, as well as the ability to upload and download application programs.



## Galil Properties, Methods, and Events

### Properties

DMCCommand  
DMCController  
DMCEnableControlCharacters  
DMCLastCommand  
DMCNoCommandWindow  
DMCPollController  
DMCPollInterval  
DMCResponse

### Methods

DMCWriteToResponseBox  
ClickSendButton  
Clear

### Events

DMCError  
DMCResponse  
DMCTransaction

## DMCTerminal Property Descriptions

### DMCController

#### Description

The number corresponding to the Galil motion controller registered in the Windows registry database.



## Usage

[form.][control.]**DMCController**[= controller]

## Setting

The setting for the DMCController property must be an integer between 1 and 16. The default setting is 1.

## Remarks

A Galil motion controller can be registered in the Windows registry database by using the DMCREG32 program for Windows 98SE, NT4.0, 2000, or XP.

## Note

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

## Data Type

Long

---

## **DMCCommand**

## Description

Used to send Galil language commands to the Galil motion controller.

## Usage

[form.][control.]**DMCCommand**[= command]

## Setting

The setting for the DMCCommand property may be any valid Galil language command. Note that a Galil language command is a string.

## Remarks

Once the DMCCommand property has been set, the Galil language command is sent to the Galil motion controller. If the Galil language command has a response, the [DMCResponse](#) property will be set by the DMCTerminal control. If there is an error, the application program will be notified through the [DMCError](#) event. A question mark ("?",) in the DMCResponse property also denotes an unrecognized command or a command error. Once the command has completed, the DMCCommand property is set to a NULL string ("") by the DMCTerminal control.

If there is a response from the Galil motion controller, the response will be displayed in the DMCTerminal output window.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the model Galil motion controller installed.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

String

---

## **DMCEnableControlChars**

### **Description**

Used to enable or disable the sending of control key combinations to the Galil motion controller, such as CTRL+R, CTRL+V (or ^R^V) which is used to display the firmware revision.

### **Usage**

[form.][control.]**DMCEnableControlChars**[= {True | False}]

### **Setting**

The DMCEnableControlChars settings are:

<b>True</b>	Enable control key combinations.
<b>False</b>	Disable control key combinations. This is the default setting.

### **Remarks**

When the DMCEnableControlChars property is set to True, the DMCTerminal control may interfere with keyboard accelerator keys in your application program. You must be careful when designing menus which appear on the same form as the DMCTerminal control, not to use the control key for any keyboard accelerators.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Integer (Boolean)

---

## **DMCLastCommand**

### **Description**

Used to store the last Galil language command sent to the Galil motion controller.

### **Usage**

command = [form.][control.]**DMCLastCommand**

### **Setting**

None. The DMCLastCommand property is read-only.

### **Remarks**

The DMCLastCommand property contains the last Galil language command that was sent to the Galil motion controller. That is, it is automatically set to the value of the DMCCCommand property after the command has been sent to the controller. This property may be useful in implementing a repeat command function for the DMCTerminal control. By setting the value of the DMCCCommand property to the value of the DMCLastCommand property, the DMCTerminal control will effectively repeat the last command that was sent to the controller.

### **Note**

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

## Data Type

String

---

### **DMCNoCommandWindow**

#### Description

Used to show or hide the “Command” window and the “Send” button.

#### Usage

[form.][control.]**DMCNoCommandWindow**[= {True | False}]

#### Setting

The DMCNoCommandWindow property settings are:

<b>True</b>	Hide the “Command” window and the “Send” button.
<b>False</b>	Show the “Command” window and the “Send” button. This is the default setting.

#### Remarks

The “Command” window at the top of the DMCTerminal control is used to enter commands to send to the Galil motion controller. The “Send” button, also located at the top of the DMCTerminal control and to the right of the “Command” window, is used to actually send the entered commands to the Galil motion controller (the enter or return key duplicates this function).

When the DMCNoCommandWindow property is set to True, both the “Command” window and the “Send” button are hidden. Commands may now be typed directly on the “Response” window and the enter or return key may be used to send the commands to the Galil motion controller. This style may be preferred when there is not a great deal of form space to work with on a given form.

If both the DMCNoCommandWindow and [DMCPollController](#) properties are set to True, be careful to not run any DMC programs which issue a lot of messages to the DMCTerminal control. You may find yourself in a situation where it is impossible to type in any more commands because the “Response” window is being continually updated.

#### Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

### **DMCPollController**

#### Description

Used to start or stop polling the Galil motion controller.

#### Usage

[form.][control.]**DMCPollController**[= {True | False}]

#### Setting

The DMCPollController property settings are:

<b>True</b>	Begins polling the Galil motion controller.
<b>False</b>	Ends polling the Galil motion controller. This is the default setting.

### Remarks

If the `DMCPollController` property is set to `True`, the `DMCTerminal` control will poll the Galil motion controller for unsolicited responses every `n` milliseconds where `n` is equal to the setting for the `DMCPollInterval*DMCPollInterval>Main` property.

Unsolicited responses are responses (output) from the Galil motion controller which result from executing a Galil language application program. Polling the Galil motion controller is not necessary to obtain responses during normal interactive communication. For this, there is the `DMCResponse` property. However, if you intend to monitor output which results from executing a Galil language program from within your application program, you must use polling.

If there is an unsolicited response from the Galil motion controller and polling is active, the unsolicited response will be displayed in the `DMCTerminal` output window.

The `DMCPollInterval` property should be set to the highest value as is practical for the application. The faster the polling (the lower the setting), the more resources are consumed.

### Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

### Data Type

Integer (Boolean)

---

## **DMCPollInterval**

### Description

The number of milliseconds between submitting Galil language commands to the Galil motion controller.

### Usage

`[form.][control.]DMCPollInterval[= interval]`

### Setting

The `DMCPollInterval` property setting must be an integer within the range 50 through 10,000. The default setting is 500.

### Remarks

If the `DMCPollController` property is set to `True`, the `DMCTerminal` control will poll the Galil motion controller for unsolicited responses every `n` milliseconds where `n` is equal to the setting for the `DMCPollInterval` property.

Unsolicited responses are responses (output) from the Galil motion controller which result from executing a Galil language application program. Polling the Galil motion controller is not necessary to obtain responses during normal interactive communication. For this, there is the `DMCResponse` property. However, if you intend to monitor output which results from executing a Galil language application program from within your application program, you must use polling.

If there is an unsolicited response from the Galil motion controller and polling is active, the unsolicited response will be displayed in the DMCTerminal output window.

The DMCPollInterval property should be set to the highest value as is practical for the application. The faster the polling (the lower the setting), the more are resources consumed.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCResponse**

### Description

Used to access the response to a Galil language command (sent via the [DMCCommand](#) property) from the Galil motion controller.

### Usage

*response* = [*form*]. [*control*]. **DMCResponse**

### Setting

None. The DMCResponse property is read-only.

### Remarks

When the [DMCCommand](#) property has been set, a Galil language command is sent to the Galil motion controller. If the Galil language command has a response, the DMCResponse property will be set by the DMCTerminal control. If there is an error, the application program will be notified through the [DMCError](#) event. A question mark (" ? ") in the DMCResponse property also denotes an unrecognized command or a command error.

If there is a response from the Galil motion controller, the response will be displayed in the DMCTerminal output window.

The DMCResponse property is overwritten each time a command is sent to the Galil motion controller.

### Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

### Data Type

String

---

## DMCTerminal Event Descriptions

### Galil Events

DMCError  
DMCResponse  
DMCTransaction

### DMCError

#### Description

Used by the DMCTerminal control to notify the application program of an error.

#### Syntax

Sub *ctlname*\_**DMCError**(*ErrorNumber*As Long, *ErrorMessage*As String)

#### Remarks

The two most frequent errors are:

ErrorNumber	Description
-1	A time-out occurred.
-2	Galil language command error.

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCTerminal control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.

3. The Galil motion controller is waiting for a trippoint, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trippoint is cleared. This is not an error.

---

## **DMCResponse**

### **Description**

Used by the DMCTerminal control to notify the application program of an unsolicited response from the Galil motion controller. Unsolicited responses from the Galil motion controller can be periodically checked by using the DMCPollController property.

### **Syntax**

Sub *ctlname*\_**DMCResponse**(ResponseAs String)

### **Remarks**

Response contains the unsolicited response from the Galil motion controller. Unsolicited responses can occur when the Galil motion controller is executing a Galil language program and the Galil language program uses Galil language commands which evoke responses.

---

## **DMCTransaction**

### **Description**

Used by the DMCTerminal control to notify the application program that a transaction between the application program and the Galil motion controller has occurred.

### **Syntax**

Sub *ctlname*\_**DMCTransaction**(*Command* As String, *Response* As String)

### **Remarks**

The DMCTransaction event may be used by the application program to log all commands and responses to/from the Galil motion controller.

---

## **DMCTerminal Method Descriptions**

### **Galil Methods**

Clear  
ClickSendButton

### **Clear**

### **Description**

Used to clear the terminal display of any responses.

### **Usage**

[*form.*][*control.*]**Clear**

### **Remarks**

Once the terminal display is cleared, previous responses from the Galil motion controller are no longer available.

## Note

This method may be only be used at run time.

---

## **ClickSendButton**

### Description

Used to simulate clicking the Send button or pressing the Alt+S key combination.

### Usage

`[form.][control.]ClickSendButton`

### Remarks

OLE control containers such as Microsoft Visual C++ which do not send the return, esc, or mnemonic keys to OLE custom controls may need to use this method to simulate clicking the Send button. You may find it helpful to create a button on your form which uses the Alt+S mnemonic. When this button is clicked or the mnemonic entered, your button control handler would invoke this method. Your button could even be created invisible so as not to clutter up your form.

Please note that Visual Basic does not have this problem.

## Note

This method may be only be used at run time.

---

## DMCTerminal Examples

### Sending a Command to the Controller:

In this example, a command will be sent to the Galil motion controller and its response will be checked. The command is input by the user from a text control used as an input field and the response is output to the user to a text control used as an output field.

```
' Send the command to the Galil motion controller as input by
'   the user
DMCTerminal1.DMCCCommand = TextIn.text

' Place the response from the Galil motion controller in the
'   output text control
TextOut.text = DMCTerminal1.DMCResponse
```

### Responding to the DMCError Event:

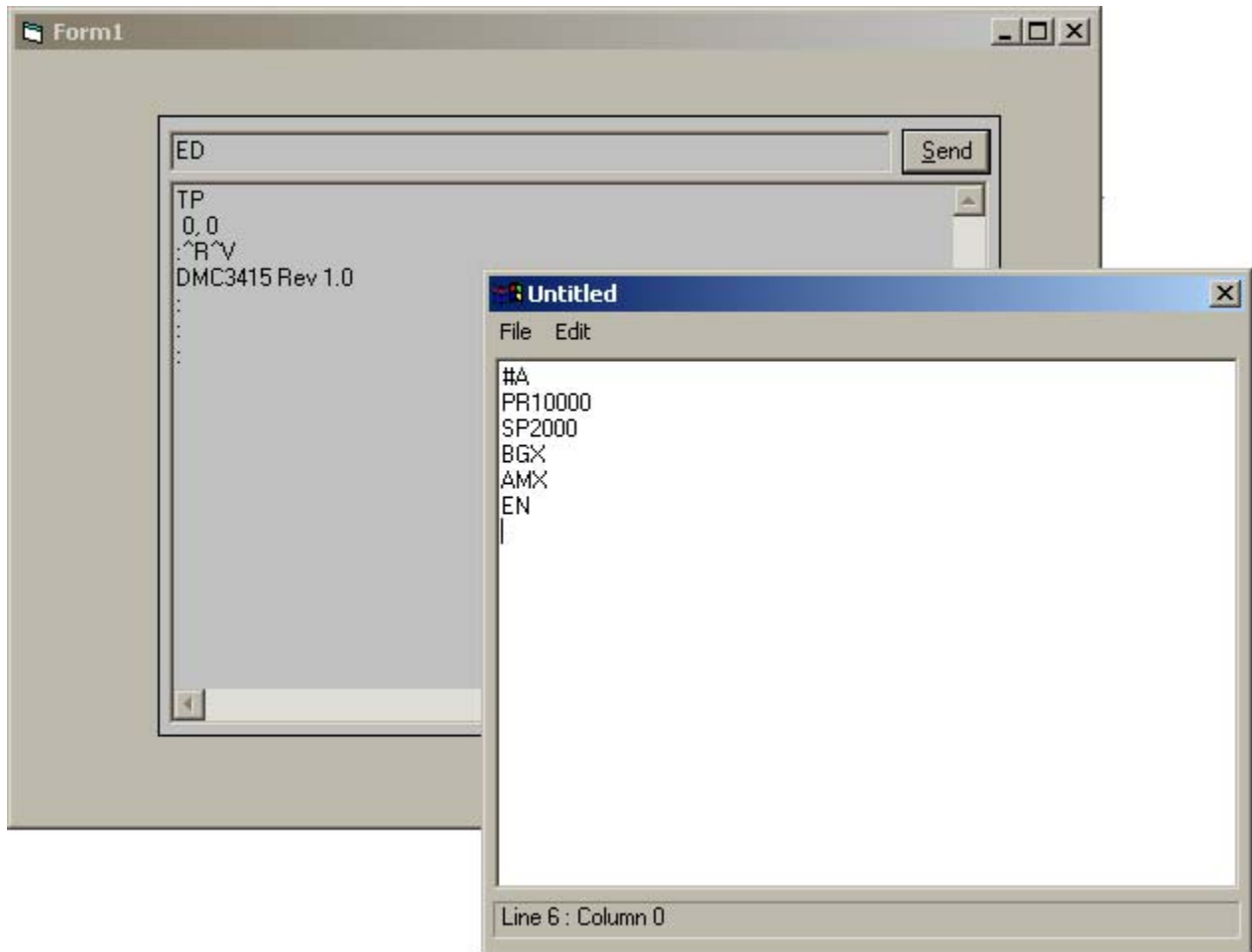
In this example, the application program reports on errors as sent by the Galil motion controller.

```
Sub DMCMove1_DMCError (ErrorNumber As Long, ErrorMessage As String)
    Dim RC As Integer
    RC = MsgBox(ErrorMessage, MB_ICONEXCLAMATION, "DMC Error")
End Sub
```



## Example Program

To use the terminal object just draw it on the form and run the program. ( Information on the advanced options is available in the help file. ) This is what the terminal looks like at run time after “ED” is entered, bringing up the built in editor:





# Chapter 5 - DMCTerminal2 Control

## Description

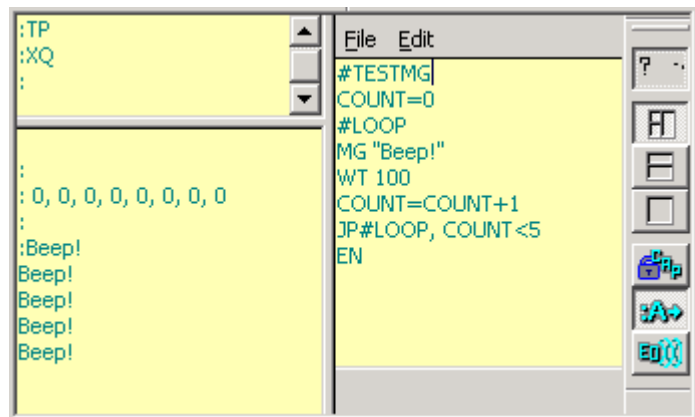
The DMCTerminal2 ActiveX control is used to communicate interactively with Galil motion controllers. This control allows the user to type Galil language commands in a window and see the Galil motion controller response. By exposing various properties and methods, such as DMCCCommand, an application program can communicate programmatically with the Galil motion controller.

The DMCTerminal2 control window has helpful features such as a toolbar and Syntax Help, and three possible terminal modes. The DMCTerminal2 control supports "F1" and "Shift+F1" help features.

### Smart Terminal with Editor

This terminal mode has three window panes separated by splitter bars:

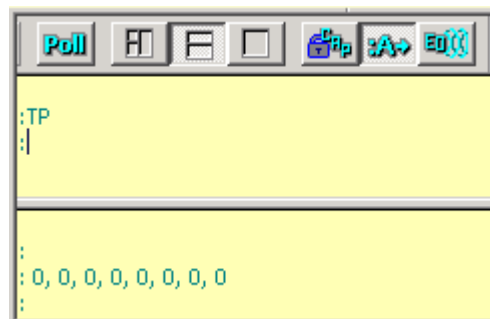
- (1) The Input or Command window.
- (2) The Output or Response window.
- (3) The Editor Window.



### Smart Terminal

This terminal mode has two window panes separated by a splitter bar:

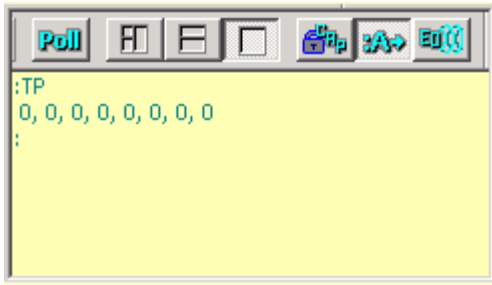
- (1) The Input or Command window.
- (2) The Output or Response window.



### Dumb Terminal

This terminal mode has a single pane:

- (1) The Dumb Terminal window.



## Note

The DMCTerminal2 ActiveX control must be used in conjunction with the DMCSHELL ActiveX Control. The DMCSHELL ActiveX control supplies the base support functions necessary for communicating with the Galil motion controller.

## Galil Properties, Methods, and Events

### Properties

DMCAAllowedTerminalMode  
DMCAAllowedToolbarAlignment  
DMCCCommand  
DMCCController  
DMCEditTermSplitterColPos  
DMCEditTermSplitterRowPos  
DMCEnableCapLock  
DMCEnableControlChars  
DMCEnableSyntaxHelp  
DMCEnableToolbar  
DMCLastCommand  
DMCResponse  
DMCPollcontroller  
DMCPollInterval  
DMCSmartTermSplitterRowPos

### Methods

Clear  
ClickSendButton  
DMCGetTerminalMode  
DMCGetToolbarPosition  
DMCOpenFileIntoEditor  
DMCSetTerminalMode  
DMCSetToolbarPosition  
DMCWriteToResponseBox

### Events

DMCError  
DMCResponse  
DMCTransaction

## DMCTerminal2 Property Descriptions

### DMCAAllowedTerminalMode

#### Description

The DMCAAllowedTerminalMode property determines what terminal modes will be allowed at run-time. This property allows the available terminal modes to be restricted.

#### Usage

[*form*.][*control*.]DMCAAllowedTerminalMode[= *nAllowedTerminalMode*]  
*nAllowedTerminalMode* = [*form*.][*control*.]DMCAAllowedTerminalMode

## Settings

Valid DMCAAllowedTerminalModes property settings are:

<b>SmartTermWithEditOnly</b>	0x01 (1 decimal)
<b>SmartTermOnly</b>	0x02 (2 decimal)
<b>DumbTermOnly</b>	0x04 (4 decimal)
<b>AnyTerm</b>	0x0F (15 decimal)

## Remarks

The DMCAAllowedTerminalMode property can be accessed at both run-time and design-time.

---

## **DMCAAllowedToolbarAlignment**

### Description

This property sets the allowable toolbar alignment.

### Usage

`[form.][control.]DMCAAllowedToolbarAlignment[= nAlignment]`  
`nAlignment = [form.][control.]DMCAAllowedToolbarAlignment`

### Setting

Valid DMCToolbarAlignment property settings are as follows: Any combination of the first four settings is valid or AllowDockAny.

<b>AllowLeft</b>	0x1000 (4096 decimal)
<b>AllowDockTop</b>	0x2000 (8192 decimal)
<b>AllowDockRight</b>	0x4000 (16384 decimal)
<b>AllowDockBottom</b>	0x8000 (32768 decimal)
<b>AllowDockAny</b>	0xF000 (61440 decimal)

## Remarks

The DMCToolbarAlignment property can be used at both design-time and run-time. This property sets the allowable dock sites that the user can dock to when the toolbar is dragged with the mouse to a new docksite or when the toolbar is docked programmatically using the [DMCSetDockToolbar\(fix\)](#) method. If the toolbar is enabled (see [DMCEnableToolbar](#) property) and a call is made to DMCToolbarAlignment limiting the docking location, the toolbar will be re-docked to a valid docksite if the current site is no longer valid.

---

## **DMCCommand**

### Description

Used to send Galil language commands to the Galil motion controller.

### Usage

`[form.][control.]DMCController[= command]`

## Setting

The setting for the DMCCCommand property may be any valid [Galil language](#) command. Note that a Galil language command is a string.

## Remarks

Once the DMCCCommand property has been set, the Galil language command is sent to the Galil motion controller. If the Galil language command has a response, the [DMCResponse](#) property will be set by the DMCTerminal control. If there is an error, the application program will be notified through the DMCTError event. A question mark ("?",) in the DMCResponse property also denotes an unrecognized command or a command error. Once the command has completed, the DMCCCommand property is set to a NULL string ("") by the DMCTerminal control. If there is a response from the Galil motion controller, the response will be displayed in the DMCTerminal [output window](#).

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the model Galil motion controller installed.

## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

---

## DMCController

### Description

The DMCController property stores the controller number that the DMCTerminal control will communicate with. The controller number is the number that corresponds to the desired Galil motion controller registered in the Windows registry database. To get information about the controllers that are currently registered in the Windows registry database, use the DMCRegistry control (e.g., the EditRegistry() function).

### Usage

```
[form.][control.]DMCController[= controller]  
nCurrentController =[form.][control.]DMCController
```

## Setting

The setting for the DMCController property must be an integer between 1 and 16. The default setting is 1.

## Remarks

A Plug and Play Galil motion controller id registered in the Galil Controller registry database automatically by the driver. Manually registered controllers can be registered in the Windows registry database by using EditRegistry() function in the DMCRegister control (DMCReg.ocx) or from the "Tools\Controller Registration Tool" menu selection in the DMCTerminal application (DMCTERM.EXE).

## Note

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

---

## **DMCEditTermSplitterColPos**

### **Description**

The DMCEditTermSplitterColPos property set/retrieves the position of the splitter bar (in pixels) that separates the Editor window from the Input/Output windows. The property can be used to persist the current configuration of the window between uses.

### **Usage**

[*form*.][*control*.]DMCEditTermSplitterColPos[= *nSplitterPos*]  
*nSplitterPos* = [*form*.][*control*.]DMCEditTermSplitterColPos

### **Setting**

*nSplitterPos* is an integer representing the column position in pixels

### **Note**

This property may be referenced at both design-time and run-time. Design-time changes are saved.

---

## **DMCEditTermSplitterRowPos**

### **Description**

The DMCEditTermSplitterRowPos property set/retrieves the position of the splitter bar (in pixels) that separates the Input/Output windows while in [Smart Terminal with Edit mode](#). The property can be used to persist the current configuration of the window between uses.

### **Usage**

[*form*.][*control*.]DMCEditTermSplitterRowPos[= *nSplitterPos*]  
*nSplitterPos* = [*form*.][*control*.]DMCEditTermSplitterRowPos

### **Setting**

*nSplitterPos* is an integer representing the row position in pixels

### **Remarks**

This property may be referenced at both design-time and run-time. Design-time changes are saved.

---

## **DMCEnableCapLock**

### **Description**

Used to set the cap lock on/off during DMCTerminal operation

### **Usage**

[*form*.][*control*.]DMCEnableCapLock [= *true/false*]

## Setting

**True** sets the caps lock on

**False** sets the caps lock off

---

## **DMCEnableControlChars**

### Description

The DMCEnableControlChars property is used to enable or disable the sending of control key combinations to the Galil motion controller, such as CTRL+R, CTRL+V (or ^R^V) that is used to display the firmware revision. When the property is set true, a caret will appear in the [Input window](#) when the control key is held down in combination with a keystroke. When the property is false, the control key will be consumed by the currently active window (e.g., CTRL+V will resume its usual behavior as an accelerator for PASTE) .

### Usage

```
[form.][control.]DMCEnableControlChars[= {True | False}]  
bEnableControlChars = [form.][control.]DMCEnableControlChars
```

### Setting

**True** enable the control characters

**False** disable the control characters

### Remarks

When the DMCEnableControlChars property is set to True, the DMCTerminal control may interfere with keyboard accelerator keys in your application program. You must be aware that when designing menus that appear on the same form as the DMCTerminal control, this property will interfere with keyboard accelerators.

False is the default setting for this property.

### Note

This property may be used at any time. This property is saved if set at design time.

---

## **DMCEnableSyntaxHelp**

### Description

The DMCEnableSyntaxHelp property enables/disables the DMCTerminal Control's [syntax help](#) feature. Please follow the previous link to read more about the syntax help feature.

### Usage

```
[form.][control.]DMCEnableSyntaxHelp[ = { True | False }]  
bSyntaxHelpEnabled = [form.][control.]DMCEnableSyntaxHelp
```

## Setting

- True**    enable the syntax help feature
- False**    disable the syntax help feature

## Remarks

If re-distributing the DMCTerminal control, please read the [distribution notes](#). The syntax help feature requires the registration of an ODBC data source and the installation of other files.

## Note

This property may be referenced at both design-time and run-time. Design-time changes are saved. The default value is false.

---

## **DMCEnableToolBar**

### Description

Used to enable and show the toolbar during DMCTerminal operation

### Usage

[[form](#).][[control](#).]DMCEnableToolBar [= [true/false](#)]

## Setting

- True**    enable and show the toolbar
- False**    disable and hide the toolbar

---

## **DMCLastCommand**

### Description

This property stores the last [Galil language](#) command sent to the Galil motion controller.

### Usage

sLastCommand = [[form](#).][[control](#).]DMCController

## Remarks

The DMCLastCommand property contains the last Galil language command that was sent to the Galil motion controller. That is, it is automatically set to the value of the DMCCCommand property after the command has been sent to the controller. This property may be useful in implementing a repeat command function for the DMCTerminal control. By setting the value of the DMCCCommand property to the value of the DMCLastCommand property, the DMCTerminal control will effectively repeat the last command that was sent to the controller.



## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

---

## **DMCResponse**

### **Description**

Used to access the response to a Galil language command (sent via the [DMCCommand](#) property) from the Galil motion controller.

### **Usage**

`sResponse = [form.][control.]DMCResponse`

### **Remarks**

When the DMCCommand property has been set, a [Galil language](#) command is sent to the Galil motion controller. If the Galil language command has a response, the DMCResponse property will be set by the DMCTerminal control. If there is an error, the application program will be notified through the DMCError event. A question mark ("?",) in the DMCResponse property also denotes an unrecognized command or a command error. While in Smart Terminal or Smart Terminal with Editor modes, a command error is automatically interrogated with the TC1 command and the response is appended to the string returned to the DMCResponse property.

If there is a response from the Galil motion controller, the response will be displayed in the DMCTerminal output window.

The DMCResponse property is overwritten each time a command is sent to the Galil motion controller using the DMCCommand property or when an unsolicited response is received from the controller via polling ([see DMCPollController](#)).

## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

---

## **DMCPollController**

### **Description**

This property is used to start or stop polling of the Galil motion controller.

Setting the property to true begins polling of the Galil motion controller. Setting the property false ends polling of the controller. False is the default setting.

### **Usage**

`[form.][control.]DMCPollController [= { True | False }]`  
`bPollController = [form.][control.]DMCPollController`

## Remarks

If the DMCPollController property is set to True, the DMCTerminal control will poll the Galil motion controller for unsolicited responses every n milliseconds where n is equal to the setting for the [DMCPollInterval property](#).

Unsolicited responses are responses (output) from the Galil motion controller that result from executing a [Galil language](#) application program. Polling the Galil motion controller is not necessary to obtain responses during normal interactive communication. For this, there is the DMCResponse property. However, if you intend to monitor output that results from executing a Galil language program from within your application program, you must use polling.

If there is an unsolicited response from the Galil motion controller and polling is active, the unsolicited response will be displayed in the DMCTerminal output window.

The DMCPollInterval property should be set to the highest value as is practical for the application. The faster the polling (the lower the property value setting), the more CPU resources are consumed by the application using the DMCTerminal control.

Other processes can automatically pause the polling. For example, when the [DMCCommand property](#) is used to send a command to the controller, polling is automatically pause while the command is processed.

## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established. The design time value is not saved as the property cannot be set true until a connection has been established.

---

## **DMCPollInterval**

### Description

This property sets the time-out value for a timer in milliseconds. Each time the timer fires, the DMCTerminal control queries the output buffer of the Galil motion controller to see if any unsolicited responses have occurred. If there are characters on the output buffer, they are read and placed in the [Outout window](#).

### Usage

```
[form.][control.]DMCPollInterval[= nInterval ]  
nInterval = [form.][control.]DMCPollController
```

### Setting

The DMCPollInterval property setting must be an integer within the range 1 through 10,000. The default setting is 50 milliseconds.

## Remarks

If the DMCPollController property is set to True, the DMCTerminal control will poll the Galil motion controller for unsolicited responses every n milliseconds where n is equal to the setting for the DMCPollInterval property.

Unsolicited responses are responses (output) from the Galil motion controller that result from executing a [Galil language](#) application program. Polling the Galil motion controller is not necessary to obtain responses during normal interactive communication. For commands that automatically produce a reponse from the controller, there is the DMCResponse property. However, if you intend to monitor output that results from

executing a Galil language program from within your application program, you must use polling.

If there is an unsolicited response from the Galil motion controller and polling is active, the unsolicited response will be displayed in the DMCTerminal output window.

The DMCPollInterval property should be set to the highest value as is practical for the application. The faster the polling (the lower the property value setting), the more CPU resources are consumed by the application using the DMCTerminal control.

Other processes can automatically pause the polling. For example, when the [DMCCommand property](#) is used to send a command to the controller, polling is automatically pause while the command is processed.

#### **Note**

This property may be used at any time. This property is saved if set at design-time.

---

### **DMCSmartTermSplitterRowPos**

#### **Description**

The DMCSmartTermSplitterRowPos property set/retrieves the position of the splitter bar (in pixels) that separates the Input window from the Output window while in Smart Terminal mode. The property can be used to persist the current configuration of the window between uses.

#### **Usage**

`[form.][control.]DMCSmartTermSplitterRowPos[= nSplitterPos]`  
`nSplitterPos = [form.][control.]DMCEditTermSplitterRowPos`

#### **Note**

This property may be referenced at both design-time and run-time. Design-time changes are saved.

---

## **DMCTerminal2 Method Descriptions**

### **Clear Method**

#### **Description**

The Clear method clears all of the terminal windows of text.

#### **Usage**

`[form.][control.]Clear( )`

#### **Remarks**

Once the terminal displays are cleared, the previously recorded commands or responses from the Galil motion controller are no longer available.

---

## **ClickSendButton\***

### **\*Method Obsolete**

#### **Description**

This method is obsolete and results in a non-operation. It has been left in the DMCTerm2.ocx interface for backwards compatibility of code. However, if this function was used previously, it is likely that the code calling this function will need to be modified to reflect the different functionality between the [previous Galil terminal control\(DMCTerm.ocx\)](#) and this control(DMCTerm2.ocx).

---

## **DMCGetToolbarPosition**

#### **Description**

The DMCGetToolbarPosition method returns the current docksite of the DMCTerminal Toolbar. The method's x and y arguments contain the screen coordinates of the toolbar (in pixels) if the toolbar is floating. Otherwise, x and y are returned set to -1. The DMCGetToolbarPosition return value will be one of the following:

<b>DockLeft</b>	0x01 (1 decimal)
<b>DockTop</b>	0x02 (2 decimal)
<b>DockRight</b>	0x04 (4 decimal)
<b>DockBottom</b>	0x08 (8 decimal)
<b>Floating</b>	0x0F (15 decimal)

#### **Usage**

*nDockSite* = [*form*].[*control*].DMCGetToolbarPosition( *x as Long, y as Long* )

#### **Remarks**

This method would be used most frequently to aid in persisting the last state of the DMCTerminal control between uses of an application.

---

## **DMCOpenFileIntoEditor**

#### **Description**

The DMCOpenFileIntoEditor method opens a text file into the DMCTerminal control's Editor window. If the terminal is not in [Smart Terminal With Editor](#) mode, the terminal is placed into this mode. DMCOpenFileIntoEditor returns false if the filepath provided in the sFileName argument does not point to a valid file or if the editor mode has been disallowed by the DMCAAllowedTerminalMode property.

#### **Usage**

*bSuccess* = [*form*].[*control*].DMCOpenFileIntoEditor( *sFileName as String, bReadOnly as Boolean* )

---

## **DMCSetTerminalMode**

### **Description**

The DMCSetTerminalMode method sets the current terminal mode, either [Smart Terminal](#) with Editor mode (SmartTermWithEdit = 0x01), the [Smart Terminal](#) mode (SmartTerm = 0x02), or the [Dumb Terminal](#) mode (DumbTerm = 0x04). If DMCTerminal successfully changes the terminal mode, the method returns true. If the requested terminal mode has been disallowed with [DMCAllowedTerminalMode](#) property, this method returns false.

### **Usage**

*bModeAllowed* = [*form.*][*control.*]**DMCSetTerminalMode**( *nNewTerminalMode* as Integer )

---

## **DMCSetToolbarPosition**

### **Description**

The DMCSetToolbarPosition method sets the current docksite of the DMCTerminal Toolbar. The method's x and y arguments contain the screen coordinates for the toolbar (in pixels) if the toolbar is to be floated. Otherwise, x and y parameters are ignored. The DMCSetToolbarPosition returns true if the method successfully docks the toolbar. If a DMCSetToolbarPosition call attempts to dock the toolbar to a site disallowed by the [DMCToolbarAlignment](#) property, the method returns false. The method's DockPos argument will be one of the following values:

<b>DockLeft</b>	0x01 (1 decimal)
<b>DockTop</b>	0x02 (2 decimal)
<b>DockRight</b>	0x04 (4 decimal)
<b>DockBottom</b>	0x08 (8 decimal)
<b>Floating</b>	0x0F (15 decimal)

### **Usage**

*nDockSite* = [*form.*][*control.*]**DMCSetToolbarPosition**( *DockPos* as DMCToolbarDockConstants, *x* as Long, *y* as Long )

### **Remarks**

This method would be used most frequently to aid in persisting the last state of the DMCTerminal control between uses of an application

---

## **DMCWriteToResponseBox**

### **Description**

The DMCWriteToResponseBox method places the string specified in the sString argument into the output window of the terminal. One possible use of this method is to use it in conjunction with the DMCSHELL control. DMCSHELL (DMCSHELL.ocx) can be configured to capture Galil Motion controller interrupts. The interrupt code

can be interpreted into a string that is then placed in the DMCTerminal control's output window using this method.

### Usage

`[form.][control.]DMCWriteToResponseBox( sString as String )`

---

## DMCTerminal2 Event Descriptions

### DMCError

#### Description

The DMCTerminal control uses this event to notify the application program of an error.

#### Usage

Sub *ctlname*\_DMCError(*ErrorNumber* As Long, *ErrorMessage* As String)

#### Remarks

The following table contains the list of possible errors. A complete, up-to-date list of error numbers can be found in the file DMCCOM40.BAS for Visual Basic programmers and the file DMCCOM.H for C/C++ programmers.

ErrorNumber	Description
-1 (DMCERROR_TIMEOUT)	A time-out occurred waiting for a response from the controller.
-2 (DMCERROR_COMMAND)	<a href="#">Galil language</a> command error.
-3 (DMCERROR_CONTROLLER)	Invalid controller number.
-4 (DMCERROR_FILE)	Error opening or saving a file.
-5 (DMCERROR_DRIVER)	Device driver failure.
-6 (DMCERROR_HANDLE)	Invalid controller handle.
-7 (DMCERROR_HMODULE)	Could not load required dynamic link library.
-8 (DMCERROR_MEMORY)	Out of memory.
-9 (DMCERROR_BUFFERFULL)	User provided data buffer was not large enough.
-10 (DMCERROR_RESPONSEDATA)	User provided data buffer is full and data truncation has occurred.
-11 (DMCERROR_DMA)	Could not communicate with DMA channel.
-12 (DMCERROR_ARGUMENT)	One or more required arguments to a DMC API function call was NULL.
-13 (DMCERROR_DATARECORD)	Could not access DMC data record.
-14 (DMCERROR_DOWNLOAD)	Download error - probable cause is a line too long or too many lines.
-15 (DMCERROR_FIRMWARE)	Could not update the controller's firmware.
-16 (DMCERROR_CONVERSION)	Could not convert DMC command (ASCII to binary or binary to ASCII).

-17 (DMCERROR_RESOURCE)	Windows reports a resource conflict with the current hardware configuration.
-18 (DMCERROR_REGISTRY)	Could not access or modify the controller's registry information.
-19 (DMCERROR_BUSY)	Controller is busy and not ready to accept commands.
-20 (DMCERROR_DEVICE_DISCONNECTED)	Controller has been disconnected from the communications channel (USB or Ethernet).
-21 (DMCERROR_TIMEING_ERROR)	Data is not being transfered to controller fast enough to maintain time synchronization.
-22 (DMCERROR_WRITEBUFFER_TOO_LARGE)	The user supplied buffer is too large. Must be < 1024 bytes.
-23 (DMCERROR_NO_MODIFY_PNP_CONTROLLER)	Registry modification of PnP controllers is not allowed.
-24 (DMCERROR_FUNCTION_OBSOLETE)	This function is obsolete.

### Further remarks:

Invalid controller handle errors most often occur because a connection to a Galil motion controller has not been established. You must use the DMCSHELL control to establish a communications session.

[Galil language](#) command errors occur for three reasons:

1. An unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands that are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

Time-outs can occur for two reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. The Galil motion controller is waiting for a trip-point, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trip-point is cleared. This is not an error.

---

## **DMCResponse Event**

### **Description**

The DMCTerminal control uses this event to notify an application program of an unsolicited response from the Galil motion controller. Unsolicited responses from the Galil motion controller can be periodically checked by using the DMCPollController property.

### **Usage**

Sub *ctlname*\_DMCResponse(*Response* As String)

#### Remarks

The *Response* argument of the event contains the unsolicited response from the Galil motion controller. Unsolicited responses can occur when the Galil motion controller is executing a Galil language program and the Galil language program executes a Galil language command that result in messages being placed on the controller's output FIFO (e.g., MG "message" or 'var1 =').

---

## **DMCTransaction**

#### Description

The DMCTerminal control uses this event to notify the application program that a transaction between the terminal or an application program and the Galil motion controller has occurred. A transaction is defined to mean that a Galil language command has been sent to the controller using the [DMCCommand](#) property. The command and the resulting response from the Galil motion controller is recorded and sent in the *Command* and *Response* arguments of this event. Since the DMCTerminal uses this property internally when the input line of the [command window](#) is used to issue a command, any transactions originating from the the terminal window are also sent to the DMCTransaction event.

#### Usage

Sub *ctlname*\_DMCTransaction(*Command* As String, *Response* As String)

#### Remarks

The DMCTransaction event may be used by the application program to log all commands and responses to/from the Galil motion controller.

---

## **DMCTerminal2 Example**

#### Description

This simple example demonstrates how to use the DMCTerminal2 control in Visual Basic 6, producing a fully functioning terminal. The following properties and methods are used:

- DMCEnableCapLock
- DMCAllowedToolbarAlignment
- DMCAllowedTerminalMode
- DMCController
- DMCPollController
- DMCPollInterval
- DMCEnableToolbar
- DMCEnableControlChars
- DMCEnableSyntaxHelp
- DMCSmartTermSplitterRowPos
- DMCEditTermSplitterRowPos
- DMCEditTermSplitterColPos
- DMCEndOfFile
- DMCSetToolbarPosition



'To reproduce this example:

- '1. In Visual Basic 6, open a standard project.
- '2. From the Project/Components menu selection, add a Galil DMCTerminal2, DMCSShell, and DMCRegister control to the toolbar.
- '3. Add these controls to your form.
- '4. Paste the following code into the form's code window and run the project.

#### Private Sub Form\_Load()

##### On Error GoTo Form\_LoadErrorHandler

```
'Ask the user to select a controller and then connect to it
Me.DMCSShell1.DMCController = Me.DMCRegister1.SelectController
Me.DMCSShell1.DMCConnect = True
```

```
'Point the terminal to Shell's controller
Me.DMCTerminal1.DMCController = Me.DMCSShell1.DMCController
```

```
'Configure the terminal with desired properties. A programmer can persist
these properties between sessions if desired. Also, many can be set at
design-time.
```

```
With Me.DMCTerminal1
```

```
.Appearance = Appearance3D
.DMCAllowedTerminalMode = AnyTerm      'Allow all terminal modes
.DMCAllowedToolbarAlignment = AllowDockAny 'Allow the toolbar to dock
to any side of frame or float

'Set the initial position of the horizontal and vertical splitter bars
for the
'edit and smart terminal modes.
.DMCEditTermSplitterRowPos = .Height / (3 * Screen.TwipsPerPixelY)
.DMCSmartTermSplitterRowPos = .Height / (3 * Screen.TwipsPerPixelY)
.DMCEditTermSplitterColPos = .Width / (3 * Screen.TwipsPerPixelX)

'Disable the Syntax help. If we haven't gone to the trouble of
installing MDAC2.5 and
'configuring the Syntax help database source, the feature will gracefully
disable itself.
.DMCEnableSyntaxHelp = True

'Don't have Control Characters or Cap Locks enabled initially.
.DMCEnableControlChars = False
.DMCEnableCapLock = False

'Enable the Toolbar and make sure it is visible
.DMCEnableToolbar = True
.DMCSetToolbarPosition DockRight, 0, 0

'Set the polling interval and start the polling. This allows the
terminal to receive unsolicited messages
.DMCPollInterval = 100 'in milliseconds
.DMCPollController = True
```

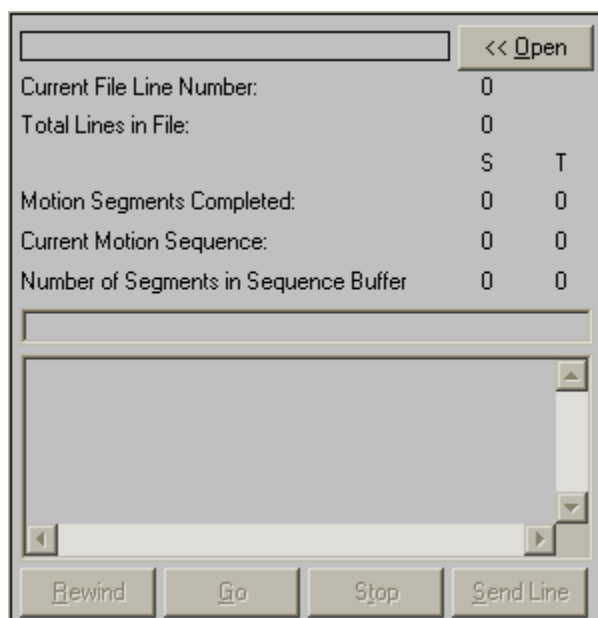
```
End With  
  
Exit Sub  
Form_LoadErrorHandler:  
  
    MsgBox ("Error: Failed to properly initialize the terminal.")  
  
End Sub
```

**THIS PAGE LEFT BLANK INTENTIONALLY**



## Chapter 6 – DMCSend Control

The send file to controller object can be used to send application programs to the controller in their entirety or one line at a time. Responses are displayed for each application program line sent. Use this object to send vector files or standard commands where the response of the card is of interest. In all other cases use the DMCFFileOperation property of the DMCSShell to send and download files. An important feature of this tool is the DMCSegments property, which allows a number of segments to be sent to the card before a BGS (begin sequence) is sent. It also provides recovery information and methods in the case of motion being stopped due to an abort or limit switch. The send control allows use of both the S and T coordinated planes.



### Galil Properties, Methods, and Events

#### Properties

DMCAbortCheckTimerInterval  
 DMCActionOnStop  
 DMCBeginMotionOnSend  
 DMCController  
 DMCCurrentLine  
 DMCCurrentMotionSSegment  
 DMCCurrentMotionSSequence  
 DMCCurrentMotionTSegment  
 DMCCurrentMotionTSequence  
 DMCCurrentSSequenceSegmentCount  
 DMCCurrentTSequenceSegmentCount  
 DMCFilename  
 DMCFFileOperation  
 DMCHideButtons

#### Methods

ClickGoButton  
 ClickOpenButton  
 ClickRewindButton  
 ClickSendButton  
 ClickStopButton  
 DMCWaitForMotionComplete  
 Open  
 Close

#### Events

DMCAbortInfo  
 DMCEndOfFile  
 DMCError  
 DMCResponse

DMCHideOpenButton  
DMCMaxSegmentsInBuffer  
DMCSegmentBackupOnAbort  
DMCSegment  
DMCStopOnError  
DMCTotalLines

## DMCSend Property Descriptions

### **DMCAbortCheckTimerInterval**

#### **Description**

This property sets the number of milliseconds that pass in between checks of the stopcodes for limit switch stops or abort stops.

#### **Usage**

[*form.*][*control.*]**DMCAbortCheckTimerInterval** [= *integer*]

#### **Setting**

The setting for the DMCAbortCheckTimerInterval property must be an integer between 2 and 1000. The default setting is 16.

#### **Note**

This property may be used at any time. This property is saved if set at design time.

#### **Data Type**

Integer

---

### **DMCActionOnStop**

#### **Description**

The action to take when the Stop button on the [DMCSend control](#) is clicked or pressed.

#### **Usage**

[*form.*][*control.*]**DMCActionOnStop**[= *action setting*]

#### **Setting**

The DMCActionOnStop property settings are:

<b>ActionOnStopStopSendingCommands</b>	Stop Sending Commands. (default setting)
<b>ActionOnStopPauseCoordinatedMotion</b>	Pause Coordinated Motion.
<b>ActionOnStopAbortMotion</b>	Abort motion
<b>ActionOnStopFireAbortInfoEvent</b>	Cause the AbortInfo Event to occur on
action	

#### **Remarks**

The DMCActionOnStop property gives the developer some flexibility on how to use the Stop button. The "Stop Sending Commands" option will simply not send any more commands to the Galil motion controller. Any commands already sent to the controller will still be executed. The "Pause Coordinated Motion" option

will send a VR0 command to the Galil motion controller, effectively pausing a coordinated motion sequence. This option will only work when the controller is in coordinated motion (VM) or linear interpolation (LM) mode. The "Abort Motion" option will send an AB1 command to the Galil motion controller stopping any motion immediately without deceleration.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCBeginMotionOnSend**

### Description

Used to begin motion immediately on linear interpolation (LI) or vector position (VP) motion segments when the Send command is issued (Send button is pushed).

### Usage

[form.][control.]DMCBeginMotionOnSend[= {True | False}]

### Setting

The DMCBeginMotionOnSend property settings are:

<b>False</b>	Do not begin motion immediately when the Send command is issued. This is the Default setting.
<b>True</b>	Begin motion immediately when the Send command is issued.

### Remarks

For Galil language application program files that use linear interpolation mode (LM) or vector mode (VM), a begin motion sequence command (BGS or BGT) will be sent for each linear interpolation (LI) or vector position (VP) motion segment sent to the Galil controller, if the DMCBeginMotionOnSend property is set to True. This may be useful for profiling motion in single steps.

Errors are reported via the [DMCError](#) event.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Integer (Boolean)

---

## **DMCController**

### Description

The number corresponding to the Galil motion controller registered in the Windows registry database.

### Usage

[form.][control.]DMCController[= controller]

### Setting

The setting for the DMCController property must be an integer between 1 and 16. The default setting is 1.

### Remarks

A Galil motion controller can be registered in the Windows registry database by using the DMCREG program for Windows 3.x or the DMCREG32 program for Windows 95, 98, NT, or 2000.

### Note

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

### Data Type

Long

---

## **DMCCurrentLine**

### Description

The current line in the Galil language application program file currently open. That is, the next line to be sent to the Galil motion controller.

### Usage

*current line* = [form.][control.]DMCCurrentLine]

### Setting

There is no setting. This property is read-only.

### Remarks

When the last line in the Galil language application program file has been sent, the DMCCurrentLine property is set to the same value as the [DMCTotalLines](#) property, that is, it is set to the total number of lines in the file.

### Note

This property is available at run-time only.

### Data Type

Long

---

## **DMCCurrentMotionSSegment**

### Description

The last linear interpolation mode (LM) or vector mode (VM) motion segment to be executed by the Galil motion controller. Returns the currently executing **S** plane segment. This property should be used in place of the Version 1 property DMCCurrentMotionSegment that is now hidden and has the same functionality as DMCCurrentMotionSSegment.

### Usage

*current motion segment* = [form.][control.]DMCCurrentMotionSSegment

### Setting

There is no setting. This property is read-only.

## Remarks

The DMCCurrentMotionSSegment property is not set to 0 after an LM or VM command is sent to the Galil motion controller, even though these commands denote the start of a new coordinated motion sequence. Therefore, motion segments are tracked globally for an entire file. To get the current sequence, use the DMCCurrentMotionSSequence property. To get the current segment count from the beginning of the current sequence, use the DMCCurrentSSequenceSegmentCount property. Motion segments defined in either LM or VM mode are not executed until a begin coordinated motion sequence command (BGS) has been sent to the Galil motion controller. For modes of motion other than LM or VM, this property has no relevance. For more information on the LM and VM modes of motion, see the Technical Reference Guide for the Galil motion controller installed.

## Note

This property is available at run-time only.

## Data Type

Long

---

## **DMCCurrentMotionTSegment**

## Description

The last linear interpolation mode (LM) or vector mode (VM) motion segment to be executed by the Galil motion controller. Returns the currently executing **T** plane segment. This property should be used in place of the Version 1 property DMCCurrentMotionSegment that is now hidden and has the same functionality as DMCCurrentMotionTSegment.

## Usage

*current motion segment* = [*form.*][*control.*]**DMCCurrentMotionTSegment**

## Setting

There is no setting. This property is read-only.

## Remarks

The DMCCurrentMotionTSegment property is not set to 0 after an LM or VM command is sent to the Galil motion controller, even though these commands denote the start of a new coordinated motion sequence. Therefore, motion segments are tracked globally for an entire file. To get the current sequence, use the DMCCurrentMotionTSequence property. To get the current segment count from the beginning of the current sequence, use the DMCCurrentTSequenceSegmentCount property. Motion segments defined in either LM or VM mode are not executed until a begin coordinated motion sequence command (BGT) has been sent to the Galil motion controller. For modes of motion other than LM or VM, this property has no relevance. For more information on the LM and VM modes of motion, see the Technical Reference Guide for the Galil motion controller installed.

## Note

This property is available at run-time only.

## Data Type

Long



---

## **DMCCurrentMotionSSequence Property**

### **Description**

The DMCCurrentMotionSSequence returns the current S plane sequence that is current being executed by the controller. If a send file has multiple VM or LM commands, then it has multiple sequences.

### **Usage**

$nLastSSequence = [form.][control.]DMCCurrentMotionSSequence$

### **Setting**

There is no setting. This property is read-only.

### **Remarks**

The DMCCurrentMotionSSequence property is set to 0 at the beginning of a send file's execution. Every time an LM or VM command is encountered in the file and the S plane is currently selected (Galil application program command 'CAS'), this property is incremented. To get the current segment, use the DMCCurrentSSequenceSegmentCount property. To get the current segment count from the beginning of the file, use the DMCCurrentMotionSSegment.

### **Note**

This property is available at run-time only.

---

## **DMCCurrentMotionTSequence Property**

### **Description**

The DMCCurrentMotionTSequence returns the current T plane sequence that is current being executed by the controller. If a send file has multiple VM or LM commands, then it has multiple sequences.

### **Usage**

$nLastTSequence = [form.][control.]DMCCurrentMotionTSequence$

### **Setting**

There is no setting. This property is read-only.

### **Remarks**

The DMCCurrentMotionSSequence property is set to 0 at the beginning of a send file's execution. Every time an LM or VM command is encountered in the file and the T plane is currently selected (Galil application program command 'CAT'), this property is incremented. To get the current segment, use the DMCCurrentTSequenceSegmentCount property. To get the current segment count from the beginning of the file, use the DMCCurrentMotionTSegment.

### **Note**

This property is available at run-time only.

---

## **DMCCurrentSSequenceSegmentCount Property**

### **Description**

The DMCCurrentSSequenceSegmentCount returns the last S Plane linear interpolation mode (LM) or vector mode (VM) motion segment to be executed by the Galil motion controller within the current sequence.

### **Usage**

$nLastSSegment = [form.][control.]DMCCurrentSSequenceSegmentCount$

### **Setting**

There is no setting. This property is read-only.

### **Remarks**

The DMCCurrentSSequenceSegmentCount property is set to 0 after an LM or VM command is sent to the Galil motion controller. The LM or VM command defines the beginning of a new sequence. To get the current sequence, use the [DMCCurrentMotionSSequence](#) property. To get the current segment count from the beginning of the file, use the [DMCCurrentMotionSSegment](#) property. Motion segments defined in either LM or VM mode are not executed until a begin coordinated motion sequence command (BGS) has been sent to the Galil motion controller. For modes of motion other than LM or VM, this property has no relevance. For more information on the LM and VM modes of motion, see the Technical Reference Guide for the Galil motion controller installed.

### **Note**

This property is available at run-time only.

---

## **DMCCurrentTSequenceSegmentCount Property**

### **Description**

The DMCCurrentTSequenceSegmentCount returns the last T Plane linear interpolation mode (LM) or vector mode (VM) motion segment to be executed by the Galil motion controller within the current sequence.

### **Usage**

$nLastTSegment = [form.][control.]DMCCurrentTSequenceSegmentCount$

### **Setting**

There is no setting. This property is read-only.

### **Remarks**

The DMCCurrentTSequenceSegmentCount property is set to 0 after an LM or VM command is sent to the Galil motion controller. The LM or VM command defines the beginning of a new sequence. To get the current sequence, use the DMCCurrentMotionTSequence property. To get the current segment count from the beginning of the file, use the DMCCurrentMotionTSegment property. Motion segments defined in either LM or VM mode are not executed until a begin coordinated motion sequence command (BGT) has been sent to the Galil motion controller. For modes of motion other than LM or VM, this property has no relevance. For more information on the LM and VM modes of motion, see the Technical Reference Guide for the Galil motion controller installed.

## Note

This property is available at run-time only.

---

## **DMCFileName**

### **Description**

Used to set the Galil language application program file name.

### **Usage**

*[form.][control.]DMCFileName[= file name]*

### **Setting**

The setting for the DMCFileName property may be any valid DOS file name, including the path. Note that a file name is a string.

### **Remarks**

The DMCFileName property must be set before the DMCSend control buttons [Go](#), [Rewind](#), [Send](#), or [Stop](#) can be used. Likewise, the [DMCFileOperation](#) property and the [Open](#) method can not be used until the DMCFileName property has been set.

## Note

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

String

---

## **DMCFileOperation**

### **Description**

This property is left in the control for binary compatibility with previous version of the control. The DMCOpenFile and ClickGoButton methods should be used instead.

### **Usage**

*[form.][control.]DMCFileOperation[= file operation]*

### **Setting**

The DMCFileOperation property settings are:

#### **SettingDescription**

0. No file operation.
1. Open the file.
2. Send the file to the Galil motion controller.

## Remarks

The DMCFilename property must be set before the open or send file operations can commence. The file must consist of valid Galil language commands.

Sending a file transmits each line in the file to the Galil motion controller as a separate Galil language command (or group of commands if there is more than one command on a line). Each Galil language command is interpreted by the Galil motion controller immediately. If the DMCSend control encounters an AM command (After Motion trip-point), the DMCSend control does not actually send the command to the Galil motion controller. It simulates the AM command internally. In that way communication to the Galil motion controller is not blocked (all trip-points when sent from the host PC block communications until they are satisfied).

Once the open or send file operation has completed, the DMCFileOperation property is set back to 0 by the DMCSend control.

## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established using the DMCSHLL control.

---

## **DMCHideButtons Property**

### Description

Allows the programmer to hide any or all of the DMCSend control's command buttons.

### Setting Description

The DMCHideButtons property can be set to any of the DMCSendButtonConstant values. These values can be or'd together to hide combinations of buttons. The default setting is ShowAllButtons.

The DMCSendButtonConstants are:

**HideRewindButton (0x01)**  
**HideGoButton (0x02)**  
**HideStopButton (0x04)**  
**HideSendButton (0x08)**  
**HideOpenButton (0x10)**  
**HideAllButtons (0x1F)**  
**ShowAllButtons (0x00)**

### Remarks

Even some or all of the buttons are hidden, a programmer can access a button's command by using the equivalent method (such as [ClickGoButton\(\)](#)).

### Note

This property may be used at any time. This property is saved if set at design time.

---

## **DMCHideOpenButton**

### Description

Show or hide the Open button.

## Usage

[form.][control.]DMCHideOpenButton[= {True | False}]

## Setting

The DMCHideOpenButton property settings are:

<b>False</b>	Do not hide the Open button. This is the Default setting.
<b>True</b>	Hide the Open button.

## Remarks

Even if the Open button is hidden, files may be opened via the [DMCFileName](#) property. This allows a custom file dialog to be used by the developer, and/or prevents the user from opening another file.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

## **DMCMaxSegmentsInBuffer Property**

## Description

The DMCMaxSegmentsInBuffer property sets the maximum number of segments that are allowed to accumulate in the motion controller's buffer at a time. The default value is -1, which means there is no limit other than the buffer's actual limit (256 or 511 segments). Setting the property to 10 would allow up to ten segments into the buffer at a time.

## Usage

[form.][control.]DMCMaxSegmentsInBuffer[= nMaxSegsInBuf]  
nMaxSegsInBuf = [form.][control.]DMCMaxSegmentsInBuffer

## Setting

The DMCMaxSegmentsInBuffer property setting must be an integer between -1 and 100. The default setting is -1.

## Remarks

This property can be useful when an application needs to dynamically create or alter a motion profile. It allows the programmer to limit the number of segments in the motion controller's sequence buffer, therefore reducing the programming that otherwise would be required to dynamically insert new motion segments into the sequence buffer.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCSegmentBackupOnAbort Property**

### **Description**

If an abort recovery occurs (the [DMCActionOnStop](#) property enables recovery and the [DMCAbortInfo](#) Event transfers the recovery information to the host application), this property will cause the recovery file to back up from the segment in which the abort/stop occurred by the number of segments specified by this property.

### **Usage**

*[form.][control.]DMCSegmentBackupOnAbort[= nNumBackupOnAbort]*  
*nNumBackupOnAbort = [form.][control.]DMCSegmentBackupOnAbort*

### **Setting**

The DMCSegmentBackupOnAbort property setting must be an integer between 0 and 100. The default setting is 0.

### **Remarks**

If an abort recovery occurs, and the value of DMCSegmentBackupOnAbort is greater than the number of segments processed by the controller, the recovery file will start from the beginning of the sequence.

### **Note**

This property may be used at any time. The property is saved if set at design time.

## **DMCSegments**

### **Description**

The number of linear interpolation mode (LM) or vector mode (VM) motion segments to send before automatically sending a begin coordinated motion sequence command (BGS).

### **Usage**

*[form.][control.]DMCSegments[= number of segments]*

### **Setting**

The DMCSegments property setting must be an integer between 0 and 511. The default setting is 5.

### **Remarks**

If the DMCSegments property is set to 0, the BGS or BGT command must be issued from the application program using the [DMCSend\\_control](#). For modes of motion other than LM or VM, this property has no relevance. For more information on the LM and VM modes of motion, see the Technical Reference Guide for the Galil motion controller installed.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCStopOnError**

### **Description**

Used to stop a send file operation if an error is detected.

### **Usage**

*[form.]**[control.]***DMCStopOnError**[= {True | False}]

### **Setting**

The DMCStopOnError property settings are:

<b>False</b>	Do not stop on errors. This is the Default setting.
<b>True</b>	Stop on errors.

### **Remarks**

Errors are reported via the [DMCError](#) event.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Integer (Boolean)

---

## **DMCTotalLines**

### **Description**

The total number of lines in the Galil language application program language file currently open.

### **Usage**

*total lines* = *[form.]**[control.]***DMCTotalLines**

### **Setting**

There is no setting. This property is read-only.

### **Remarks**

When a Galil language application program file is opened, the DMCTotalLines property is set to the total number of lines in the file. A line is a block of text separated by a carriage return and line feed (CR/LF).

### **Note**

This property is available at run-time only.

### **Data Type**

Long

---

## DMCSend Event Descriptions

### Galil Events

DMCAbortInfo  
DMCEndOfFile  
DMCError  
DMCResponse

### DMCAbortInfo Event

#### Description

The DMCAbortInfo event is used by the DMCSend control to notify the application program that an abort has occurred and a recovery file created. This mode is enabled using the [DMCActionOnStop](#) property. The following information is passed by the event's parameters:

**bUsingSPlane** - if true, motion was on the S plane at the time of the abort. If false, motion was on the T plane.

**nLastSequence** - the S or T plane sequence in which the abort occurred. Sequences are counted independently for the S and T planes.

**nLastSegment** - the segment of the nLastSequence on which the stop or abort occurred.

**sRecoveryFile** - a string containing the path to the recovery file. The file is always named AbortRecovery.sen and is placed in the same directory as the sen file that was being executed when the abort occurred. The AbortRecovery.sen file contains the instruction that will correctly complete the original motion once the problem has been fixed and the vector motion plane has been returned to the coordinates passed in the nAxis1AbsCoord and nAxis2AbsCoord parameters.

**sPString** - a string containing a PA command that can be used to return the motors to the correct absolute coordinates prior to sending the AbortRecovery.sen file.

#### Syntax

Sub *ctlname* **DMCAbortInfo**(bUsingSPlane As Boolean, nLastSequence As Long, nLastSegment As Long, sRecoveryFile As String, sPString As String)

#### Remarks

This event, when enabled by the DMCActionOnStop property, allows the programmer to more easily create a program that can recover from an abort. For example, if a tool breaks and an operator presses a button that calls the [DMCAbortSend](#) method, the DMCAbortInfo event will fire, passing the information described above. The programmer can then move the tool to a specified location for replacement, return the tool to the location specified by the sPString and then send the sRecoveryFile to complete the tool's originally programmed path. If it is desired that a number of motion segments prior to the tool breakage be repeated, the [DMCSegmentBackupOnAbort](#) property can be used to specify the number of motion segments that the recovery file should repeat.



## **DMCEndOfFile**

### **Description**

Used by the DMCSend control to notify the application program that the last line of the Galil language application program file was sent to the Galil motion controller.

### **Syntax**

Sub *ctlname\_DMCEndOfFile*()

### **Remarks**

The DMCEndOfFile event may be fired before motion is complete because one or more motion segments may still be executing on the Galil motion controller.

---

## **DMCError**

### **Description**

Used by the DMCSend control to notify the application program of an error.

### **Syntax**

Sub *ctlname\_DMCError*(*ErrorNumber*As Long, *ErrorMessage*As String)

### **Remarks**

The two most frequent errors are:

<b>ErrorNumber</b>	<b>Description</b>
-1	A time-out occurred.
-2	Galil language command error.

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCSend control may detect a time-out

condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.

3. The Galil motion controller is waiting for a trippoint, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trippoint is cleared. This is not an error.

---

## **DMCResponse**

### **Description**

Used by the DMCSend control to notify the application program of a response from the Galil motion controller.

### **Syntax**

Sub *ctlname*\_DMCResponse(*ResponseAs* String)

### **Remarks**

Response contains the response from the Galil motion controller for each Galil language application program line sent.

---

## **DMCSend Method Descriptions**

### **Galil Methods**

Close  
Open  
ClickGoButton  
ClickOpenButton  
ClickRewindButton  
ClickSendButton  
ClickStopButton  
DMCAbortSend  
DMCWaitForMotionComplete

### **Close**

### **Description**

Used to close the current Galil language application program file.

### **Usage**

[*form.*][*control.*]Close

### **Remarks**

The Close method is equivalent to setting the [DMCFileName](#) property to NULL, and then setting the [DMCFileOperation](#) property to 1.

### **Note**

This method may be only be used at run time.

---

## **Open**

### **Description**

Used to open the Galil language application program file.

### **Usage**

RC = [*form.*][*control.*]**Open**

RC is a variable of data type long.

### **Remarks**

The [DMCFileName](#) property must be set before using the Open method. The Open method is equivalent to setting the [DMCFileOperation](#) property to 1.

### **Note**

This method may be only be used at run time.

---

## **ClickGoButton**

### **Description**

Used to simulate clicking the Go button or pressing the Alt+G key combination.

### **Usage**

[*form.*][*control.*]**ClickGoButton**

### **Remarks**

OLE control containers such as Microsoft Visual C++ which do not send the return, esc, or mnemonic keys to OLE custom controls may need to use this method to simulate clicking the Go button. You may find it helpful to create a button on your form which uses the Alt+G mnemonic. When this button is clicked or the mnemonic entered, your button control handler would invoke this method. Your button could even be created invisible so as not to clutter up your form.

Please note that Visual Basic does not have this problem.

### **Note**

This method may be only be used at run time.

---

## **ClickOpenButton**

### **Description**

Used to simulate clicking the Open button or pressing the Alt+O key combination.

### **Usage**

[*form.*][*control.*]**ClickOpenButton**

### **Remarks**

OLE control containers such as Microsoft Visual C++ which do not send the return, esc, or mnemonic keys to OLE custom controls may need to use this method to simulate clicking the Open button. You may find it

helpful to create a button on your form which uses the Alt+O mnemonic. When this button is clicked or the mnemonic entered, your button control handler would invoke this method. Your button could even be created invisible so as not to clutter up your form.

Please note that Visual Basic does not have this problem.

### **Note**

This method may be only be used at run time.

---

## **ClickRewindButton**

### **Description**

Used to simulate clicking the Rewind button or pressing the Alt+R key combination.

### **Usage**

*[form.][control.]ClickRewindButton*

### **Remarks**

OLE control containers such as Microsoft Visual C++ which do not send the return, esc, or mnemonic keys to OLE custom controls may need to use this method to simulate clicking the Rewind button. You may find it helpful to create a button on your form which uses the Alt+R mnemonic. When this button is clicked or the mnemonic entered, your button control handler would invoke this method. Your button could even be created invisible so as not to clutter up your form.

Please note that Visual Basic does not have this problem.

### **Note**

This method may be only be used at run time.

---

## **ClickSendButton**

### **Description**

Used to simulate clicking the Send button or pressing the Alt+S key combination.

### **Usage**

*[form.][control.]ClickSendButton*

### **Remarks**

OLE control containers such as Microsoft Visual C++ which do not send the return, esc, or mnemonic keys to OLE custom controls may need to use this method to simulate clicking the Send button. You may find it helpful to create a button on your form which uses the Alt+S mnemonic. When this button is clicked or the mnemonic entered, your button control handler would invoke this method. Your button could even be created invisible so as not to clutter up your form.

Please note that Visual Basic does not have this problem.

### **Note**

This method may be only be used at run time.

---

## **ClickStopButton**

### **Description**

Used to simulate clicking the Stop button or pressing the Alt+T key combination.

### **Usage**

`[form.][control.]ClickGoButton`

### **Remarks**

OLE control containers such as Microsoft Visual C++ which do not send the return, esc, or mnemonic keys to OLE custom controls may need to use this method to simulate clicking the Stop button. You may find it helpful to create a button on your form which uses the Alt+T mnemonic. When this button is clicked or the mnemonic entered, your button control handler would invoke this method. Your button could even be created invisible so as not to clutter up your form.

Please note that Visual Basic does not have this problem.

### **Note**

This method may be only be used at run time.

DMCShell Visual Basic control supplies the base support functions necessary for communicating with the Galil motion controller.

## **DMCAbortSend Method**

### **Description**

The DMCAbortSend method causes DMCSend control to abort the current send file and fire the DMCAbortInfo event.

### **Usage**

`[form.][control.]DMCAbortSend( )`

### **Remarks**

If the abort recovery mode has been enabled (see [DMCActionOnStop](#)), this method has the same effect as if a stop code had occurred. This method allows the programmer to create an abort button on a form so that an operator can perform an abort recovery in the event, for example, of a tooling failure.

### **Note**

This method may only be used at run time.

## **DMCWaitForMotionComplete Method**

### **Description**

The DMCWaitForMotionComplete method checks for motion on all axis and returns when motion on all axis is complete.

### **Usage**

`[form.][control.]DMCWaitForMotionComplete( )`

## DMCSend Button Descriptions

### Go

#### Description

Send the entire Galil language application file (from the current line) to the Galil motion controller.

---

### Open

#### Description

Locate and open a Galil language application file.

---

### Rewind

#### Description

Rewind the Galil language application file to the beginning (the first line).

---

### Send

#### Description

Send the current line from the Galil language application file to the Galil motion controller. If the [DMCBeginMotionOnSend](#) property is set to True, linear interpolation (LI) or vector position (VP) motion segments will execute immediately.

---

### Stop

#### Description

Stop sending the Galil language application file to the Galil motion controller. In addition, a stop command ("ST") is issued as well as a clear sequence command ("CS"). This has the effect of halting motion on all axis as well as clearing any pending linear interpolation (LI) or vector position (VP) motion segments.

## DMCSend Examples

### Responding to the DMCError event

In this example, the application program reports on errors as sent by the Galil motion controller.

```
Sub DMCMove1_DMCError (ErrorNumber As Long, ErrorMessage As String)
    Dim RC As Integer
    RC = MsgBox(ErrorMessage, MB_ICONEXCLAMATION, "DMC Error")
End Sub
```

### Responding to the Response Event

In this example, the application program is placing the responses from the Galil motion controller in a text control named ResponseOutput.

```
Sub DMCSend1_DMCRresponse (Response As String)
    ResponseOutput.Text = Response
End Sub
```

## Simple Interface Example

### Description

This simple example demonstrates how to use the DMCSend control in Visual Basic 6, producing a very simple interface that is presented to a user. The following properties and methods are used:

```
DMCBeginMotionOnSend
DMCMaxSegmentsInBuffer
DMCActionOnStop
DMCController
DMCFileName
DMCWaitForMotionComplete
ClickGoButton
ClickRewindButton
ClickStopButton
Close
Open
DMCEndOfFile
```

'To reproduce this example: ' '1. In Visual Basic 6, open a standard project.  
'2. From the Project/Components menu selection, add a Galil DMCTerminal2 and DMCSHLL to the toolbar.  
'3. Add these controls to your form.  
'4. Add a command button named 'cmdStart' with the caption 'Start'.  
'5. Add a second command button named 'cmdStop' with the caption 'Stop'.  
'6. Paste the following code into the form's code window (do not include the send file below).  
'7. Paste the text from below marked as "SimpleExample.sen" into a text file and save the  
' file as "SimpleExample.sen" in a known directory location.

```
Private Sub cmdStart_Click()

'open send file and start
Me.DMCSend1.DMCFileName = "\\YourPathHere\\SimpleExample.sen"
Me.DMCSend1.Open
Me.DMCSend1.ClickGoButton

cmdStart.Enabled = False
cmdStop.Enabled = True

End Sub

Private Sub cmdStop_Click()

Me.DMCSend1.ClickStopButton
Me.DMCSend1.ClickRewindButton

cmdStart.Enabled = True
cmdStop.Enabled = False

End Sub

Private Sub DMCSend1_DMCEndOfFile()
Me.DMCSend1.DMCWaitForMotionComplete
Me.DMCSend1.ClickRewindButton
```

```

cmdStart.Enabled = True
cmdStop.Enabled = False

End Sub

Private Sub Form_Load()

' Make DMCSend Control invisible
Me.DMCSend1.Visible = False

' Connect to a motion controller
Me.DMCShell1.DMCController = 3
Me.DMCShell1.DMCConnect = True
Me.DMCSend1.DMCController = Me.DMCShell1.DMCController

' Set stop and begin motion property
Me.DMCSend1.DMCActionOnStop = ActionOnStopStopSendingCommands
Me.DMCSend1.DMCBeginMotionOnSend = True
Me.DMCSend1.DMCMaxSegmentsInBuffer = 30

cmdStart.Enabled = True
cmdStop.Enabled = False

End Sub

Private Sub Form_Unload(Cancel As Integer)

Me.DMCSend1.Close

End Sub

```

\*\*\*\*\*

Copy and paste the text that follows into a file named SimpleExample.sen

```

#SIMPEX
CSS
CAS
VS5000,5000
VMXY
VP 5000,0
VP 5000,5000
VP 10000,5000
VP 10000,10000
VP 15000,10000
VP 15000,15000
CR 5000,180,-180
VP 25000,5000
VP 20000,5000
VP 20000,0
VE

```

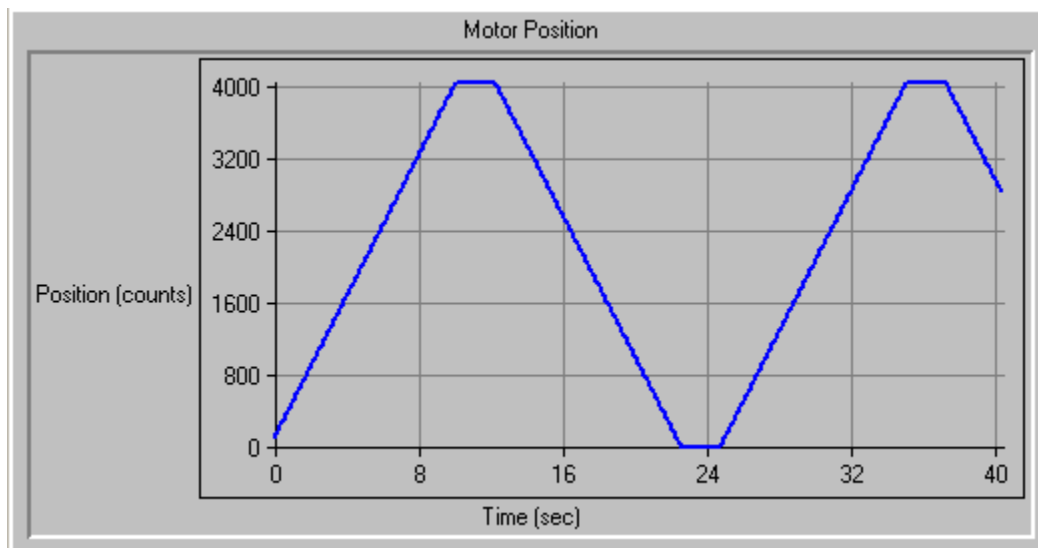


**THIS PAGE LEFT BLANK INTENTIONALLY**



## Chapter 7 DMCScope Control

The storage scope object allows you record and graph a variety of data from the controller. It can be used to plot trajectories such as position versus time or X versus Y. A very accurate snapshot of data can be taken then displayed or a continuous stream can scroll across the scope. Any data type supported by 'automatic data capture' can be displayed.



### Galil Properties and Events

#### Properties

DMCAxisColor  
DMCBevelInner  
DMCBevelOuter  
DMCBevelWidth  
DMCBorderWidth  
DMCCaptionPosition  
DMCCaptureData  
DMCContinuous  
DMCController  
DMCDataColor  
DMCDataType  
DMCDerivative  
DMCGridColor  
DMCGridOn  
DMCHorizontalSource  
DMCHorizontalTitle  
DMCMaxYValue  
DMCMinYValue  
DMCPointsPerAxis  
DMCSampleTime  
DMCStopOnTrigger

#### Events

DMCError  
DMCTrigger

DMCTriggerMaxValue  
DMCTriggerMinValue  
DMCTriggerOn  
DMCValuesOn  
DMCVerticalSource  
DMCVeritcalTitle

## DMCScope Property Descriptions

### **DMCAxisColor**

#### **Description**

The color for the axis displayed on the graph.

#### **Usage**

`[form.][control.]DMCAxisColor[= color]`

#### **Setting**

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

#### **Remarks**

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

#### **Note**

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed, one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

#### **Data Type**

Long

---

## **DMCBevelInner**

### **Description**

Used with or without the DMCBevelOuter property to give the DMCScope control a 3D appearance.

### **Usage**

`[form.][control.]DMCBevelInner[= setting]`

### **Setting**

The DMCBevelInner property settings are:

Setting Description	
0	None.
1	Inset. This is the default setting.
2	Raised.

### **Remarks**

The DMCBevelInner property is closely related to the [DMCBorderWidth](#) property. That is, the [DMCBorderWidth](#) property determines the width of the inner bevel.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

## **DMCBevelOuter**

### **Description**

Used with or without the DMCBevelInner property to give the DMCScope control a 3D appearance.

### **Usage**

`[form.][control.]DMCBevelOuter[= setting]`

### **Setting**

The DMCBevelOuter property settings are:

Setting Description	
0	None.
1	Inset.
2	Raised. This is the default setting.

### **Remarks**

The DMCBevelOuter property is closely related to the [DMCBevelWidth](#) property. That is, the [DMCBevelWidth](#) property determines the width of the outer bevel.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCBevelWidth**

### **Description**

Used to set the width of the bevel used by the DMCBevelInner property to give the DMCScope control a 3D appearance.

### **Usage**

`[form.][control.]DMCBevelWidth[= width]`

### **Setting**

The DMCBevelWidth property setting must be an integer between 1 and 30 inclusive.

### **Remarks**

The DMCBevelWidth property is closely related to the [DMCBevelOuter](#) property. That is, the DMCBevelWidth property determines the width of the outer bevel. The [DMCBevelOuter](#) property must be set to non-zero in order for the DMCBevelWidth property to have any effect.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCBorderWidth**

### **Description**

Used to set the width of the bevel used by the DMCBevelInner property to give the DMCScope control a 3D appearance.

### **Usage**

`[form.][control.]DMCBorderWidth[= width]`

### **Setting**

The DMCBorderWidth property setting must be an integer between 1 and 30 inclusive.

### **Remarks**

The DMCBorderWidth property is closely related to the [DMCBevelInner](#) property. That is, the DMCBorderWidth property determines the width of the inner bevel. The [DMCBevelInner](#) property must be set to non-zero in order for the DMCBevelWidth property to have any effect.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCCaptionPosition**

### **Description**

Used to position the display of the caption in the DMCScope control.

### **Usage**

`[form.][control.]DMCCaptionPosition[= position]`

### **Setting**

The DMCCaptionPosition property settings are:

<b>Setting Description</b>	
0	None.
1	Top. This is the default setting.
2	Left.
3	Bottom.
4	Right.

### **Remarks**

If the Caption property has not been set, the DMCCaptionPosition property will have no effect. To disable the display of the caption, set the DMCCaptionPosition property to 0 (none).

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCCaptureData**

### **Description**

This starts the DMCScope control recording data within the Galil motion controller. When the recording is done, the DMCCaptureData property is reset to False. If there is an error during the operation, the DMSError event is triggered.

### **Usage**

`[form.][control.]DMCCaptureData[= {True | False}]`

### **Setting**

The DMCCaptureData property settings are:

<b>Setting Description</b>	
True	Capture data.
False	Do not capture data. This is the default setting.

### **Remarks**

DMCCaptureData will fail under the following conditions:

- Unable to allocate memory within the Galil motion controller.
- Unable to communicate with the motion controller.

Conflict with another DMCScope control - an application can only have one DMCScope control per form.

## Note

You must set the Horizontal and Vertical sources (Position, Position Error, etc. - see the [DMCHorizontalSource](#) and [DMCVerticalSource](#) properties), as well as the Horizontal and Vertical Axis (X, Y, Time, etc. - see the [DMCDataType](#) property) before setting the [DMCCaptureData](#) property to True.

## Data Type

Integer (Boolean)

---

## **DMCContinuous**

## Description

When set to True, provides a quasi real-time storage scope; data is recorded and displayed continuously at the rate specified by the [DMCSampleTime](#) property.

## Usage

[form.][control.]**DMCContinuous**[= {True | False}]

## Setting

The DMCContinuous property settings are:

Setting	Description
---------	-------------

True	Record and display data continuously.
------	---------------------------------------

False	Record only the specified number of data points and display the data. This is the default setting.
-------	--

## Remarks

When the DMCContinuous property is set to False, the DMCScope control records the number of data points specified by the [DMCPointsPerAxis](#) property and displays the data. No further processing takes place.

When the DMCContinuous property is set to True, the DMCScope control records and displays data continuously. The [DMCHorizontalSource](#) property must be set to 0 or "Time". The [DMCMaxYValue](#) or [DMCMinYValue](#) properties must be set to define the range of values to be displayed by the DMCScope control. These properties may be set at run-time so that the range of values to be displayed can be dynamically expanded or contracted. The DMCScope control stop processing only when the DMCContinuous property is set to False.

When in "continuous "mode, the DMCScope control allows you to define a trigger which will notify your application through an event when the trigger has been set-off. For more information on setting a trigger, see the [DMCStopOnTrigger](#), [DMCTriggerMaxValue](#), [DMCTriggerMinValue](#), [DMCTriggerOn](#), properties and the [DMCTrigger](#) event.

## Note

You may not set this property when the DMCScope control is capturing data (the [DMCCaptureData](#) property is set to True). This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

## **DMCController**

### **Description**

The number corresponding to the Galil motion controller registered in the Windows registry database.

### **Usage**

`[form.][control.]DMCController[= controller]`

### **Setting**

The setting for the DMCController property must be an integer between 1 and 16. The default setting is 1.

### **Remarks**

A Galil motion controller can be registered in the Windows registry database by using the DMCREG program for Windows 3.x or the DMCREG32 program for Windows 95, 98, NT, or 2000.

### **Note**

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCDataColor**

### **Description**

The color for the data displayed on the graph.

### **Usage**

`[form.][control.]DMCDataColor[= color]`

### **Setting**

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

### **Remarks**

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

### **Note**

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green,



and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed, one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

## Data Type

Long

---

## **DMCDataType**

### Description

This is the type of data that will be collected and displayed on the graph.

### Usage

[*form.*][*control.*]**DMCDataType**[= *setting*]

### Setting

The DMCDataType property settings are:

Setting	Description
0	Position. This is the default setting.
1	PositionError.
2	CommandedPosition.
3	LatchedPosition.
4	DualEncoderPosition.
5	Inputs.
6	Outputs.
7	Switches.
8	StopCode.
.	Torque.
10-17	Analog Inputs 1-8

### Remarks

This property may be used at any time. This property is saved if set at design time.

### Note

Analog input recording is limited by the number of axes on a controller. For example, a DMC-1750 is able to record a maximum of 5 inputs with this tool.

You need to change this property if you do not want the default of Position.

## Data Type

Long

---

## **DMCDerivative**

### Description

When the data is recorded, the derivative of the data is performed before the data is displayed on the graph.

## Usage

[form.][control.]DMCDerivative[= {True | False}]

## Setting

The DMCDerivative property settings are:

	Setting Description
True	Produces a derivative of the recorded data.
False	Does no processing of the data. This is the default setting.

## Remarks

This property may be used at any time. This property is saved if set at design time.

## Note

The derivative is  $(\text{Value}[n] - \text{Value}[n-1]) / \text{sample\_time}$ .

## Data Type

Integer (Boolean)

---

## DMCGridColor

## Description

The color for the grid displayed on the graph.

## Usage

[form.][control.]DMCGridColor[= color]

## Setting

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

## Remarks

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

## Note

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed,

one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

## Data Type

Long

---

## **DMCGridOn**

### Description

This enables or disables the display of the grid.

### Usage

*[form.]**[control.]***DMCGridOn**[= {True | False}]

### Setting

The DMCGridOn property settings are:

Setting Description	
---------------------	--

True	Displays the grid with the data.
------	----------------------------------

False	No display of the grid. This is the default setting.
-------	--

### Remarks

When you enable the grid, it will be displayed with the DMCGridColor.

### Note

You should set the DMCGridColor before enabling this option.

## Data Type

Integer (Boolean)

---

## **DMCHorizontalSource**

### Description

Sets the source axis for the horizontal graph axis.

### Usage

*[form.]**[control.]***DMCHorizontalSource**[= *setting*]

### Setting

The DMCHorizontalSource property settings are:

Setting Description	
---------------------	--

0	Time. This is the default setting.
---	------------------------------------

1	X.
---	----

2	Y.
---	----

3	Z.
---	----

4	W.
---	----

5	E.
---	----

6	F.
---	----

7	G.
---	----

8	H.
---	----

### Remarks

This property may be used at any time. This property is saved if set at design time.

### Note

This is used to tell the controller which axis to record data from. Be sure to set this before capturing data, that is, before setting the [DMCCaptureData](#) property to True.

### Data Type

Long

---

## **DMCHorizontalTitle**

### Description

Sets the title for the horizontal data.

### Usage

`[form.][control.]DMCHorizontalTitle[= title]`

### Setting

You can set a string up to 80 chars into the title.

### Remarks

This property may be used at any time. This property is saved if set at design time.

### Note

This will be displayed as soon as you set the string.

### Data Type

String

---

## **DMCMaxYValue**

### Description

Sets the maximum Y axis data value to display when the DMCScope control is in "continuous" mode (the [DMCContinuous](#) property is set to True).

### Usage

`[form.][control.]DMCMaxYValue[= value]`

### Setting

The DMCMaxYValue property setting must be an integer within the range –2,147,483,648 through 2,147,483,647. The default setting is 10000.

### Remarks

A change to the DMCMaxYValue property at run-time immediately affects the display of the DMCScope control.

### Note

This property may be used at any time. This property is saved if set at design time.

**Data Type**  
Long

---

## **DMCMinYValue**

### **Description**

Sets the minimum Y axis data value to display when the DMCScope control is in "continuous" mode (the [DMCContinuous](#) property is set to True).

### **Usage**

`[form.][control.]DMCMinYValue[= value]`

### **Setting**

The DMCMinYValue property setting must be an integer within the range –2,147,483,648 through 2,147,483,647. The default setting is 10000.

### **Remarks**

A change to the DMCMinYValue property at run-time immediately affects the display of the DMCScope control.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCPointsPerAxis**

### **Description**

Sets the number of data points to record.

### **Usage**

`[form.][control.]DMCPointsPerAxis[= points]`

### **Setting**

The DMCPointsPerAxis property setting must be an integer within the range 50 through the maximum the controller is capable of. The default setting is 100.

### **Remarks**

This setting will be used on the next recording cycle.

### **Note**

A standard DMC-1000 can hold a total of 1600 points. The DMC-1000-MX can hold total of 8000 points. A DMC-700 can hold a total of 1600 points. A DMC-1500 can hold a total of 1600 points. All Optima controllers can hold up to 8000 points.

### **Data Type**

Long

---

## **DMCSampleTime**

### **Description**

This sets the time between samples.

### **Usage**

[*form.*][*control.*]**DMCSampleTime**[= *setting*]

### **Setting**

The DMCSampleTime property settings are:

<b>Setting Description</b>	
0	2 ms. This is the default setting.
1	4 ms.
2	8 ms.
3	16 ms.
4	32 ms.
5	64 ms.
6	128 ms.
7	256 ms

### **Remarks**

This setting will be used in the next recording cycle.

### **Note**

The times noted here are based upon the default sampling period of 1000 microseconds. To find out the actual sampling period, use the TM command. To find out the actual time between samples, calculate: (Nominal Sample Time)\* (actual sample period) / 1000.

### **Data Type**

Long

---

## **DMCStopOnTrigger**

### **Description**

Determines whether or not to stop the continuous recording and displaying of data when the trigger has been set off.

### **Usage**

[*form.*][*control.*]**DMCStopOnTrigger**[= {True | False}]

### **Setting**

The DMCStopOnTrigger property settings are:

<b>Setting Description</b>	
True	Stop processing when the trigger has been set off. This is the default setting.
False	Continue processing even if the trigger has been set-off.

### **Remarks**

The DMCStopOnTrigger property is used only if both the [DMCContinuous](#) and [DMCTriggerOn](#) properties are set to True.

**Note**

This property may be used at any time. This property is saved if set at design time.

**Data Type**

Integer (Boolean)

---

**DMCTriggerMaxValue****Description**

Sets the maximum value for determining whether the trigger has been set-off when the DMCScope control is in "continuous" mode (the [DMCContinuous](#) property is set to True).

**Usage**

`[form.][control.]DMCTriggerMaxValue[= value]`

**Setting**

The DMCTriggerMaxValue property setting must be a floating-point within the range 1.7E-308 through 1.7E+308. The default setting is 10000000.0.

**Remarks**

If a data point is captured which is greater than or equal to the value of the DMCTriggerMaxValue property, the DMCScope control will send the [DMCTrigger](#) event to your application program.

**Note**

This property may be used at any time. This property is saved if set at design time.

**Data Type**

Double

---

**DMCTriggerMinValue****Description**

Sets the minimum value to for determining whether the trigger has been set-off when the DMCScope control is in "continuous" mode (the [DMCContinuous](#) property is set to True).

**Usage**

`[form.][control.]DMCTriggerMinValue[= value]`

**Setting**

The DMCTriggerMinValue property setting must be a floating-point within the range 1.7E-308 through 1.7E+308. The default setting is -10000000.0.

**Remarks**

If a data point is captured which is less than or equal to the value of the DMCTriggerMinValue property, the DMCScope control will send the [DMCTrigger](#) event to your application program.



## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Double

---

## **DMCTriggerOn**

### Description

Determines whether or not to use a trigger when the DMCScope control is in "continuous" mode (the [DMCContinuous](#) property is set to True)..

### Usage

[form.][control.]**DMCTriggerOn**[= {True | False}]

### Setting

The DMCTriggerOn property settings are:

Setting Description	
---------------------	--

True	Use a trigger.
------	----------------

False	Do not use a trigger. This is the default setting.
-------	--

### Remarks

If the DMCTriggerOn property is set to True, the DMCScope control will fire the [DMCTrigger](#) event if a data point is above the [DMCTriggerMaxValue](#) property value or below the [DMCTriggerMinValue](#) property value. If the [DMCStopOnTrigger](#) property is set to False, the [DMCTrigger](#) event will continue to be fired each time the trigger is "set-off".

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

## **DMCValuesOn**

### Description

If enabled, this will display the values next to the axis.

### Usage

[form.][control.]**DMCValuesOn**[= {True | False}]

### Setting

The DMCValuesOn property settings are:

Setting Description	
---------------------	--

True	Displays tick mark values on graph.
------	-------------------------------------

False	No display of values on graph. This is the default setting.
-------	---

### Remarks

Setting this will have the values of the axis tick marks displayed on the next capture.

## Note

If you display the values, this will use space for display of the values instead of graphing the data.

## Data Type

Integer (Boolean)

---

## **DMCVerticalSource**

### Description

Sets the source axis for the vertical graph axis.

### Usage

[*form.*][*control.*]**DMCVerticalSource**[= *setting*]

### Setting

The DMCVerticalSource property settings are:

Setting	Description
0	X. This is the default setting.
1	Y.
2	Z.
3	W.
4	E.
5	F.
6	G.
7	H.

### Remarks

This property may be used at any time. This property is saved if set at design time.

## Note

This is used to tell the controller which axis to record data from. Be sure to set this before capturing data, that is, before setting the [DMCCaptureData](#) property to True.

## Data Type

Long

---

## **DMCVerticalTitle**

### Description

Allows a title to be set for the vertical axis.

### Usage

[*form.*][*control.*]**DMCHorizontalTitle**[= *title*]

### Setting

You can set a string up to 80 chars into the title.

## Remarks

Default is to have no vertical title.

## Note

The title is displayed left-to-right (instead of top-to-bottom), taking up valuable screen real estate.

## Data Type

String

---

## Galil Events

DMCError

DMCTrigger

## DMCScope Event Descriptions

### DMCError

### Description

Used by the DMCScope control to notify the application program of an error.

### Syntax

Sub *ctlname*\_DMCError(*ErrorNumber*As Long, *ErrorMessage*As String)

### Remarks

The two most frequent errors are:

ErrorNumber	Description
-1	A time-out occurred.
-2	Galil language command error.

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.

2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCScope control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.
3. The Galil motion controller is waiting for a trippoint, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trippoint is cleared. This is not an error.

---

## **DMCTrigger**

### **Description**

Used by the DMCScope control to notify the application program that the trigger has been set off.

### **Syntax**

Sub *ctlname*\_**DMCTrigger**(*DataValue* As Double)

### **Remarks**

The *DataValue* will contain the value of the data point which set-off the trigger (caused the DMCTrigger event to fire).

---

## **DMCScope Examples**

### **Example: Setting the DMCCaptureData Property**

In this example, The DMCCaptureData property will be set to "True", causing data collection to start.

```
' Collect data on the x axis for the horizontal data
DMCScope1.DMCHorizontalSource = 1

' Collect data on the y axis for the vertical data
DMCScope1.DMCVerticalSource = 1

' Set the sample time to 4 ms between points
DMCScope1.DMCSampleTime = 2

' Set the cursor to the hourglass figure
MousePointer = 11

' Start collecting
DMCScope1.DMCCaptureData = True
```

### **Example: Using the DMCScope Control in Continuous Mode**

In this example, The DMCContinuous and DMCCaptureData properties will be set to "True", causing continuous data collection to start.

```
' Set the horizontal source to time
DMCScope1.DMCHorizontalSource = 0
```

```

' Set the vertical source to the x axis
DMCScopel.DMCVerticalSource = 0

' Set the sample time to 8 ms between points
DMCScopel.DMCSampleTime = 2

' Set the display range (this can be changed dynamically)
DMCScopel.DMCMaxYValue = 5000
DMCScopel.DMCMinYValue = -5000

' Set the trigger properties
DMCScopel.DMCTriggerOn = True
DMCScopel.DMCStopOnTrigger = True
DMCScopel.DMCTriggerMaxValue = 4500
DMCScopel.DMCTriggerMinValue = -4500
' Set continuous mode
DMCScopel.DMCContinuous = True

' Start collecting
DMCScopel.DMCCaptureData = True

```

### **Example: Responding to the DMCError Event**

In this example, the application program reports on errors as sent by the Galil motion controller.

```

Sub DMCScopel_DMCError (ErrorNumber As Long, ErrorMessage As String)
    Dim RC As Integer
    RC = MsgBox(ErrorMessage, MB_ICONEXCLAMATION, "DMC Error")
End Sub

```

**THIS PAGE LEFT BLANK INTENTIONALLY**



# Chapter 8 – DMCDiagnostics Control

The diagnostics for axis dependent information object provides a snap shot of selected axis dependent information such as position, error, and velocity. The information can be displayed on command or through polling at regular time intervals. Useful as a diagnostic tool, this control is designed to present a great deal of information in a small area. Current system setup information displayed on this form can be sent to a file on the PC and used to pinpoint the cause of many problems.

	Galil Command	X Axis	Y Axis	Z Axis	W Axis
Actual Position	TP	100	200	300	400
Reference Position	RP	1100	700	800	900
Dual Encoder Position	TD	25	50	80	100
Latched Position	RL	0	0	0	0
Position Error	TE	1000	500	500	500
Stop Code	SC	1	1	1	1
Switches: Axis in Motion	TS (bit 7)	0	0	0	0
Switches: Axis Error Exceeds Error Limit	TS (bit 6)	0	0	0	0
Switches: Motor Off	TS (bit 5)	0	0	0	0
Switches: Undefined	TS (bit 4)	0	0	0	0
Switches: Forward Limit Inactive	TS (bit 3)	1	1	1	1
Switches: Reverse Limit Inactive	TS (bit 2)	1	1	1	1
Switches: Home	TS (bit 1)	1	1	1	1
Switches: Latched	TS (bit 0)	1	1	1	1
Torque	TT	7.3733	3.6866	3.6866	3.6866
Velocity	TV	0	0	0	0

## Galil Properties

DMCController ( OCX only )  
 DMCFilename  
 DMCGridColor  
 DMCCModel ( OCX only )  
 DMCPollController  
 DMCPollInterval  
 DMCPrint  
 DMCTRefreshDisplay  
 DMCSaveToFile  
 DMCSHowCommand  
 DMCSHowEAxis  
 DMCSHowFAxis  
 DMCSHowGAxis  
 DMCSHowHAxis  
 DMCSHowOP ( VBX only )  
 DMCSHowRL

DMCSHowRP  
 DMCSHowSC  
 DMCSHowSetupParameters ( VBX only )  
 DMCSHowTB ( VBX only )  
 DMCSHowTD  
 DMCSHowTE  
 DMCSHowTI ( VBX only )  
 DMCSHowTP  
 DMCSHowTS  
 DMCSHowTT  
 DMCSHowTV  
 DMCSHowWAxis  
 DMCSHowXAxis  
 DMCSHowYAxis  
 DMCSHowZAxis

## Galil Events

DMCError







## Chapter 9 – DMCDiagsNonAxis Control

The diagnostics for axis independent information object provides a snap shot of selected axis independent information such as the state of output and input ports. The information can be displayed on command or through polling at regular time intervals. Use this tool to display and set the I/O states. Like the Diagnostic 1, the information displayed here can be stored in a file for diagnostic purposes.

	Galil Command	Value
1	TB (bit 7)	0
2	TB (bit 6)	1
3	TB (bit 5)	0
4	TB (bit 4)	0
5	TB (bit 3)	0
6	TB (bit 2)	0
7	TB (bit 1)	0
8	TB (bit 0)	0
Output Port 8	MG_OP (port 8)	0
Output Port 7	MG_OP (port 7)	0
Output Port 6	MG_OP (port 6)	0
Output Port 5	MG_OP (port 5)	0
Output Port 4	MG_OP (port 4)	0
Output Port 3	MG_OP (port 3)	0
Output Port 2	MG_OP (port 2)	0
Output Port 1	MG_OP (port 1)	0
Input Port 8	TI (port 8)	1
Input Port 7	TI (port 7)	1
Input Port 6	TI (port 6)	1
Input Port 5	TI (port 5)	1
Input Port 4	TI (port 4)	1
Input Port 3	TI (port 3)	1
Input Port 2	TI (port 2)	1
Input Port 1	TI (port 1)	0
Analog Input 1	AN (input 1)	1.6296
Analog Input 2	AN (input 2)	1.8726
Analog Input 3	AN (input 3)	1.7505
Analog Input 4	AN (input 4)	1.7615
Analog Input 5	AN (input 5)	1.7261
Analog Input 6	AN (input 6)	1.9385
Analog Input 7	AN (input 7)	1.7627

## Galil Properties

DMCController  
DMCFileName  
DMCGridColor  
DMCModel  
DMCPollController  
DMCPollInterval  
DMCPrint

DMCRefreshDisplay  
DMCSaveToFile  
DMCShowAN  
DMCShowDescription  
DMCShowOP  
DMCShowTB  
DMCShowTI

## Galil Events

DMCError



# Chapter 10 – DMCSetup Control

The parameter set up object provides a convenient way to display and/or edit all of the parameters saved in the EEPROM of the controller.

	Galil Command	X Axis	Y Axis	
CONFIGURATION				
Motor Type	MT	Servo motor	Servo motor	S
Main Encoder	CE	Normal Quadrature	Normal Quadrature	N
Auxiliary Encoder	CE	Normal Quadrature	Normal Quadrature	N
Dual Velocity (Dual Loop)	DV	Enabled (On)	Disabled (Off)	C
Forward Software Limit	FL	2147483647	2147483647	
Reverse Software Limit	BL	-2147483648	-2147483648	
FILTER				
Derivative Constant	KD	64.000	64.000	
Proportional Constant	KP	6.000	6.000	
Integrator	KI	0.000	0.000	
Integrator Limit	IL	9.9982	9.9982	
Torque Limit	TL	9.9982	9.9982	
Offset	OF	0.000	0.000	
Error Limit	ER	16384	16384	
Off on Error	OE	Disabled (Off)	Disabled (Off)	C
Acceleration Feedforward	FA	0	0	
Velocity Feedforward	FV	0	0	
Motor State	MO	On	On	C
Step Motor Smoothing	KS	1	1	
INDEPENDENT MOTION				
Acceleration	AC	256000	256000	
Deceleration	DC	256000	256000	
Speed	SP	25000	25000	
Gear Ratio	GR	0.0000	0.0000	
Independent Time Constant	IT	1.000	1.000	
Vector Time Constant	VT	1.000	1.000	
OTHER CONFIGURATION				
Configure Limit Switch	CN	Active Low		
Configure Home Switch	CN	Active Low		
Configure Latch Input	CN	Active Low		

## Galil Properties

DMCChangeImmediate  
DMCClear  
DMCController  
DMCFileName  
DMCGetParameters  
DMCGridColor  
DMCLoadFromFile  
DMCModel  
DMCPrint  
DMCReadOnly  
DMCSaveToFile

DMCSetParameters  
DMCShowCommand  
DMCShowEAxis  
DMCShowFAxis  
DMCShowGAxis  
DMCShowHAxis  
DMCShowWAxis  
DMCShowXAxis  
DMCShowYAxis  
DMCShowZAxis

## Galil Events

DMCError



# Chapter 11 – DMCArray Control

Use this tool to upload an array from the controller to the PC or download from the PC to the controller. This tool simplifies the process of recording data into an array within the controller and allows infinite data recording using the PC. Any data type that can be used in the automatic data capture mode can be used with this tool. If required, this tool can be used as the 'teach' portion of a teach and playback application. ( For infinite recording the DMC-1000 controller firmware must be version 1.5 or higher. )

## Galil Properties

[DMCAddPlaybackHeaderToFile](#)  
[DMCArrayData](#)  
[DMCArrayIndex](#)  
[DMCArrayName](#)  
[DMCArrayOperation](#)  
[DMCAxisToRecord](#)  
[DMCCaptureDataToBuffer](#)  
[DMCCaptureDataToFile](#)  
[DMCUseSystemTimer](#)  
[DMCController](#)

[DMCDataType](#)  
[DMCElementDelimiter](#)  
[DMCFileName](#)  
[DMCFirstElement](#)  
[DMCLastElement](#)  
[DMCRecordData](#)  
[DMCRecordTime](#)  
[DMCSampleTime](#)  
[DMCTotalElements](#)

## DMCArray Property Descriptions

### **DMCAddPlaybackHeaderToFile**

#### **Description**

Used to add a playback header record to the captured data file when the [DMCRecordData](#) property is set to 1 "Record into File and/or Buffer", and the [DMCCaptureDataToFile](#) property is set to True.

#### **Usage**

[form.][control.]**DMCAddPlaybackHeaderToFile**[={True | False}]

#### **Setting**

The DMCAddPlaybackHeaderToFile property settings are:

<b>False</b>	Do not add playback header to the captured data file.
<b>True</b>	Add playback header to the captured data file. This is the Default setting.

#### **Remarks**

If the DMCAddPlaybackHeaderToFile property is set to True, a header record is added to the captured data file (as the first record) which is required for playback by the DMCPlayback control. The header record contains information on sample time and axes recorded.

The DMCAddPlaybackHeaderToFile property has no effect unless the [DMCCaptureDataToFile](#) property is set to True.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

## **DMCController**

### Description

The number corresponding to the Galil motion controller registered in the Windows registry database.

### Usage

`[form.][control.]DMCController[= controller]`

### Setting

The setting for the DMCController property must be an integer between 1 and 16. The default setting is 1.

### Remarks

A Galil motion controller can be registered in the Windows registry database by using the DMCREG 16 or DTERM16 programs for Windows 3.x or the DMCREG32 or DTERM32 programs for Windows 95, 98, NT, or 2000. Galil motion controllers can also be registered using the DMCRegister control (OCX).

## Note

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

## Data Type

Long

---

## **DMCArrayData**

### Description

Used to send or receive data from a data array.

### Usage

`[form.][control.]DMCArrayData[= array data]`  
`array data = [form.][control.]DMCArrayData`

### Setting

The setting for the DMCArrayData property may be any string up to 64K (approximately 65,500) bytes in size.

### Remarks

The DMCArrayData is used as an output field for the [DMCArrayOperation](#) 3 "Array to Buffer" and an input field for the [DMCArrayOperation](#) 4 "Buffer to Array". When the DMCArrayData property is used as input data, it is expected that array elements will be separated by the element delimiter as specified by the [DMCElementDelimiter](#) property. Likewise, array elements are separated by the element delimiter as specified by the [DMCElementDelimiter](#) property when the DMCArrayData property is used as output data.

**Note**

This property may be used at any time.

**Data Type**

String

---

**DMCArrayIndex****Description**

Used to determine the current or active data array for the [DMCArrayOperation](#) property.

**Usage**

*[form.][control.]DMCArrayIndex*[= array index]

**Setting**

The setting for the DMCArrayIndex property is an integer from 0 to 4 or 0 to 8 depending on the model Galil Motion Controller installed. If the DMCArrayIndex property is set to a number outside of these boundaries, the DMCArray control will return the Visual Basic error code 380 "Invalid property array index". The default value is 0 or the first data array.

The maximum number of data arrays (that is, Galil data arrays) per DMCArray control is determined by the model Galil Motion Controller installed. The limit is 4 for the DMC-700 series and the DMC-1010 through the DMC-1040. The limit is 8 for the DMC-1050 through DMC-1080, the DMC-1000 MX, the DMC-1500 series, and all Optima series controllers. Any attempt to access a property array with a subscript less than 0 or greater than the maximum number of data arrays will generate an error: Visual Basic error code 380 "Invalid property array index". This property may be used at any time.

**Remarks**

The DMCArrayIndex property must be set before the [DMCArrayOperation](#).property can be used.

**Note**

This property is not visible at design time. This property may be used at any time.

**Data Type**

Long

---

## **DMCArrayControl>MainDMCArrayName**

### **Description**

Used to assign a name to a data array.

### **Usage**

[*form.*][*control.*]**DMCArrayName**(*index*)[= *array name*]

### **Setting**

The setting for the DMCArrayName property may be any 8 characters, starting with an uppercase alphabetic character. Note that a data array name is a string.

### **Remarks**

The DMCArrayName property must be set before the [DMCArrayOperation](#) property can be used.

### **Note**

This property is actually a property array, and therefore, is not visible in the Visual Basic properties window. It can only be referenced with a subscript. The current instance is defined by the value of the [DMCArrayIndex](#) property.

The maximum number of data arrays (that is, Galil data arrays) per DMCArray control is determined by the model Galil Motion Controller installed. The limit is 4 for the DMC-700 series and the DMC-1010 through the DMC-1040. The limit is 8 for the DMC-1050 through DMC-1080, the DMC-1000 MX, the DMC-1500 series, and all Optima series controllers. Any attempt to access a property array with a subscript less than 0 or greater than the maximum number of data arrays will generate an error: Visual Basic error code 380 "Invalid property array index". This property may be used at any time.

### **Data Type**

String

---

## **DMCArrayOperation**

### **Description**

Used to perform an array operation for the current or active data array as defined by the value of the [DMCArrayIndex](#) property.

### **Usage**

[*form.*][*control.*]**DMCArrayOperation**[= *file operation*]

### **Setting**

The DMCArrayOperation property settings are:

<b>Setting Description</b>	
0	None. This is the default setting.
1	Array to File.
2	File to Array.
3	Array to Buffer.
4	Buffer to Array.
5	Define.
6	Expand.



7	Undefine.
8	Undefine All.

## Remarks

With the exception of array operation 8 "Undefine All" which pertains to all currently defined data arrays, the array operation will pertain to the current or active data array as defined by the value of the [DMCArrayIndex](#) property. Before array operations 1 "Array to File", 2 "File to Array", 3 "Array to Buffer", 4 "Buffer to Array", 6 "Expand", and 7 "Undefine" may be used, at least one data array must have already been defined.

Using a comma as a data array element delimiter will generally yield the fastest file to array and buffer to array performance. For more information, see the [DMCElementDelimiter](#) property.

Errors are reported via the [DMCError](#) event.

Once the array operation has completed, the [DMCArrayOperation](#) property is reset to 0 by the [DMCArray](#) control.

## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

## Data Type

Long

## DMCAxisToRecord

## Description

The axis to record for a record data operation (the [DMCRecordData](#) property is set to 1 "Record into File and/or Buffer" or 2 "Record into Array").

## Usage

*[form.]*[\[control.\]DMCAxisToRecord\(index\)](#)*[= axis to record]*

## Setting

The [DMCAxisToRecord](#) property settings are:

Setting	Description
0	None. This is the default setting.
1	X.
2	Y.
3	Z.
4	W.
5	E.
6	F.
7	G.
8	H

## Remarks

The [DMCAxisToRecord](#) property must be set before the [DMCRecordData](#) property can be used.

## Note

This property is actually a property array, and therefore, is not visible in the Visual Basic properties window. It can only be referenced with a subscript. The current instance is defined by the value of the [DMCArrayIndex](#) property.

The maximum number of data arrays (that is, Galil data arrays) per DMCArray control is determined by the model Galil Motion Controller installed. The limit is 4 for the DMC-700 series and the DMC-1010 through the DMC-1040. The limit is 8 for the DMC-1050 through DMC-1080, the DMC-1000 MX , DMC-1500 series, and all Optima series controllers. Any attempt to access a property array with a subscript less than 0 or greater than the maximum number of data arrays will generate an error: Visual Basic error code 381 "Invalid property array index". This property may be used at any time.

## Data Type

Long

---

## **DMCCaptureDataToBuffer**

### Description

Used to capture data to a buffer when the [DMCRecordData](#) property is set to 1 "Record into File and/or Buffer".

### Usage

[*form.*][*control.*]**DMCCaptureDataToBuffer**[={True | False}]

### Setting

The DMCCaptureDataToBuffer property settings are:

Setting	Description
---------	-------------

False	Do not capture data to a buffer. This is the Default setting.
True	Capture data to a buffer.

### Remarks

The application receives data through the [DMCDataAvailable](#) event. The DMCArray control will periodically fire this event as data is retrieved from the Galil motion controller.

Both the DMCCaptureDataToBuffer property and the [DMCCaptureDataToFile](#) property may be set to True at the same time. That is, data may be captured to both a buffer and a file simultaneously.

The DMCCaptureDataToBuffer property is used only for [DMCRecordData](#) property value 1 "Record into File and/or Buffer". It is ignored for [DMCRecordData](#) property value 2 "Record into Array".

### Note

This property may be used at any time This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

## **DMCCaptureDataToFile**

### Description

Used to capture data to a file when the [DMCRecordData](#) property is set to 1 "Record into File and/or Buffer".

### Usage

[*form.*][*control.*]**DMCCaptureDataToFile**[={True | False}]

## Setting

The DMCCaptureDataToFile property settings are:

Setting	Description
---------	-------------

False	Do not capture data to a file.
True	Capture data to a file. This is the Default setting.

## Remarks

In order to use the DMCCaptureDataToFile property, a file name must first be assigned to the [DMCFileName](#) property.

The format for the captured data file is the following: columns are delimited by tabs and rows are delimited by new lines. This allows the file to be opened by programs such as Microsoft Excel and have the data format immediately recognized as that of a table. Columns are the axes and their data types recorded such as TPX and TPY, and rows are the instances of the data captured at the specified recording interval such as 2 ms, 4 ms, etc..

If the [DMCAddPlaybackHeaderToFile](#) property is set to True, a header record is added to the captured data file (as the first record) which is required for playback by the DMCPlayback control. The header record contains information on sample time and axes recorded.

The DMCCaptureDataToFile property is used only for [DMCRecordData](#) property value 1 "Record into File and/or Buffer". It is ignored for [DMCRecordData](#) property value 2 "Record into Array".

Both the DMCCaptureDataToFile property and the [DMCCaptureDataToBuffer](#) property may be set to True at the same time. That is, data may be captured to both a file and a buffer simultaneously.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

## DMCDataType

### Description

The data type for a record data operation (the [DMCRecordData](#) property is set to 1 "Record into File and/or Buffer" or 2 "Record into Array").

### Usage

*[form.][control.]DMCDataType(index)[= data type]*

## Setting

The DMCDataType property settings are:

Setting	Description
---------	-------------

0	None. This is the Default setting.
1	Position.
2	Position Error.
3	Commanded Position.
4	Latched Position.
5	Dual Encoder Position.
6	Inputs.

7	Outputs.
8	Switches.
9	Stop Code.
.	Torque.
.	Analog Inputs

## Remarks

The DMCDatatype property must be set before the [DMCRecordData](#) property can be used.

## Note

This property is actually a property array, and therefore, is not visible in the Visual Basic properties window. It can only be referenced with a subscript. The current instance is defined by the value of the [DMCArrayIndex](#) property.

Analog input recording is limited by the number of axes on the controller. For example, a DMC-1750 is able to record 5 inputs with this tool.

The maximum number of data arrays (that is, Galil data arrays) per DMCArray control is determined by the model Galil Motion Controller installed. The limit is 4 for the DMC-700 series and the DMC-1010 through the DMC-1040. The limit is 8 for the DMC-1050 through DMC-1080, the DMC-1000 MX, the DMC-1500 series, and all Optima series controllers. Any attempt to access a property array with a subscript less than 0 or greater than the maximum number of data arrays will generate an error: Visual Basic error code 381 "Invalid property array index". This property may be used at any time.

## Data Type

Long

---

## **DMCElementDelimiter**

## Description

The delimiter or character to use to distinguish between consecutive data array elements.

## Usage

*[form.]*[*control.*]**DMCElementDelimiter**[= *delimiter*]

## Setting

The DMCElementDelimiter property settings are:

Setting Description	
0	Comma. This is the default setting.
1	New Line.
2	Tab.
3	Space.
4	Semi-Colon.

## Remarks

The DMCElementDelimiter property is used for [DMCArrayOperations](#) 1 "Array to File", 2 "File to Array", 3 "Array to Buffer", and 4 "Buffer to Array". Data array elements are expected to be delimited by the designated delimiter on input and are delimited by the designated delimiter on output. Using a comma as a data array element delimiter will generally yield the fastest file to array and buffer to array performance.

## Note

This property may be used at any time. This property is saved if set at design time.

**Data Type**  
Long

---

## **DMCFileName**

### **Description**

Used to set the file name for [DMCArrayOperations](#) 1 "Array to File" and 2 "File to Array" and [DMCRecordData](#) operation 1 "Record into File and/or Buffer " when the [DMCCaptureDataToFile](#) property is set to True.

### **Usage**

*[form.][[control.]DMCFileName[= file name]*

### **Setting**

The setting for the DMCFileName property may be any valid DOS file name, including the path. Note that a file name is a string.

### **Remarks**

The DMCFileName property must be set before using the [DMCArrayOperations](#) 1 "Array to File" and 2 "File to Array", or before data can be captured to a file with record data operation 1 "Record into File and/or Buffer ".

### **Note**

This property may be used at any time. This property is saved if set at design time.

**Data Type**  
String

---

## **DMCFirstElement**

### **Description**

The first data array element to access during the [DMCArrayOperations](#) 1 "Array to File", 2 "File to Array", 3 "Array to Buffer", and 4 "Buffer to Array".

### **Usage**

*[form.][[control.]DMCFirstElement[= element]*

### **Setting**

The setting for the DMCFirstElement property may be any number between 0 and the value of the [DMCTotalElements](#) property for the current or active data array. The default setting is 0 or the first data array element.

### **Remarks**

The DMCFirstElement and [DMCLastElement](#) properties are useful for defining a subset of a data array to transfer data to or from.

### **Note**

This property may be used at any time. This property is saved if set at design time.

**Data Type**  
Long

---

## **DMCLastElement**

### **Description**

The last data array element to access during the [DMCArrayOperations](#) 1 "Array to File", 2 "File to Array", 3 "Array to Buffer", and 4 "Buffer to Array".

### **Usage**

[*form.*][*control.*]**DMCLastElement**[= *element*]

### **Setting**

The setting for the DMCLastElement property may be any number between 0 and the value of the [DMCTotalElements](#) property for the current or active data array. The default setting is -1 or the last data array element.

### **Remarks**

The [DMCFirstElement](#) and DMCLastElement properties are useful for defining a subset of a data array to transfer data to or from. A value of -1 for the DMCLastElement property means the last data array element.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCRecordData**

### **Description**

Used to record data into a file and/or buffer or a data array.

### **Usage**

[*form.*][*control.*]**DMCRecordData**[= *record operation*]

### **Setting**

The DMCRecordData property settings are:

<b>Setting Description</b>	
0	Record Off. This is the default setting.
1	Record into File and/or Buffer.
2	Record into Array.

### **Remarks**

The data recording feature of the DMCArray control may be used in two modes through the DMCRecordData property. When the DMCRecordData property is set to 1 "Record into File and/or Buffer", data (as specified by the [DMCDataType](#) property) is recorded to a file and/or buffer and is limited in duration by the value of the [DMCRecordTime](#) property. If the [DMCRecordTime](#) property is set to -1, data is recorded until the DMCRecordData property is explicitly set to 0 "Record Off".

When the DMCRecordData property is set to 2 "Record into Array", data is recorded to one or more data arrays and is limited by the size or dimension of the data arrays.

**Important:** To speed up the start of recording, call the [GetLimits](#) method before setting the `DMCRecordData` property. This will enable the [DMCArray control](#) to collect required statistics about the host PC and the Galil motion controller before it starts recording. The [GetLimits](#) method only needs to be invoked one per application instance so it can be placed in the Form Load procedure of your main Visual Basic form. It should be called after a connection to the controller has already been made.

## Recording Data into a File and/or Buffer

The `DMCArray` control determines what data to record by looking at the [DMCAxisToRecord](#), and [DMCDataType](#) properties. Since these are property arrays, the `DMCArray` control starts with a property array index equal to 0 and looks for the last property array index where all of the above properties have non-zero or non-null values. The `DMCArray` control uses its own data arrays during this record data operation so no data arrays need to be previously defined. In fact, the `DMCArray` control will free all currently allocated array space before beginning this record data operation. If the [DMCCaptureDataToFile](#) property is set to True, data will be written to a file. The [DMCFileName](#) property provides the file name. If the [DMCCaptureDataToBuffer](#) property is set to True, data will be written to a buffer. The data is made available to the application program via the [DMCDataAvailable](#) event. Data may be written to a file and buffer simultaneously. If recording is limited by time as specified by the [DMCRecordTime](#) property, recording is stopped and the `DMCRecordData` property reset to 0 "Record Off" when the time limit expires.

The `DMCArray` control will attempt to make sure that the desired recording can take place without losing samples from the Galil motion controller and issue a warning if it believes the sample rate (via the [DMCSampleTime](#) property) is too fast for the amount of data to be recorded. Ultimately, it is your responsibility to make sure that samples have not been lost by inspecting the data that was recorded.

## Recording Data into One or More Arrays

The `DMCArray` control determines what data to record and which data arrays to record into by looking at the [\\_DMCArrayName](#), [DMCTotalElements](#), [DMCAxisToRecord](#), and [DMCDataType](#) properties. Since these are property arrays, the `DMCArray` control starts with a property array index equal to 0 and looks for the last property array index where all of the above properties have non-zero or non-null values. You must first define the data arrays you wish to record data into before attempting to use the `DMCRecordData` property. Recording is complete when all data array elements have been used. The `DMCRecordData` property is reset to 0 "Record Off" when that occurs.

Errors are reported via the [DMCError](#) event.

## Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

## Data Type

Long

---

## DMCRecordTime

### Description

Used to specify the record time in seconds for the record data into file and/or buffer operation (the [DMCRecordData](#) property is set to 1 "Record into File and/or Buffer").

### Usage

`[form.][control.].DMCRecordTime [= record time]`



## Setting

The setting for the DMCRecordTime property may be any positive integer up to 2,147,483,647, or -1. The default setting is -1.

## Remarks

A setting of -1 for the DMCRecordTime property means that recording will continue until the application program sets the [DMCRecordData](#) property to 0 "Record Off".

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## DMCSampleTime

## Description

The recording interval to use for record data operations (the [DMCRecordData](#) property is set to 1 "Record into File and/or Buffer" or 2 "Record into Array").

## Usage

*[form.][control.]DMCSampleTime[= sample time]*

## Setting

The DMCSTime property settings are:

Setting	Description
0	2 ms. This is the default setting.
1	4 ms.
2	8 ms.
3	16 ms.
4	32 ms.
5	64 ms.
6	128 ms.
7	256 ms.

## Remarks

If you are recording into more than 2 data arrays, or you are using [DMCRecordData](#) operation 1 "Record into File and/or Buffer" and you have data array space limited to 1600 data array elements (because of the model Galil motion controller installed), you will most likely have to use a sample time greater than 2 or 4 ms. It is possible to lose samples when it takes significantly longer to process a series of data than it does to create it.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCTotalElements**

### **Description**

The number of data array elements for a given data array.

### **Usage**

*[form.]**[control.]***DMCTotalElements**(*index*)[= *elements*]

### **Setting**

The setting for the DMCTotalElements property may be any number between 1 and the data array space capacity of the installed Galil motion controller.

### **Remarks**

The DMCTotalElements property must be set before the [DMCArrayOperations](#) 5 "Define" or 6 "Expand" can be used. For both array operations, the current or active data array as defined by the value of the [DMCArrayIndex](#) property is created with or enlarged to the value of the DMCTotalElements property.

The total number of data array elements for all data arrays (data array space capacity) is determined by the model Galil Motion Controller installed. The limit is 1600 for the DMC-700 series and the DMC-1010 through the DMC-1040. The limit is 8000 for the DMC-1050 through DMC-1080, the DMC-1000 MX , the DMC-1500 series and all Optima series controllers.

### **Note**

This property is actually a property array, and therefore, is not visible in the Visual Basic properties window. It can only be referenced with a subscript. The current instance is defined by the value of the [DMCArrayIndex](#) property.

The maximum number of data arrays (that is, Galil data arrays) per DMCArray control is determined by the model Galil Motion Controller installed. The limit is 4 for the DMC-700 series and the DMC-1010 through the DMC-1040. The limit is 8 for the DMC-1050 through DMC-1080, the DMC-1000 MX , and the DMC-1500 series. Any attempt to access a property array with a subscript less than 0 or greater than the maximum number of data arrays will generate an error: Visual Basic error code 381 "Invalid property array index". This property may be used at any time.

### **Data Type**

Long

---

## **DMCUseSystemTimer Property**

### **Applies To**

DMCPoll control property under the DMCArray control

### **Description**

Used to determine which type of timer to use for polling.

## Usage

[form.][control.]DMCUseSystemTimer[= {True | False}]

## Setting

The DMCUseSystemTimer property settings are:

Setting	Description
True	Use a high-resolution system timer for polling.
False	Use a standard windows timer for polling. This is the default setting.

## Remarks

The DMCPoll control by default uses a standard Windows timer to poll the Galil motion controller. The standard Windows timer is not very accurate, but it does not consume much in the way of resources. As an option, the DMCPoll control can use a high-resolution system timer to poll the controller. The high-resolution timer is accurate to within +/- 5ms and will allow you use much faster poll intervals. The trade-off is that this type of timer consumes much more resources. It is not recommended that more than four DMCPoll controls be used simultaneously with the DMCUseSystemTimer property set to True. Most Windows versions will allow up to eight.

## Note

This property may only be used when the DMCPollController property is set to False. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

## Galil Methods

GetLimits

## DMCArray Method Description

### GetLimits

## Description

Used to gather performance statistics about the host PC and Galil motion controller which is used to determine the limits on the [DMCRecordData](#) and [DMCArrayOperation](#) properties.

## Usage

`[form.][control.]GetLimits`

## Remarks

Calling this method is optional as the [DMCRecordData](#) and [DMCArrayOperation](#) properties will call it automatically if the required performance statistics have not already been acquired. However, calling this method directly at start-up time will speed up the [DMCRecordData](#) and [DMCArrayOperation](#) properties (for the first time only). The [DMCArray](#) control only needs to gather this data once per application instance.

## Note

This method may only be used at run time.

---

## Galil Events

DMCError  
DMCDataAvailable

## DMCArray Event Descriptions

### DMCError

## Description

Used by the DMCArray control to notify the application program of an error.

## Syntax

Sub *ctlname\_DMCError*(*ErrorNumber* As Long, *ErrorMessage* As String)

## Remarks

Below is a list of possible errors. A complete list of error numbers can be found in the file DMCCOM40.BAS for Visual Basic programmers and the file DMCCOM.H for C/C++ programmers.

ErrorNumber	Description
-1	A time-out occurred.
-2	Galil language command error.
-3	Invalid controller number.
-4	Error opening or saving a file.
-5	Device driver failure.
-6	Invalid controller handle.
-7	Could not load required dynamic link library.
-8	Out of memory.
-9	User provided data buffer is almost full.
-10	User provided data buffer is full and data truncation has occurred.

Invalid controller handle errors most often occur because a connection to a Galil motion controller has not been established. You must use the DMCSHELL control to establish a communications session.

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCArray control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.
3. The Galil motion controller is waiting for a trip-point, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trip-point is cleared. This is not an error.

---

## **DMCDataAvailable**

### **Description**

Used by the DMCArray control to notify the application program that data is available when [DMCRecordData](#) operation 1 "Record into File and/or Buffer " is active.

### **Syntax**

Sub *ctlname*\_DMCDataAvailable(*ArrayDataAs* String)

### **Remarks**

ArrayData contains all the data recorded by the Galil motion controller for the last 25 ms (approximately). Therefore, depending on the recording interval used (see the [DMCSampleTime](#) property), ArrayData may contain 0 to 25 data points for each axis/data type combination recorded.

In order to receive the DMCDataAvailable event, you must set the [DMCCaptureDataToBuffer](#) property to True before setting the [DMCRecordData](#) property to 1 "Record into File and/or Buffer ".

## DMCArray Examples

### Example: Responding to the DMSError Event

In this example, the application program reports on errors as sent by the Galil motion controller.

```
Sub DMCMove1_DMSError (ErrorNumber As Long, ErrorMessage As String)
    Dim RC As Integer
    RC = MsgBox(ErrorMessage, MB_ICONEXCLAMATION, "DMC Error")
End Sub
```

### Example: Responding to the DMCDataAvailable Event

In this example, the application program is placing the array data from the Galil motion controller in a text control named Text1.

```
Sub DMCArray1_DMCDataAvailable (ArrayData As String)
    Text1.Text = ArrayData
End Sub
```

### Example: Array Operations

This example demonstrates the various array operations of the DMCArray control.

#### ' Define 2 arrays

```
DMCArray1.DMCArrayName(0) = "ARRAY0"
DMCArray1.DMCTotalElements(0) = 500
DMCArray1.DMCArrayIndex = 0
DMCArray1.DMCArrayOperation = 5
```

```
DMCArray1.DMCArrayName(1) = "ARRAY1"
DMCArray1.DMCTotalElements(1) = 500
DMCArray1.DMCArrayIndex = 1
DMCArray1.DMCArrayOperation = 5
```

#### ' Expand 1 array

```
DMCArray1.DMCTotalElements(0) = DMCArray1.DMCTotalElements(0) + 20
DMCArray1.DMCArrayIndex = 0
DMCArray1.DMCArrayOperation = 6
```

#### ' Delete 2 arrays

```
DMCArray1.DMCArrayName(0) = "ARRAY0"
DMCArray1.DMCArrayIndex = 0
DMCArray1.DMCArrayOperation = 7
```

```
DMCArray1.DMCArrayName(0) = "ARRAY1"
DMCArray1.DMCArrayIndex = 1
DMCArray1.DMCArrayOperation = 7
```

#### ' Delete all arrays

```
DMCArray1.DMCArrayOperation = 8
```

#### ' Move data from an array to a file

```
DMCArray1.DMCFileName = "data.txt"
DMCArray1.DMCArrayOperation = 1
```

#### ' Move data from a file to an array

```
DMCArray1.DMCFileName = "data.txt"
DMCArray1.DMCArrayOperation = 2
```

```
' Move data from an array to a buffer
DMCArray1.DMCArrayOperation = 3
Text1.Text = DMCArray1.DMCArrayData
```

```
' Move data from a buffer to an array
DMCArray1.DMCArrayData = Text1.Text
DMCArray1.DMCArrayOperation = 4
```

## Example: Array Recording

This example demonstrates the data recording features of the DMCArray control.

### Record data into arrays:

```
' Call the GetLimits method to allow the DMCArray control to gather required
statistics before starting to record data
DMCArray1.GetLimits
```

```
' Define 2 arrays
DMCArray1.DMCArrayName(0) = "ARRAY0"
DMCArray1.DMCTotalElements(0) = 500
DMCArray1.DMCArrayIndex = 0
DMCArray1.DMCArrayOperation = 5
```

```
DMCArray1.DMCArrayName(1) = "ARRAY1"
DMCArray1.DMCTotalElements(1) = 500
DMCArray1.DMCArrayIndex = 1
DMCArray1.DMCArrayOperation = 5
```

```
' Download a DMC program to the controller and execute
DMCShell1.DMCFileName = "step.dmc"
DMCShell1.DMCFileOperation = 1
DMCShell1.DMCCommand = "XQ"
```

```
' Record the position of X and Y at 8 ms intervals
DMCArray1.DMCAxisToRecord(0) = 1
DMCArray1.DMCDataType(0) = 1
DMCArray1.DMCAxisToRecord(1) = 2
DMCArray1.DMCDataType(1) = 1
DMCArray1.DMCSampleTime = 2
DMCArray1.DMCRecordData = 2
```

```
' While the DMCArray control is still recording, give control back to Windows
Do While DMCArray1.DMCRecordData
    DoEvents
Loop
```

```
' Recording has completed, stop executing the DMC program
DMCShell1.DMCCommand = "ST"
```

### Record data into a file (infinite data recording feature):

```
' Download a DMC program to the controller and execute
DMCShell1.DMCFileName = "step.dmc"
DMCShell1.DMCFileOperation = 1
DMCShell1.DMCCommand = "XQ"
```

```
' Record the commanded position of X and Y at 4 ms intervals
```

```

DMCArray1.DMCAxisToRecord(0) = 1
DMCArray1.DMCDataType(0) = 2
DMCArray1.DMCAxisToRecord(1) = 2
DMCArray1.DMCDataType(1) = 2
DMCArray1.DMCFileName = "data.txt"
DMCArray1.DMCCaptureDataToFile = True
DMCArray1.DMCSampleTime = 1

' Turn on recording
DMCArray1.DMCRecordData = 1

' Turn off recording
DMCArray1.DMCRecordData = 0

' Stop executing the DMC program
DMCShell1.DMCCommand = "ST"

'Upload data from an array to a buffer then display the data in a text box
DMCArray1.DMCArrayName(0) = "TEST"
DMCArray1.DMCTotalElements(0) = 10
DMCArray1.DMCArrayIndex = 0
DMCArray1.DMCFirstElement = 0
DMCArray1.DMCLastElement = 9
DMCArray1.DMCElementDelimiter = ElementDelimiterComma
DMCArray1.DMCArrayOperation = ArrayOperationArraytoBuffer
Text1.Text = DMCArray1.DMCArrayData

```

See Teach and Playback Example Listed Under the Playback Tool





# Chapter 12 – DMCPlayback Control

This tool takes a file created by the Array record tool and plays it back; moving the motors. Two modes of playback, contour and linear interpolation, are available. Contour mode will playback to recorded motion exactly, leaving all pauses and infinite accelerations. Linear mode playback will run through all the points with a constant vector speed. File types include comma delimited text without Galil command prefixes (nnnn,nnnn,nnnn...), vector mode commands prefixed with VPs and CRs or linear interpolation mode commands prefixed with LI.

## Galil Properties

DMCController  
DMCInputFileName  
DMCOutputFileFormat  
DMCOutputFileName  
DMCPlaybackMode  
DMCPlaybackOperation  
DMCPlaybackRate

DMCPrefixCommands  
DMCSegmentsBeforeBeginMotion  
DMCSegmentsPerBlock  
DMCSuffixCommands  
DMCVectorAcceleration  
DMCVectorDeceleration  
DMCVectorSpeed

## DMCPlayback Property Descriptions

### DMCController

#### Description

The number corresponding to the Galil motion controller registered in the Windows registry database.

#### Usage

[form.][control.]**DMCController**[= controller]

#### Setting

The setting for the DMCController property must be an integer between 1 and 16. The default setting is 1.

#### Remarks

A Galil motion controller can be registered in the Windows registry database by using the DMCREG 16 or DTERM16 programs for Windows 3.x or the DMCREG32 or DTERM32 programs for Windows 95, 98, NT, or 2000. Galil motion controllers can also be registered using the DMCRegister control (OCX).

#### Note

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

## Data Type

Long

---

## **DMCInputFileName**

### Description

Used to set the name of the input file which contains the motion data to be converted into a DMC file.

### Usage

*[form.][control.]DMCInputFileName[= input file name]*

### Setting

The setting for the DMCInputFileName property may be any valid DOS file name, including the path. Note that a file name is a string.

### Remarks

The DMCInputFileName property must be set before setting the [DMCPlaybackOperations](#) property to 1 "Convert File".

### Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

String

---

## **DMCOutputFileFormat**

### Description

Used to set the format of the output file to either DMC send file format or DMC download file format when converted using the DMCFileOperation=1 command.

### Usage

*[form.][control.]DMCOutputFileFormat[= output file format]*

### Setting

The DMCOutputFileFormat property settings are:

Setting Description	
0	Send (*.SEN). This is the default setting.
1	Download (*.DMC).

### Remarks

The DMCOutputFileFormat property must be set to 0 "Send (\*.SEN)" if you wish to playback the file using the DMCPlayback control (setting the [DMCPlaybackOperations](#) property to 2 "Playback File"). Files can not be played back to the Galil motion controller by the DMCPlayback control unless they are in DMC send file format. If the input file (motion data) is converted to DMC download file format, the output file may be

downloaded to the Galil motion controller and executed by using the DMCSHELL control. By convention, DMC send files used the filename suffix "SEN" and DMC download files use the filename suffix "DMC".

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

### Description

Used to set the name of the output file (DMC file) which contains the Galil language commands after the input file (motion data) has been converted.

### Usage

[form.][control.]**DMCOutputFileName**[= *output file name*]

### Setting

The setting for the DMCOutputFileName property may be any valid DOS file name, including the path. Note that a file name is a string.

### Remarks

The DMCOutputFileName property must be set before converting the input file (setting the [DMCPlaybackOperations](#) property to 1 "Convert File") or playing back the output file (setting the [DMCPlaybackOperations](#) property to 1 "Playback File").

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

String

---

## **DMCPlaybackMode**

### Description

Used to determine if the input file (motion data) will be converted to a DMC file using the contour mode or linear interpolation mode of motion.

### Usage

[form.][control.]**DMCPlaybackMode**[= *mode*]

### Setting

The DMCPlaybackMode property settings are:

Setting	Description
0	Contour. This is the default setting.
1	Linear Interpolation.

### Remarks

If the DMCPlaybackMode property is set to 0 "Contour", the following property is used by the DMCPlayback control when converting the input file: DMCPlaybackRate. If the DMCPlaybackMode property is set to 1 "Linear Interpolation", the following properties are used by the DMCPlayback control when converting the

input file: [DMCSegmentsBeforeBeginMotion](#), [DMCVectorAcceleration](#), [DMCVectorDeceleration](#), and [DMCVectorSpeed](#).

The [DMCPlaybackMode](#) property must be set before converting the input file (setting the [DMCPlaybackOperations](#) property to 1 "Convert File").

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## DMCPlaybackOperation

### Description

Used to convert the input file (motion data) to a DMC file, or playback the DMC file to the Galil motion controller.

### Usage

*[form.][control.]DMCPlaybackOperation[= playback operation]*

### Setting

The [DMCPlaybackOperation](#) property settings are:

Setting	Description
0	None. This is the default setting.
1	Convert File.
2	Playback File.
3	Pause Playback.
4	Resume Playback.

### Remarks

Playback operation 1 "Convert File" converts the input file (motion data) to a DMC file. For playback operation 1 "Convert File", the following properties must have already been set: [DMCInputFileName](#), [DMCOutputFileName](#), [DMCOutputFileFormat](#), [DMCPrefixCommands](#), and [DMCSuffixCommands](#). In addition, if the [DMCPlaybackMode](#) property is set to 0 "Contour", the following property must have already been set: [DMCPlaybackRate](#). If the [DMCPlaybackMode](#) property is set to 1 "Linear Interpolation", the following property must have already been set: [DMCSegmentsBeforeBeginMotion](#), [DMCVectorAcceleration](#), [DMCVectorDeceleration](#), and [DMCVectorSpeed](#).

Playback operation 2 "Playback File" sends a DMC file to the Galil motion controller. For playback operation 2 "Playback File", the input file (motion data) must have already been converted to a DMC file in send file format. Note that any DMC file that is in send file format may be played back by the [DMCPlayback](#) control.

For playback operation 3 "Pause Playback", the [DMCPlayback](#) control will stop sending contour data or linear interpolation motion segments immediately. However, those motion segments already in the Galil motion controller's internal segment buffer will continue to be processed. To resume playback that has been paused, set the [DMCPlaybackOperation](#) property to 4 "Resume Playback". Playback will resume with the next motion segment. Pause has no effect unless the [DMCPlaybackOperation](#) property is set to 2 "Playback File". Resume has no effect unless the [DMCPlaybackOperation](#) property is set to 3 "Pause Playback" and there are still motion segments to be sent to the Galil motion controller.

Errors are reported via the [DMCError](#) event.

Once a playback operation has completed (with the exception of 3 "Pause Playback"), the `DMCPlaybackOperation` property is reset to 0 by the `DMCPlayback` control.

### Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

### Data Type

Long

---

## **DMCPlaybackRate**

### Description

The rate at which contour data is sent to the Galil motion controller when playing back a DMC file (the `DMCPlaybackOperations` property is set to 2 "Playback File").

### Usage

`[form.][control.]DMCPlaybackRate[= playback rate]`

### Setting

The `DMCPlaybackRate` property settings are:

Setting Description	
0	2 ms. This is the default setting.
1	4 ms.
2	8 ms.
3	16 ms.
4	32 ms.
5	64 ms.
6	128 ms.
7	256 ms.

### Remarks

The `DMCPlaybackRate` property is used only for contour mode (the `DMCPlaybackMode` property is set to 0 "Contour"). If the playback rate is faster than the rate at which data was recorded, input data records will be aggregated to compensate for the increased rate they will be sent to the Galil motion controller.

The `DMCPlaybackRate` property must be set before setting the `DMCPlaybackOperations` property to 1 "Convert File".

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCPrefixCommands**

### **Description**

Used to add Galil language commands to the beginning of the DMC file when the input file (motion data) is converted.

### **Usage**

*[form.][control.]DMCPrefixCommands* [= *prefix commands*]

### **Setting**

The setting for the DMCPrefixCommands property may be any string up to 64K (approximately 65,500) bytes in size.

### **Remarks**

If the you wish to include more commands than can fit within the line length restriction of the model Galil motion controller you are using, the commands may be broken up into multiple lines by separating them by carriage return and line feed characters. This is easily done in Visual Basic by using the Chr function. For example:

```
DMCPrefixCommands = "Line 1 Commands" + Chr(13) + Chr(10) + "Line 2 Commands"
```

The DMCPrefixCommands property must be set before setting the [DMCPlaybackOperations](#) property to 1 "Convert File".

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

String

---

## **DMCSegmentsBeforeBeginMotion**

### **Description**

The number of linear interpolation segments to send to the Galil motion controller before sending a BGS command (begin coordinated motion sequence). The BGS command is inserted into the DMC send or download file during the conversion process (the [DMCPlaybackOperations](#) is set to 1 "Convert File").

### **Usage**

*[form.][control.]DMCSegmentsBeforeBeginMotion* [= *element*]

### **Setting**

The setting for the DMCSegmentsBeforeBeginMotion property must be an integer between 1 and 511. The default setting is 32.

### **Remarks**

The DMCSegmentsBeforeBeginMotion property is used only for linear interpolation mode (the [DMCPlaybackMode](#) property is set to 1 "Linear Interpolation").

The DMCSegmentsBeforeBeginMotion property must be set before setting the [DMCPlaybackOperations](#) property to 1 "Convert File".

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCSegmentsPerBlock**

### Description

The number of contour data or linear interpolation segments to send to the Galil motion controller without returning control back to Windows during playback (the [DMCPlaybackOperations](#) property is set to 2 "Playback File").

### Usage

*[form.][control.]DMCSegmentsPerBlock[= segments]*

### Setting

The setting for the DMCSegmentsPerBlock property must be an integer between 1 and 32,767. The default setting is 32.

### Remarks

When playing back a file, the DMCPlayback control will not return control back to Windows until the number of motion segments specified by the DMCSegmentsPerBlock property has been sent to the Galil motion controller. If the number is relatively small, e.g. 4, your application will be very responsive to mouse and keyboard actions. However, there is a high probability that the Galil motion controller will process the motion segments faster than they are being sent and motion will stop until more segments can be sent. If the number is relatively high, e.g. 1000, your application will not be very responsive to mouse and keyboard actions, but the probability that the Galil motion controller will process the motion segments faster than they are being sent is very low. The ideal number depends on the speed of the host CPU, the number of concurrent Windows applications running, and the motion profile. You may need to tune this number to fit your application.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCSuffixCommands**

### Description

Used to add Galil language commands to the end of the DMC file when the input file (motion data) is converted.

### Usage

*[form.][control.]DMCSuffixCommands[= Suffix commands]*

## Setting

The setting for the DMCSuffixCommands property may be any string up to 64K (approximately 65,500) bytes in size.

## Remarks

If the you wish to include more commands than can fit within the line length restriction of the model Galil motion controller you are using, the commands may be broken up into multiple lines by separating them by carriage return and line feed characters. This is easily done in Visual Basic by using the Chr function. For example:

```
DMCSuffixCommands = "Line 1 Commands" + Chr(13) + Chr(10) + "Line 2 Commands"
```

The DMCSuffixCommands property must be set before setting the [DMCPlaybackOperations](#) property to 1 "Convert File".

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

String

---

## DMCVectorAcceleration

### Description

The vector acceleration to use when the input file (motion data) is converted to a DMC file and the mode of motion is linear interpolation (the [DMCPlaybackMode](#) property is set to 1 "Linear Interpolation").

### Usage

*[form.][control.]*DMCVectorAcceleration[= *acceleration*]

### Setting

The setting for the DMCVectorAcceleration property must be an integer between 1024 and 68,431,360. The default value is 262,144.

### Remarks

The vector acceleration will be rounded down to the nearest factor of 1024 by the Galil motion controller. The units for the DMCVectorAcceleration property is counts per second squared.

The DMCVectorAcceleration property must be set before setting the [DMCPlaybackOperations](#) property to 1 "Convert File".

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## DMCVectorDeceleration

### Description



The vector deceleration to use when the input file (motion data) is converted to a DMC file (either send or download) and the mode of motion is linear interpolation (the [DMCPlaybackMode](#) property is set to 1 "Linear Interpolation").

## Usage

*[form.][control.]DMCVectorDeceleration[= deceleration]*

## Setting

The setting for the DMCVectorDeceleration property must be an integer between 1024 and 68,431,360. The default value is 262,144.

## Remarks

The vector deceleration will be rounded down to the nearest factor of 1024 by the Galil motion controller. The units for the DMCVectorDeceleration property is counts per second squared.

The DMCVectorDeceleration property must be set before setting the [DMCPlaybackOperations](#) property to 1 "Convert File".

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCVectorSpeed**

## Description

The vector speed to use when the input file (motion data) is converted to a DMC file (either send or download) and the mode of motion is linear interpolation (the [DMCPlaybackMode](#) property is set to 1 "Linear Interpolation").

## Usage

*[form.][control.]DMCVectorSpeed[= speed]*

## Setting

The setting for the DMCVectorSpeed property must be an integer between 0 and 12,000,000. The default value is 8192.

## Remarks

The DMCVectorSpeed property has a direct influence on how the input file (motion data) is converted to a DMC file. The DMCPlayback control will use the vector speed to determine the minimum linear interpolation segment length. The higher the vector speed, the higher the minimum linear interpolation segment length. Input file data records may be combined in order to satisfy the minimum linear interpolation segment length requirement.

The vector speed will be rounded down to the nearest factor of 2 by the Galil motion controller. The units for the DMCVectorSpeed property is counts per second.

The DMCVectorSpeed property must be set before setting the [DMCPlaybackOperations](#) property to 1 "Convert File".

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

## Galil Events

DMCError

## DMCPlayback Event Descriptions

### DMCError

#### Description

Used by the DMCPlayback control to notify the application program of an error.

#### Syntax

Sub *ctlname*\_DMCError(*ErrorNumber* As Long, *ErrorMessage* As String)

#### Remarks

Below is a list of possible errors. A complete list of error numbers can be found in the file DMCCOM40.BAS for Visual Basic programmers and the file DMCCOM.H for C/C++ programmers.

ErrorNumber	Description
-1	A time-out occurred.
-2	Galil language command error.
-3	Invalid controller number.
-4	Error opening or saving a file.
-5	Device driver failure.
-6	Invalid controller handle.
-7	Could not load required dynamic link library.
-8	Out of memory.
-9	User provided data buffer is almost full.
-10	User provided data buffer is full and data truncation has occurred.

Invalid controller handle errors most often occur because a connection to a Galil motion controller has not been established. You must use the DMCSHELL control to establish a communications session.

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCPlayback control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.
3. The Galil motion controller is waiting for a trip-point, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trip-point is cleared. This is not an error.

---

## DMCPlayback Examples

### Example: Responding to the DMCError Event

In this example, the application program reports on errors as sent by the Galil motion controller.

```
Sub DMCMove1_DMCError (ErrorNumber As Long, ErrorMessage As String)
    Dim RC As Integer
    RC = MsgBox(ErrorMessage, MB_ICONEXCLAMATION, "DMC Error")
End Sub
```

### Example: Playback Operations

This example demonstrates the various playback operations of the DMCPlayback control.

```
' Convert the motion data to a DMC send file
' Use contour mode with an 8 ms playback rate
DMCPlayback1.DMCInputFileName = "MYDATA.TXT"
DMCPlayback1.DMCOutputFileName = "MYDMC.SEN"
DMCPlayback1.DMCOutputFileFormat = 0
DMCPlayback1.DMCPlaybackMode = 0
DMCPlayback1.DMCPlaybackRate = 2
DMCPlayback1.DMCPlaybackOperation = 1

' Convert the motion data to a DMC send file
' Use linear interpolation mode with a 10,000 counts per second vector speed
DMCPlayback1.DMCInputFileName = "MYDATA.TXT"
DMCPlayback1.DMCOutputFileName = "MYDMC.SEN"
DMCPlayback1.DMCOutputFileFormat = 0
DMCPlayback1.DMCPlaybackMode = 1
DMCPlayback1.DMCVectorSpeed = 10000
DMCPlayback1.DMCPlaybackOperation = 1

' Playback the motion data
DMCPlayback1.DMCSegmentsPerBlock = 24
DMCPlayback1.DMCPlaybackOperation = 2
```

### Example: Teach and Playback

This example uses the teach tool and the playback tool together to produce a program that records the positions of the X and Y motors then plays them back at the same speed. The form load procedure connects the system to the DMC-1000:

```
Sub Form_Load ()
DMCSHELL1.DMCCConnect = True
If DMCSHELL1.DMCCConnect = False Then
    End
End If
End Sub
```

A “Record” button is created to turn off the motors and start infinite recording . This code is played behind it:  
( move the motors by hand )

```
DMCSHELL1.DMCCCommand = "MOX"
DMCSHELL1.DMCCCommand = "MOY"
dmcarray1.DMCAxisToRecord(0) = 1
dmcarray1.DMCDataType(0) = 1
dmcarray1.DMCAxisToRecord(1) = 2
dmcarray1.DMCDataType(1) = 1
dmcarray1.DMCFileName = "C:\VB\arrayout.txt"
dmcarray1.DMCCaptureDataToFile = True
dmcarray1.DMCSampleTime = 2
dmcarray1.DMCRecordData = 1
```

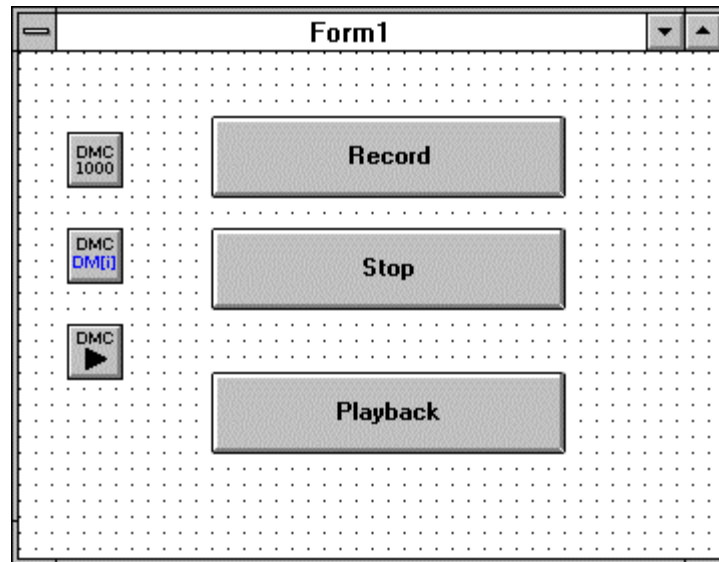
The “Stop” button stops recording and has this code under it:

```
dmcarray1.DMCRecordData = 0
DMCSHELL1.DMCCCommand = "SH"
```

Finally, the “Playback” button runs the file just created back through the motors:

```
dmcplayback1.DMCOutputFileName = "c:\vb\sendout.sen"
dmcplayback1.DMCInputFileName = "c:\vb\arrayout.txt"
dmcplayback1.DMCPlaybackRate = 2
dmcplayback1.DMCSegmentsPerBlock = 32
dmcplayback1.DMCPlaybackOperation = 1
dmcplayback1.DMCPlaybackOperation = 2
```

The entire project, with the VBXs and buttons looks like this:



**THIS PAGE LEFT BLANK INTENTIONALLY**



# Chapter 13 – DMCCompress Control

This tool can free up memory space by placing a number of Galil commands on one line.

## Galil Properties

DMCCompressedFileName  
DMCCompressOperation  
DMCController

DMCFileName  
DMCLineWidth  
DMCProgramLines

## DMCCompress Property Descriptions

### DMCCompressedFileName

#### Description

Used to set the name of the output file when using the compress operation "Compress DMC File to Another File".

#### Usage

[form.][control.]DMCCompressedFileName[= file name]

#### Setting

The setting for the DMCCompressedFileName property may be any valid DOS file name, including the path. Note that a file name is a string.

#### Remarks

The DMCCompressedFileName property must be set before setting the DMCCompressOperation property to 2 "Compress DMC File to Another File".

#### Note

This property may be used at any time. This property is saved if set at design time.

#### Data Type

String

---

### DMCCompressOperation

#### Description

Used to compress a DMC file for subsequent downloading to a Galil motion controller.

#### Usage

[form.][control.]DMCCompressOperation[= compress operation]

#### Setting

The DMCCompressOperation property settings are:

Setting	Description
0	None. This is the default setting.
1	Compress DMC File.
2	Compress DMC File to Another File.

### Remarks

Compress operation 1 "Compress DMC File" compresses the DMC file in place (the input file specified by the DMCFilename property is over-written). Compress operation 2 "Compress DMC File to Another File" compresses the DMC file to the output file specified by the DMCCompressedFileName property. Individual program lines are grouped according to the maximum line length for the Galil motion controller installed. This is determined from the value of the DMCLineWidth property. The DMCLineWidth, DMCFilename, and DMCCompressedFileName properties (for compress operation 2) must have already been set.

Errors are reported via the DMSError event.

Once a compress operation has completed the DMCCompressOperation property is reset to 0 by the DMCCompress control.

### Note

This property may not be used until runtime.

### Data Type

Long

---

## **DMCController**

### Description

The number corresponding to the Galil motion controller registered in the Windows registry database.

### Usage

[form.][[control.]]DMCController[= controller]

### Setting

The setting for the DMCController property must be an integer between 1 and 16. The default setting is 1.

### Remarks

A Galil motion controller can be registered in the Windows registry database by using the DMCREG 16 or DTERM16 programs for Windows 3.x or the DMCREG32 or DTERM32 programs for Windows 95, 98, NT, or 2000. Galil motion controllers can also be registered using the DMCTRegister control (OCX).

### Note

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

### Data Type

Long



---

## **DMCFileName**

### **Description**

Used to set the name of the DMC file which will be compressed. In the case of the compress operation "Compress DMC File", this is both the input file and output file. In the case of the compress operation "Compress DMC File to Another File", this is only the input file.

### **Usage**

[form.][[control.]]DMCFileName[= file name]

### **Setting**

The setting for the DMCFileName property may be any valid DOS file name, including the path. Note that a file name is a string.

### **Remarks**

The DMCFileName property must be set before setting the DMCCompressOperation property to 1 "Compress DMC File" or 2 "Compress DMC File to Another File".

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

String

---

## **DMCLineWidth**

### **Description**

Used to specify, in characters per line, the maximum program line width for the target Galil motion controller.

### **Usage**

[form.][[control.]]DMCLineWidth[= line width]

### **Setting**

The DMCLineWidth property settings are:

Setting	Description
0	32.
1	40. This is the default setting.
2	80.

### **Remarks**

The DMCLineWidth property must be set before compressing the file (setting the DMCCompressOperation property to 1 or 2).

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCProgramLines**

### **Description**

Used to specify the maximum number of program lines available for the target Galil motion controller.

### **Usage**

[form.][[control.]DMCProgramLines[= lines]

### **Setting**

The DMCProgramLines property setting is a positive integer. The default value is 0.

### **Remarks**

The DMCProgramLines property is an optional property which is used by the DMCCompress control to determine if the compressed file will fit into the available program space of the Galil motion controller. If the DMCProgramLines property is set to any number greater than 0, the DMCCompress control will compare that to the number of lines in the compressed file. If the number of lines in the compressed file is greater than the value of the DMCProgramLines property, the DMCCompress control will issue a warning (through the DMSError event) that the compressed file will not be able to be downloaded to the target controller. You must, of course, supply the correct number for the maximum number of program lines available in order for the DMCCompress control warning to have any significance.

The DMCProgramLines property must be set before compressing the file (setting the DMCCompressOperation property to 1 or 2).

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **Galil Events**

DMCDataAvailable  
DMSError

## **DMCCompress Event Descriptions**

### **DMSError**

#### **Description**

Used by the DMCCompress control to notify the application program of an error.

#### **Syntax**

Sub ctlname\_DMSError(ErrorNumber As Long, ErrorMessage As String)

#### **Remarks**

The two most frequent errors are:

ErrorNumber	Description
-------------	-------------

- 1 A time-out occurred.
- 2 Galil language command error

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode. For a list of all the Galil language commands and their syntax, see the Command Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

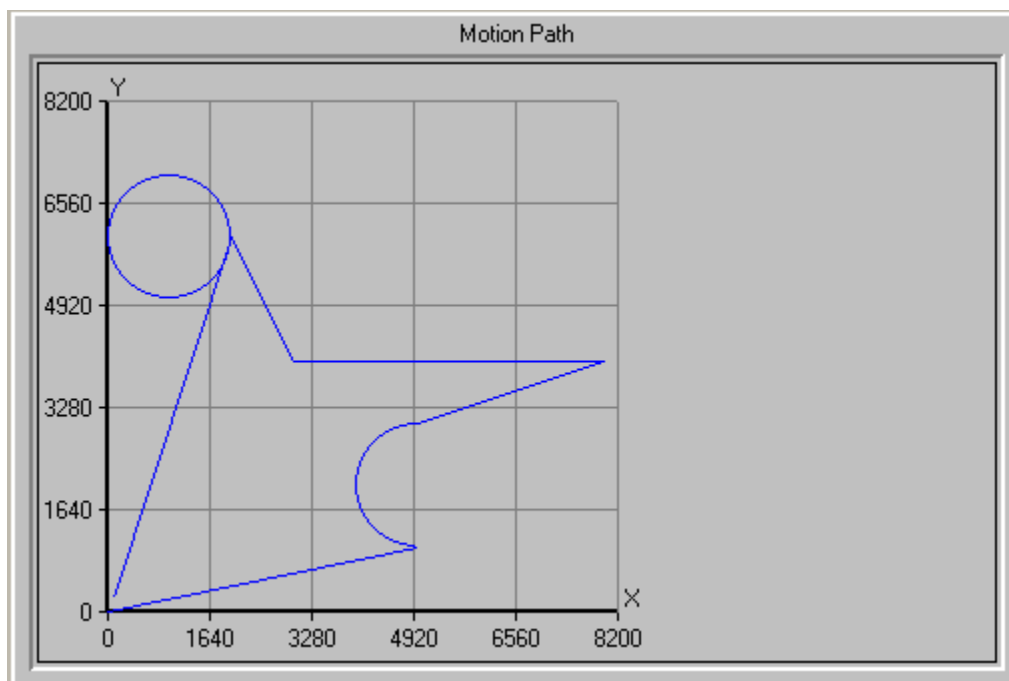
1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCCompress control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.
3. The Galil motion controller is waiting for a trip-point, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trip-point is cleared. This is not an error.

**THIS PAGE LEFT BLANK INTENTIONALLY**



## Chapter 14 – DMCMove Control

This tool displays a 2-D motion profile defined by a Galil program file. This file can contain either 'VP's, 'LI's, or 'PR's. Useful for cutting operations, this tool can display 'rapid' moves as a different color.



### Galil Properties

DMCAxisColor  
DMCAxisSelect  
DMCBevelInner  
DMCBevelOuter  
DMCBevelWidth  
DMCBorderWidth  
DMCCaptionPosition  
DMCDataColor  
DMCFileName

DMCGridColor  
DMCGridOn  
DMCHighlightColor  
DMCHighlightPrevious  
DMCHighlightPreviousColor  
DMCHighlightSegment  
DMCRapidMoveColor  
DMCValuesOn

## DMCMove Property Descriptions

### DMCAxisColor

#### Description

The color for the axis displayed on the graph.

#### Usage

[form.][[control.]]DMCAxisColor[= color]

#### Setting

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

#### Remarks

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

#### Note

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed, one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

#### Data Type

Long

---

### DMCAxisSelect

#### Description

Used to determine which two axes to display for the motion path.

#### Usage

[form.][[control.]]DMCAxisSelect[= axes]

#### Setting

The setting for the DMCAxisSelect property may be any two letter combination of the following letters: X, Y, Z, W, or A, B, C, D, E, F, G, H. The default setting is "XY".

## Remarks

For a given motion segment, if there is no movement along the designated axes as determined by the DMCAxisSelect property, nothing is displayed for that motion segment. For example, XY motion segments will not be displayed if the DMCAxisSelect property is "ZW".

## Note

This property may be used at any time.

## Data Type

String

---

## **DMCBevelInner**

### Description

Used with or without the DMCBevelOuter property to give the DMCMove control a 3D appearance.

### Usage

[form.][control.]DMCBevelInner[= setting]

### Setting

The DMCBevelInner property settings are:

Setting	Description
·	None.
·	Inset. This is the default setting.
·	Raised.

## Remarks

The DMCBevelInner property is closely related to the DMCBorderWidth property. That is, the DMCBorderWidth property determines the width of the inner bevel.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCBevelOuter**

### Description

Used with or without the DMCBevelInner property to give the DMCMove control a 3D appearance.

### Usage

[form.][control.]DMCBevelOuter[= setting]

### Setting

The DMCBevelOuter property settings are:

Setting	Description
·	None.
·	Inset.
·	Raised. This is the default setting.

**Remarks**

The DMCBevelOuter property is closely related to the DMCBevelWidth property. That is, the DMCBevelWidth property determines the width of the outer bevel.

**Note**

This property may be used at any time. This property is saved if set at design time.

**Data Type**

Long

---

**DMCBevelWidth****Description**

Used to set the width of the bevel used by the DMCBevelInner property to give the DMCMove control a 3D appearance.

**Usage**

[form.][[control.]DMCBevelWidth[= width]

**Setting**

The DMCBevelWidth property setting must be an integer between 1 and 30 inclusive.

**Remarks**

The DMCBevelWidth property is closely related to the DMCBevelOuter property. That is, the DMCBevelWidth property determines the width of the outer bevel. The DMCBevelOuter property must be set to non-zero in order for the DMCBevelWidth property to have any effect.

**Note**

This property may be used at any time. This property is saved if set at design time.

**Data Type**

Long

---

**DMCBorderWidth****Description**

Used to set the width of the bevel used by the DMCBevelInner property to give the DMCMove control a 3D appearance.

**Usage**

[form.][[control.]DMCBorderWidth[= width]

**Setting**

The DMCBorderWidth property setting must be an integer between 1 and 30 inclusive.

**Remarks**

The DMCBorderWidth property is closely related to the DMCBevelInner property. That is, the DMCBorderWidth property determines the width of the inner bevel. The DMCBevelInner property must be set to non-zero in order for the DMCBevelWidth property to have any effect.



**Note**

This property may be used at any time. This property is saved if set at design time.

**Data Type**

Long

---

**DMCCaptionPosition****Description**

Used to position the display of the caption in the DMCMove control.

**Usage**

[form.][control.]DMCCaptionPosition[= position]

**Setting**

The DMCCaptionPosition property settings are:

Setting	Description
.	None.
.	Top. This is the default setting.
.	Left.
.	Bottom.
.	Right.

**Remarks**

If the Caption property has not been set, the DMCCaptionPosition property will have no effect. To disable the display of the caption, set the DMCCaptionPosition property to 0 (none).

**Note**

This property may be used at any time. This property is saved if set at design time.

**Data Type**

Long

---

**DMCDataColor****Description**

The color for the data displayed on the graph.

**Usage**

[form.][control.]DMCDataColor[= color]

**Setting**

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

**Remarks**

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

### Note

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed, one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

---

## **DMCFileName**

### Data Type

Long

### Description

Used to set the Galil language application program file name, open the file, and display the motion path.

### Usage

[form.][control.]DMCFileName[= file name]

### Setting

The setting for the DMCFileName property may be any valid DOS file name, including the path. Note that a file name is a string.

### Remarks

Once the motion path is displayed, the file is closed.

### Note

This property may be used at any time.

### Data Type

String

---

## **DMCGridColor**

### Description

The color for the grid displayed on the graph.

### Usage

[form.][control.]DMCGridColor[= color]

### Setting

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

## Remarks

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from `CONSTANT.TXT`, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

## Note

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in `CONSTANT.TXT`. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed, one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

## Data Type

Long

---

## DMCGridOn

### Description

This enables or disables the display of the grid.

### Usage

[form.][[control.]DMCGridOn[= {True | False}]]

### Setting

The DMCGridOn property settings are:

Setting	Description
True	Displays the grid with the data.
False	No display of the grid. This is the default setting.

## Remarks

When you enable the grid, it will be displayed with the DMCGridColor.

## Note

You should set the DMCGridColor before enabling this option.

## Data Type

Integer (Boolean)

---

## DMCHighlightColor

### Description

The color for the current highlighted motion segment displayed on the graph.

## Usage

[form.][control.]DMCHighlightColor[= color]

## Setting

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

## Remarks

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

## Note

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed, one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

## Data Type

Long

---

## **DMCHighlightPrevious**

## Description

If enabled, this will highlight all the motion segments previous to (and including) the motion segment defined by the DMCHighlightSegment property.

## Usage

[form.][control.]DMCHighlightPrevious[= {True | False}]

## Setting

The DMCHighlightPrevious property settings are:

Setting	Description
True	Highlight all previous motion segments.
False	Do not highlight all previous motion segments. This is the default setting.

## Remarks

If the DMCHighlightSegment property is set to 0, the DMCHighlightPrevious property has no effect.

## Note

This property may be used at any time.

## Data Type

Integer (Boolean)

---

### **DMCHHighlightPreviousColor**

#### **Description**

The color for all the motion segments previous to the current highlighted motion segment displayed on the graph.

#### **Usage**

[form.][[control.]]DMCHHighlightPreviousColor[= color]

#### **Setting**

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

#### **Remarks**

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

#### **Note**

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed, one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

## Data Type

Long

---

### **DMCHHighlightSegment**

#### **Description**

Used to highlight the current motion segment or the current motion segment plus all segments previous to the current motion segment.

#### **Usage**

[form.][[control.]]DMCHHighlightSegment[= segment]

#### **Setting**

The DMCHHighlightSegment property setting is any positive integer including 0.

## Remarks

If the DMCHighlightSegment property is set to 0, the all highlighted segments will be removed, that is, redrawn with the color as specified by the DMCDataColor property.

## Note

This property may be used at any time.

## Data Type

Integer (Boolean)

---

## **DMCRapidMoveColor**

### Description

The color for the rapid move data displayed on the graph. Rapid move data is any line segment which follows the rapid move command (actually a remark statement) in a Galil language application program file. Include the following lines in your Galil language application program file to turn on or turn off the rapid move designation for one or more motion segments:

REM RAPID\_ON            Turns rapid move on.  
REM RAPID\_OFF          Turns rapid move off.

If you set the DMCRapidMoveColor property to the same value as the BackColor property, rapid moves will disappear. This may be useful in displaying the motion path without the rapid moves so as to get a clearer picture of the final product being worked on. You should also note that when the DMCRapidMoveColor property is the same value as the BackColor property, the DMCHighlightSegment and DMCHighlightPreviousColor properties have no effect on rapid moves.

### Usage

[form.][[control.]]DMCRapidMoveColor[= color]

### Setting

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

### Remarks

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

### Note

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed,

one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

## Data Type

Long

---

## **DMCValuesOn**

### Description

If enabled, this will display the values next to the axis.

### Usage

[form.][control.]DMCValuesOn[= {True | False}]

### Setting

The DMCValuesOn property settings are:

Setting	Description
True	Displays tick mark values on graph.
False	No display of values on graph. This is the default setting.

### Remarks

Setting this will have the values of the axis tick marks displayed on the next capture.

### Note

If you display the values, this will use space for display of the values instead of graphing the data.

## Data Type

Integer (Boolean)

---

## Galil Events

DMCError

## DMCMove Event Descriptions

### **DMCError**

### Description

Used by the DMCMove control to notify the application program of an error.

### Syntax

Sub ctlname\_DMCError(ErrorNumberAs Long, ErrorMessageAs String)

### Remarks

The two most frequent errors are:

ErrorNumber	Description
-------------	-------------

- |    |                               |
|----|-------------------------------|
| -1 | A time-out occurred.          |
| -2 | Galil language command error. |

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode. For a list of all the Galil language commands and their syntax, see the Command Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCMove control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.
3. The Galil motion controller is waiting for a trip-point, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trip-point is cleared. This is not an error.

---

## Galil Methods

Clear  
Open  
PrintDisplay

## DMCMove Method Descriptions

### DMCClear

#### Description

Used to clear the display.

#### Usage

[form.][control.]Clear

#### Remarks

Once the display of the motion path is cleared using the Clear method, a Galil language application program file must be opened using the Open method to display or redisplay the motion path.

#### Note

This method may be only be used at run time.

---

### DMCOpen



**Description**

Used to open the Galil language application program file and display the motion path.

**Usage**

RC = [form.][control.]Open

RC is a variable of data type long.

**Remarks**

The DMCFileName property must be set before using the Open method. Once the motion path is displayed, the file is closed so that it is available to other applications.

**Note**

This method may be only be used at run time.

---

**DMCPrint****Description**

Used to print the display of the motion path.

**Usage**

[form.][control.]Print

**Remarks**

A file must have been previously opened and displayed using the Open method before using the Print method.

**Note**

This method may be only be used at run time.

**THIS PAGE LEFT BLANK INTENTIONALLY**



# Chapter 15 – DMCRegister Control

Use this tool within a VB program to add, delete, or change the list of controllers within the DMC registry.

## Galil Properties and Methods

### Properties

CommPort  
CommSpeed  
Controller  
DataRecordAccess  
Delay  
EthernetRefreshRate  
Handshake  
hWnd  
Index  
IPAddress  
Model  
MsqProtocol  
NoMulticastHandle  
Protocol  
RefreshRate  
Timeout  
UseEthernetWait  
UseUnsolicitedDR  
UnsolicitedMsqHandle

### Methods

Add  
Delete  
EditRegistry  
Find  
Replace  
SelectController

## DMCRegister Property Descriptions

### CommPort

#### Description

The communications port used to communicate with the Galil serial communications motion controller.

#### Usage

[form.][control.]CommPort[= port ]

#### Setting

The CommPort property settings are:

Setting	Description	
0	COMM1:.	This is the default setting.
1	COMM2:.	
2	COMM3:.	
3	COMM4:.	
4	COMM5:.	
5	COMM6:.	
6	COMM7:.	

**Remarks**

The CommPort property must correspond to the communications port used to communicate with the Galil serial communications motion controller.

**Note**

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

**Data Type**

Long

---

**CommSpeed****Description**

The communications port used to communicate with the Galil serial communications motion controller.

**Usage**

[form.][control.]CommSpeed[= speed ]

**Setting**

The CommSpeed property settings are:

Setting	Description
0	300.
1	1200.
2	9600. This is the default setting.
3	19200.
4	38400.
5	115200.

**Remarks**

The CommSpeed property must correspond to the communications speed used to communicate with the Galil serial communications motion controller.

**Note**

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

**Data Type**

Long

---

**Controller****Description**

The number corresponding to the Galil motion controller registered in the Windows Registry database.

**Usage**

[form.][control.]Controller [= controller]

**Setting**

The setting for the Controller property must be an integer between 1 and 16. The default setting is 1.

## Remarks

Galil motion controllers are identified by their controller number. The controller number is used as the key to the Windows Registry database where configuration information is stored in the Windows environment.

## Note

This property is used as input to the Find, Replace, and Delete methods and used as output to the Add method. This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## DataRecordAccess

### Description

Used to set the method of communicating with the Galil motion controllers alternate communications channel which is provided for fast polling. This feature is currently only supported by the DMC-1600/1800 PCI bus and the DMC-1700 ISA bus motion controllers.

### Usage

[form.][[control.]]DataRecordAccess[= access]

### Setting

The DataRecordAccess property settings are:

Setting	Description
0	None. Data record access not enabled. This is the default setting.
1	DMA. Use a PC host DMA channel for communications.
2	FIFO. Use the controllers second FIFO for communications.

### Remarks

If option 1 (DMA) is selected, you must also select a DMA channel by using the DMAChannel property. Option 1 (DMA) is not available for the DMC-1600 or DMC-1800.

The speed at which the data record access communicates is affected by the RefreshRate property.

For Windows 95, 98, NT, or 2000 environments, you must use the Galil device driver in order to use the DMA option of the data record access feature. See the DeviceDriver property for more information.

### Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## Delay

### Description

The amount of delay to add between sending a command to the Galil motion controller and attempting to receive a response back.

### Usage

[form.][control.]Delay[= delay]

### Setting

The Delay property setting must be a long integer within the range 0 through 2,000,000. The default setting is 5.

### Remarks

The Delay property is a short pause (measured in microseconds) between a write to and read from the Galil motion controller. A delay may be required for host PCs with very fast CPUs. You should add a delay if all data from commands is not received, or if time-out errors (-1) occur even though the Time-out property is relatively large (1000 or more milliseconds). This most affects serial communications controllers. When adjusting the Delay property, start with a small number such as 5 or 10. In most cases, it is safe to set the Delay property to 0.

### Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **EthernetRefreshRate**

### Applies To

DMCRegister control

### Description

This property sets the refresh rate (in milliseconds) for data records when the UseUnsolicitedDR property is set to true. Valid settings are from 8 to 65535. The default value is 16 milliseconds.

### Usage

[form.][control.]EthernetRefreshRate[= Integer]

### Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

### Data Type

Integer

---

## **Handshake**

### Description

The handshake protocol used to communicate with the Galil serial communications motion controller.

### Usage

[form.][control.]Handshake[= handshake]

### Setting

The Handshake property settings are:

0	Hardware (RTS/CTS). This is the default setting.
1	Software (XOnXOff)

### Remarks

Unless you have a specially configured Galil serial communications motion controller, the Handshake property should be set to option 0 (Hardware (RTS/CTS)).

The handshake jumper must be enabled on the Galil serial communications motion controller itself.

### Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

### hWnd

---

### Index

---

### IPAddress

### Description

The IPAddress of the controller for Galil Ethernet motion controllers.

### Usage

[form.][[control.]IPAddress[=ipaddress]

### Setting

The IPAddress property setting is a string represented as a standard, four part Internet Protocol dotted address. For example, "150.29.1.100".

### Remarks

The IPAddress for a Galil Ethernet motion controllers is usually assigned by a network administrator. Once assigned, the IPAddress must be burned into the controller's flash EEPROM. This can not be done via any property setting, but it can be done using the configuration dialogs accessible through the EditRegistry method.

### Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

### Data Type

String

## **Model**

### **Description**

Used to set the controller model to match the installed Galil motion controller.

### **Usage**

[form.][control.]Model[= model]

### **Setting**

The Model property settings are:

0	Unknown. This is the default setting.
1	ModelDMC-600.
2	ModelDMC-700.
3	ModelDMC-1000.
4	ModelDMC1200
5	ModelDMC1300
6	ModelDMC1410
7	ModelDMC1411
8	ModelDMC1412
11	ModelDMC1415
12	ModelDMC1416
13	ModelDMC1417
14	ModelDMC1425
15	ModelDMC3425
16	ModelDMC1500
17	ModelDMC1600
18	ModelDMC1700
19	ModelDMC1800
20	ModelDMC1802
21	ModelDMC2000
22	ModelDMC2100
24	ModelDMC2200
25	ModelIOC7007

### **Remarks**

The model property is used by the DMCTRegister control to determine which other properties such as Address or CommSpeed will be used in configuring the controller.

### **Note**

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **MsgProtocol**

### **Applies To**

DMCTRegister control

### **Description**



When a dedicated handle is used to receive unsolicited messages from the controller, this property sets the IP protocol that is used on the handle. The default is UDP.

## Usage

[form.][control.]MsgProtocol[= Integer]

## Setting

The MsgProtocol property settings are:

Setting	Description
EthernetProtocolTCP.	Use TCP.
EthernetProtocolUDP.	Use UDP.

## Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

## Data Type

Integer

---

## **NoMulticastHandle**

### Description

Used to control whether or not to automatically open a multi-cast channel whenever starting communications with the controller. The multi-cast channel is used to broadcast Galil commands. For Galil Ethernet motion controllers only.

### Usage

[form.][control.]NoMulticastHandle[= True | False]

### Setting

The NoMulticastHandle property settings are:

Setting	Description
False.	Automatically open a multi-cast channel. This is the default setting.
True.	Do not open a multi-cast channel.

### Remarks

When you wish to broadcast a Galil command to all Ethernet controllers connected to the same network, you simply prefix the command with an exclamation point or '!'. This is same syntax that is used to send Galil commands to all controllers connected via an RS-232 daisy chain.

You may choose to not utilize the multi-cast channel feature of Galil Ethernet motion controllers in order to save an Ethernet handle, which is a limited resource on the controller. One Ethernet handle is required for each open UDP or TCP session, whether it be for client or server (master or slave) purposes.

### Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

## Data Type

Boolean

---

## **Protocol**

### Description

Used to set the Ethernet Protocol for Galil Ethernet motion controllers.

### Usage

[form.][control.]Protocol[= Protocol]

---

### Setting

The Protocol property settings are:

Setting	Description
0	TCP. This is the default setting.
1	UDP.

### Remarks

UDP provides a faster, but somewhat less reliable means of communicating to a Galil Ethernet motion controller. TCP, because of its greater reliability, is recommended for most applications.

### Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **RefreshRate**

### Description

The rate at which the data is refreshed for the data record access feature available on the Galil DMC-1600/DMC-1800 PCI bus and the DMC-1700 ISA bus motion controllers.

### Usage

[form.][control.]RefreshRate[=rate]

### Setting

The RefreshRate property settings are:

Setting	Description
0	2 ms.
1	4 ms.
2	8 ms.
3	16 ms.
4	32 ms. This is the default setting.
5	64 ms.
6	128 ms.

7	256 ms.
8	512 ms.
9	1024 ms.
10	2048 ms.
11	4096 ms.
12	8192 ms.
13	16384 ms.
14	32768 ms.

### Remarks

For Windows 95, 98, NT, or 2000 environments, you must use the Galil device driver in order to use the DMA option of the data record access feature. See the DeviceDriver property for more information.

### Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **Time-out**

### Description

The number of milliseconds to wait for a response to a command sent to the Galil motion controller.

### Usage

[form.][control.]Time-out[= time-out]

### Setting

The Time-out property setting must be a long integer within the range 1 through 8,6400,000. The default setting is 1,000.

### Remarks

The Time-out property should be set high enough so that the Galil motion controller has an adequate amount of time to respond to a given command or file operation. However, it should not be so high that any communication problem will cause excessive waiting on the part of the software.

### Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **UseEthernetWait**

### Applies To

DMCRegister control

### Description

This property enables event driven DMCCommand communications for Ethernet controllers. The default value is false.

## Usage

[form.][control.]UseEthernetWait[= True/False]

## Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

## Data Type

Boolean

---

## **UseUnsolicitedDR**

### Applies To

DMCRegister control

### Description

This property causes a dedicated data record UDP handle to be opened when a communication session is opened with an Ethernet controller. Requires newer firmware that supports the Ethernet DR command. The default value is false.

### Usage

[form.][control.]UseUnsolicitedDR[= True/False]

### Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

### Data Type

Boolean

---

## **UnsolicitedMsgHandle**

### Applies To

DMCRegister control

### Description

Used to control whether or not to open an unsolicited message channel when starting communications with the controller. The unsolicited message channel is used to notify the host PC of unsolicited messages without the need for polling the controller. For Galil Ethernet motion controllers only. The default value is NoMsgHandleChange.

### Usage

[form.][control.]UnsolicitedMsgHandle[= Integer]

### Setting

The UnsolicitedMsgHandle property settings are:

#### Setting

#### Description

NoMsgHandleChange.	Do not open an unsolicited response channel. Leave any existing unsolicited message handle settings unchanged.
UsePrimaryCommunicationHandle.	Send unsolicited messages on the primary communications handle.
UseDedicatedHandle	Send unsolicited messages on a dedicated communications handle.

## Remarks

If you are using either the DMCSHELL or DMCTerminal controls to receive unsolicited messages via the DMCResponse event, there are no special steps necessary to make use of the unsolicited message channel feature. Use the DMCSHELL and DMCTerminal controls as you would for any other type of Galil motion controller when processing unsolicited messages.

You may choose to not utilize the unsolicited message channel feature of Galil Ethernet motion controllers in order to save an Ethernet handle, which is a limited resource on the controller. One Ethernet handle is required for each open UDP or TCP session, whether it be for client or server (master or slave) purposes.

## Note

This property is used as input to the Add and Replace methods and used as output to the Find method. This property may be used at any time. This property is saved if set at design time.

## Data Type

Integer

# DMCRegister Method Descriptions

## Add

### Description

Used to add a Galil motion controller to the Windows Registry database.

### Usage

RC = [form.][control.]Add

RC is a variable of data type long. Zero indicates success. If RC is non-zero, the method has encountered an error. Error codes are documented in the file DMCCOM40.BAS.

## Remarks

If the Add method is successful, the controller number which has been added to the Windows Registry database is returned in the Controller property.

## Note

This method may only be used at run time.

## Delete

### Description

Used to delete a Galil motion controller from the Windows Registry database.

### Usage

RC = [form.][control.]Delete

RC is a variable of data type long. Zero indicates success. If RC is non-zero, the method has encountered an error. Error codes are documented in the file DMCCOM40.BAS.

### **Remarks**

The Controller property must be set prior to invoking this method..

### **Note**

This method may only be used at run time.

---

## **EditRegistry**

### **Description**

Used to edit Galil motion controllers in the Windows registry database. A dialog window is presented to the user which allows them to add, replace, and delete controllers to/from the Windows registry database.

### **Usage**

[form.][control.]EditRegistry

### **Description**

Used to edit Galil motion controllers in the Windows registry database. A dialog window is presented to the user which allows them to add, replace, and delete controllers to/from the Windows registry database.

### **Usage**

[form.][control.]EditRegistry

---

## **Find**

### **Description**

Used to find a Galil motion controller in the Windows Registry database and query its configuration properties..

### **Usage**

RC = [form.][control.]Find

RC is a variable of data type long. Zero indicates success. If RC is non-zero, the method has encountered an error. Error codes are documented in the file DMCCOM40.BAS.

### Remarks

The Controller property must be set prior to invoking this method. If the return code from the Find method is zero, the Galil motion controller was found in the Windows Registry database and the relevant DMCTRegister control properties will have been refreshed.

### Note

This method may only be used at run time.

---

## **RegisterPNPControllers**

### Description

Used to find all installed plug-and-play enabled Galil motion controllers and register them in the Windows Registry database. This method will also update the configuration for those Galil motion controllers already registered.

### Usage

Count = [form.][control.]RegisterPnPControllers

Count is a variable of data type long. Count indicates how many Galil motion controllers were found and registered.

### Remarks

The RegisterPnPControllers method does nothing under Windows 95. Galil motion controllers which are plug-and-play enabled are automatically registered under Windows 95, and all configuration changes (either done manually through the Windows 95 Control Panel or done automatically by Windows 95) are reflected in the Windows Registry database.

### Note

This method may only be used at run time.

---

## **Replace**

### Description

Used to replace a Galil motion controller in the Windows Registry database.

### Usage

RC = [form.][control.]Replace

RC is a variable of data type long. Zero indicates success. If RC is non-zero, the method has encountered an error. Error codes are documented in the file DMCCOM40.BAS.

### Remarks

The Controller property must be set prior to invoking this method.

### Note

This method may only be used at run time.

---

## **SelectController**

### Description

Used to select a Galil motion controller from the Windows Registry database. A dialog window is presented to the user which allows them to select a controller.

## Usage

Controller = [form.][control.]SelectController

Controller is a variable of data type integer. If the user selected a controller from the Select Controller dialog, the SelectController method will return a positive integer. If the user did not select a controller, a value of -1 is returned.

## Remarks

If there are no controllers registered in the Windows Registry database, the Select Controller dialog will still show, but the list box containing all the registered controllers will be empty.

## Note

This method may only be used at run time.

---

## DMCRegister Examples

This program adds a new controller to the Galil registry:

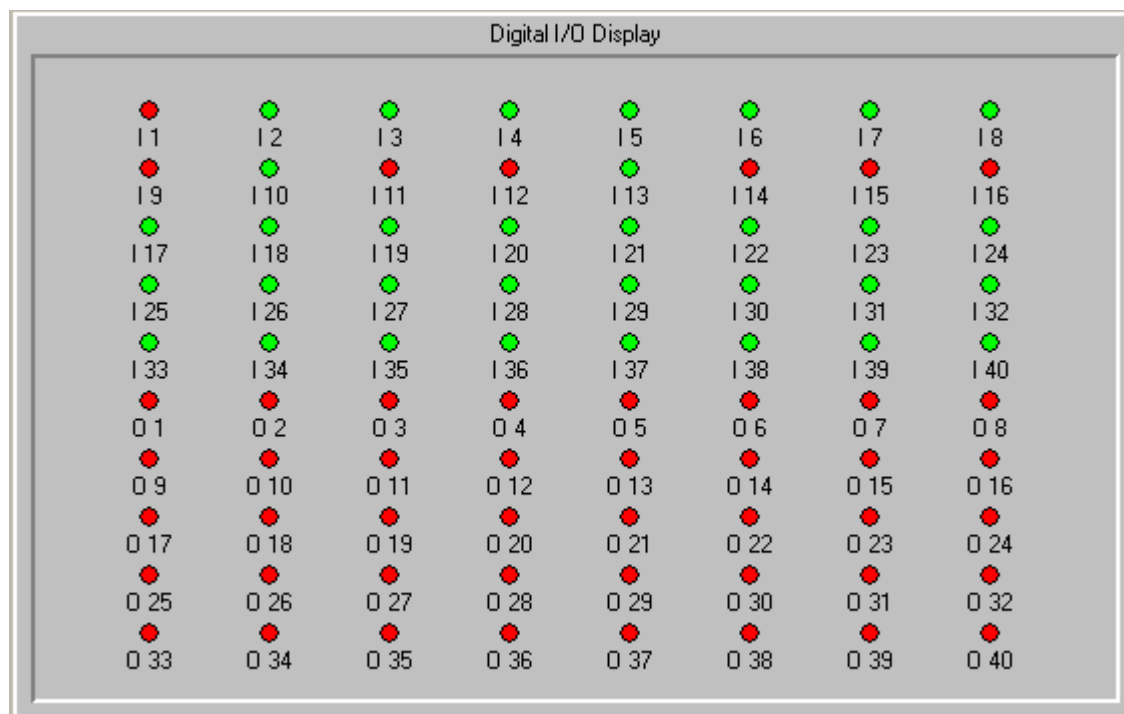
```
' Attempt to find controller 1 in the Windows registry
DMCRegister1.Controller = 1
RC = DMCRegister1.Find
' If RC = 0, controller 1 was found in the Windows registry
If RC <> 0 Then
    ' Add controller 1 to the Windows registry
    DMCRegister1.Model = 3          ' DMC-1000
    DMCRegister1.Address = 1000    ' Standard I/O port address
    DMCRegister1.Interrupt = 0     ' No interrupts
    RC = DMCRegister1.Add
    ' If RC = 0, the controller has been added
    If RC <> 0 Then
        MsgBox "Could not register controller. Program Terminating."
    End
End If
End If
```





# Chapter 16 – DMCI/O Control

Used to display the current status of the general inputs and outputs on the controller in a graphical format. This display is configurable for up to 80 inputs and 80 outputs.



## Galil Properties

DMCBevelInner  
DMCBevelOuter  
DMCBevelWidth  
DMCBorderWidth  
DMCCaptionPosition  
DMCController  
DMCHiColor  
DMCIgnoreError  
DMCLabelPosition  
DMCLabels  
DMCLoColor  
DMCOrientation  
DMCPollController  
DMCPollInterval  
DMCShowInputBank0  
DMCShowInputBank1  
DMCShowInputBank2  
DMCShowInputBank3

DMCShowInputBank4  
DMCShowInputBank5  
DMCShowInputBank6  
DMCShowInputBank7  
DMCShowInputBank8  
DMCShowInputBank9  
DMCShowOutputBank0  
DMCShowOutputBank1  
DMCShowOutputBank2  
DMCShowOutputBank3  
DMCShowOutputBank4  
DMCShowOutputBank5  
DMCShowOutputBank6  
DMCShowOutputBank7  
DMCShowOutputBank8  
DMCShowOutputBank9  
DMCUserDefinedInputLabel  
DMCUserDefinedOutputLabel

## DMCIO Property Descriptions

### DMCBevelInner

#### Description

Used with or without the DMCBevelOuter property to give the DMCIO control a 3D appearance.

#### Usage

*[form.][control.]DMCBevelInner*[= *setting*]

#### Setting

The DMCBevelInner property settings are:

Setting Description	
0	None.
1	Inset. This is the default setting.
2	Raised.

#### Remarks

If either the DMCBevelInner or DMCBevelOuter properties are set to any value other than 0 (none), the ForeColor and BackColor properties will have no effect. If a 3D appearance is not desired or if you wish to change either the foreground or background colors of the DMCIO control, both the DMCBevelInner or DMCBevelOuter properties should be set to 0 (none). This property may be used with or without the BorderStyle property.

#### Note

This property may be used at any time. This property is saved if set at design time.

#### Data Type

Long

---

### DMCBevelOuter

#### Description

Used with or without the DMCBevelInner property to give the DMCIO control a 3D appearance.

#### Usage

*[form.][control.]DMCBevelOuter*[= *setting*]

#### Setting

The DMCBevelOuter property settings are:

Setting Description	
0	None.
1	Inset.
2	Raised. This is the default setting.

#### Remarks

If either the DMCBevelInner or DMCBevelOuter properties are set to any value other than 0 (none), the ForeColor and BackColor properties will have no effect. If a 3D appearance is not desired or if you wish to change either the foreground or background colors of the DMCIO control, both the DMCBevelInner or DMCBevelOuter properties should be set to 0 (none). This property may be used with or without the BorderStyle property.

#### Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

### **DMCCaptionPosition**

#### **Description**

Used to position the display of the caption in the DMCIO control.

#### **Usage**

[*form.*][*control.*]**DMCCaptionPosition**[= *position*]

#### **Setting**

The DMCCaptionPosition property settings are:

Setting	Description
0	None.
1	Top. This is the default setting.
2	Left.
3	Bottom.
4	Right.

#### **Remarks**

If the Caption property has not been set, the DMCCaptionPosition property will have no effect. To disable the display of the caption, set the DMCCaptionPosition property to 0 (none).

#### **Note**

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

### **DMCController**

#### **Description**

The number corresponding to the Galil motion controller registered in the Windows registry database.

#### **Usage**

[*form.*][*control.*]**DMCController**[= *controller*]

#### **Setting**

The setting for the DMCController property must be an integer between 1 and 16. The default setting is 1.

#### **Remarks**

A Galil motion controller can be registered in the Windows registry database by using the DMCREG program for Windows 3.x or the DMCREG32 program for Windows 95, 98, NT, or 2000.

#### **Note**

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

## Data Type

Long

---

### **DMCHiColor**

#### **Description**

The color to use for the I/O LEDs when their state is "high" or "on".

## Usage

[form.][control.]**DMCHiColor**[= color]

## Setting

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

## Remarks

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

## Note

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed, one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

## Data Type

Long

---

## **DMCIgnoreError**

### Description

Used to control the notification of errors.

### Usage

[form.][control.]**DMCIgnoreError**[= {True | False}]

### Setting

The DMCIgnoreError property settings are:

Setting Description	
True	Ignore errors.
False	Do not ignore errors. This is the default setting.

### Remarks

If the DMCIgnoreError property is set to False and an error occurs, the application is notified through the event and the property will be set to False, effectively shutting down communications between the DMCIO control and the Galil motion controller.

If the DMCIgnoreError property is set to True, the DMCIO control will not notify the application of any errors which may occur. Likewise, polling is not suspended on recognition of an error.

This property should be used with some caution as important information regarding an error condition may not be sent to the application program. The DMCIgnoreError property is best used when the application uses many DMCIO or related controls and it is possible or even likely that the DMCIO control may experience a

communications time-out while waiting for another control to complete its activity with the Galil motion controller.

### Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

### Data Type

Integer (Boolean)

---

## **DMCLabelPosition**

### Description

Used to position the display of the labels in the DMCIO control.

### Usage

[*form.*][*control.*]DMCLabelPosition[= *position*]

### Setting

The DMCLabelPosition property settings are:

Setting	Description
0	Bottom. This is the default setting.
1	Top.
2	Right.
3	Left.

### Remarks

It is up to the developer to allow enough space for labels, especially if the labels are user-defined. If the DMCLabelPosition property is set to Bottom or Top, the labels will be centered along the horizontal axis with respect to the I/O LED. If the DMCLabelPosition property is set to Right or Left, the labels will be centered along the vertical axis with respect to the I/O LED.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCLabels**

### Description

Used to determine what, if any, labels to use for the display of the individual I/O LEDs.

### Usage

[*form.*][*control.*]DMCLabels[= *label*]

### Setting

The DMCLabels property settings are:

Setting	Description
0	None. This is the default setting.
1	Default.
2	User Defined.

### Remarks

The default labels simply identify the I/O LED as an input or output using "I" or "O", and its number. An example would be "I 7" or "O 14".

It is up to the developer to allow enough space for labels, especially if the labels are user-defined. If the DMCLabelPosition property is set to Bottom or Top, the labels will be centered along the horizontal axis with respect to the I/O LED. ). If the DMCLabelPosition property is set to Right or Left, the labels will be centered along the vertical axis with respect to the I/O LED.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCLoColor**

### Description

The color to use for the I/O LEDs when their state is "low" or "off".

### Usage

`[form.][control.]DMCLoColor[= color]`

### Setting

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

### Remarks

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

### Note

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed, one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

### Data Type

Long

---

## **DMCOrientation**

### **Description**

Used to arrange the display of the individual I/O LEDs in a horizontal or vertical orientation.

### **Usage**

[form.][control.]**DMCOrientation**[= orientation]

### **Setting**

The DMCOrientation property settings are:

<b>Setting Description</b>	
0	Horizontal. This is the default setting.
1	Vertical.

### **Remarks**

None.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCPollController**

### **Description**

Used to start or stop polling the Galil motion controller.

### **Usage**

[form.][control.]**DMCPollController**[= {True | False}]

### **Setting**

The DMCPollController property settings are:

<b>Setting Description</b>	
True	Begins polling the Galil motion controller.
False	Ends polling the Galil motion controller. This is the default setting.

### **Remarks**

When the DMCPollController property is set to True, the DMCI/O control will poll the Galil motion controller for the status of the selected I/O banks every n milliseconds where n is equal to the setting for the DMCPollInterval property.

### **Note**

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

### **Data Type**

Integer (Boolean)

---

## **DMCPollInterval**

### **Description**

The number of milliseconds between submitting Galil language commands to the Galil motion controller.

### **Usage**

[form.][control.]**DMCPollInterval**[= interval]

### Setting

The DMCPollInterval property setting must be an integer within the range 50 through 10,000. The default setting is 500.

### Remarks

The DMCPollInterval property should be set to the highest value as is practical for the application. The faster the polling (the lower the setting), the more resources consumed.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCShowInputBank0**

### Description

Used to show or hide the display of input bank 0 which is inputs 1 through 8.

### Usage

[form.][control.]DMCShowInputBank0 [= {True | False}]

### Setting

The DMCShowInputBank0 property settings are:

Setting Description	
---------------------	--

True	Display input bank 0. This is the default setting.
False	Do no display input bank 0.

### Remarks

Input bank 0 (inputs 1 through 8) is standard on all Galil motion controllers supported by this control.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Integer (Boolean)

---

## **DMCShowInputBank1**

### Description

Used to show or hide the display of input bank 1 which is inputs 9 through 16.

### Usage

[form.][control.]DMCShowInputBank1 [= {True | False}]

### Setting

The DMCShowInputBank1 property settings are:

Setting Description	
---------------------	--

True	Display input bank 1. This is the default setting.
False	Do no display input bank 1.

### Remarks

Input bank 1 (inputs 9 through 16) is standard on some Galil motion controller models supported by this control. Other models require an optional I/O expansion module. See the documentation on your Galil motion controller for more details.

### Note

This property may be used at any time. This property is saved if set at design time.



**Data Type**

Integer (Boolean)

**DMCShowInputBank2****Description**

Used to show or hide the display of input bank 2 which is inputs 17 through 24.

**Usage**

[form.][control.]DMCShowInputBank2 [= {True | False}]

**DMCShowInputBank3****Description**

Used to show or hide the display of input bank 3 which is inputs 25 through 32.

**Usage**

[form.][control.]DMCShowInputBank3 [= {True | False}]

**DMCShowInputBank4****Description**

Used to show or hide the display of input bank 4 which is inputs 33 through 40.

**Usage**

[form.][control.]DMCShowInputBank4 [= {True | False}]

**DMCShowInputBank5****Description**

Used to show or hide the display of input bank 5 which is inputs 41 through 48.

**Usage**

[form.][control.]DMCShowInputBank5 [= {True | False}]

**DMCShowInputBank6****Description**

Used to show or hide the display of input bank 6 which is inputs 49 through 56.

**Usage**

[form.][control.]DMCShowInputBank6 [= {True | False}]

**DMCShowInputBank7****Description**

Used to show or hide the display of input bank 7 which is inputs 57 through 64.

**Usage**

[form.][control.]DMCShowInputBank7 [= {True | False}]

**DMCShowInputBank8****Description**

Used to show or hide the display of input bank 8 which is inputs 65 through 72.

### **Usage**

[form.][control.]DMCShowInputBank8[= {True | False}]

### **DMCShowInputBank9**

#### **Description**

Used to show or hide the display of input bank 9 which is inputs 73 through 80.

### **Usage**

[form.][control.]DMCShowInputBank9[= {True | False}]

### **DMCShowOutputBank0**

#### **Description**

Used to show or hide the display of output bank 0 which is outputs 1 through 8.

### **Usage**

[form.][control.]DMCShowOutputBank0[= {True | False}]

### **DMCShowOutputBank1**

#### **Description**

Used to show or hide the display of output bank 1 which is outputs 9 through 16.

### **Usage**

[form.][control.]DMCShowOutputBank1[= {True | False}]

### **DMCShowOutputBank2**

#### **Description**

Used to show or hide the display of output bank 2 which is outputs 17 through 24.

### **Usage**

[form.][control.]DMCShowOutputBank2[= {True | False}]

### **DMCShowOutputBank3**

#### **Description**

Used to show or hide the display of output bank 3 which is outputs 25 through 32.

### **Usage**

[form.][control.]DMCShowOutputBank3[= {True | False}]

### **DMCShowOutputBank4**

#### **Description**

Used to show or hide the display of output bank 4 which is outputs 33 through 40.

### **Usage**

[form.][control.]DMCShowOutputBank4[= {True | False}]

### **DMCShowOutputBank5**

#### **Description**

Used to show or hide the display of output bank 5 which is outputs 41 through 48.

### Usage

[form.][control.]DMCShowOutputBank5 [= {True | False}]

### **DMCShowOutputBank6**

### Description

Used to show or hide the display of output bank 6 which is outputs 49 through 56.

### Usage

[form.][control.]DMCShowOutputBank6 [= {True | False}]

### **DMCShowOutputBank7**

### Description

Used to show or hide the display of output bank 7 which is outputs 57 through 64.

### Usage

[form.][control.]DMCShowOutputBank7 [= {True | False}]

### **DMCShowOutputBank8**

### Description

Used to show or hide the display of output bank 8 which is outputs 65 through 72.

### Usage

[form.][control.]DMCShowOutputBank8 [= {True | False}]

### **DMCShowOutputBank9**

### Description

Used to show or hide the display of output bank 9 which is outputs 73 through 80.

### Usage

[form.][control.]DMCShowOutputBank9 [= {True | False}]

---

## **DMCUserDefinedInputLabel**

### Description

Used to assign user labels to individual input LEDs.

### Usage

[form.][control.]DMCUserDefinedInputLabel(index) [= label]

---

## **DMCUserDefinedOutputLabel**

### Description

Used to assign user labels to individual output LEDs.

### Usage

[form.][control.]DMCUserDefinedOutputLabel(index) [= label]

### Setting

The setting for the DMCUserDefinedOutputLabel property may be a string of any length.

### Remarks

If you wish to use user defined labels, a label must be assigned to each I/O included in the display. Any I/O which does not have a label assigned will show no label. There is one label array for inputs and one label array for outputs. The index for the DMCUserDefinedOutputLabel property is the I/O number minus one because the I/Os are counted starting from one and the index is counted starting from zero. The DMCLabel property must be set to 2 or "User Defined" in order for this property to have any effect. It is up to the developer to allow enough space for labels.

## Note

This property may be used at run-time only.

## Data Type

Array of Strings

---

## Galil Events

[DMCError](#)

[DMCInputChanged](#)

[DMCOutputChanged](#)

## DMCIO Event Descriptions

### DMCError

#### Description

Used by the DMCIO control to notify the application program of an error.

#### Syntax

Sub *ctlname*\_DMCError(*ErrorNumber*As Long, *ErrorMessage*As String)

#### Remarks

The two most frequent errors are:

ErrorNumber	Description
-1	A time-out occurred.
-2	Galil language command error.

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCIO control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.

3. The Galil motion controller is waiting for a trippoint, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trippoint is cleared. This is not an error.

---

### **DMCInputChanged**

#### **Description**

Used by the DMCIO control to notify the application program that at least one input on the Galil motion controller changed state. That is, at least one input changed from high to low or low to high.

#### **Syntax**

Sub *ctlname*\_DMCInputChanged()

#### **Remarks**

It is up to the application program to determine which input or inputs changed state.

---

### **DMCOutputChanged**

#### **Description**

Used by the DMCIO control to notify the application program that at least one output on the Galil motion controller changed state. That is, at least one output changed from high to low or low to high.

#### **Syntax**

Sub *ctlname*\_DMCOutputChanged()

#### **Remarks**

It is up to the application program to determine which output or outputs changed state.

**THIS PAGE LEFT BLANK INTENTIONALLY**



# Chapter 17 – DMCVector Control

DMCVector is used to process a 2-D Galil vector file and reduce the vector speed at corners and around arcs. Tool offset paths can also be created.

## Galil Properties

DMCApplyCornerSlowDown

DMCApplyToolOffset

DMCCornerSlowDownMode

DMCInputFileName

DMCMaxCentrifugalAcceleration

DMCMaxCornerAngle

DMCMinArcSpeed

DMCMinCornerAngle

DMCMinCornerSpeed

DMCOutputFileName

DMCToolOffsetError

DMCToolOffsetSide

DMCToolRadius

DMCVectorAcceleration

DMCVectorDeceleration

DMCVectorSpeed

## DMCVector Property Descriptions

### DMCApplyCornerSlowDown

#### Description

Used by the DMCVector control to determine whether or not to apply corner slow down to the input vector file.

#### Usage

[form.][[control.]DMCApplyCornerSlowDown[= {True | False}]]

#### Remarks

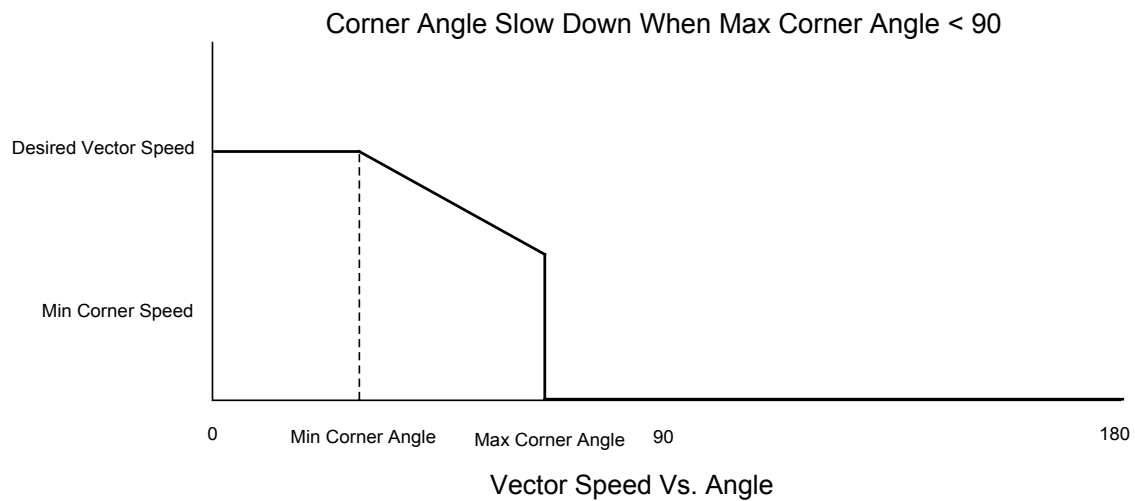
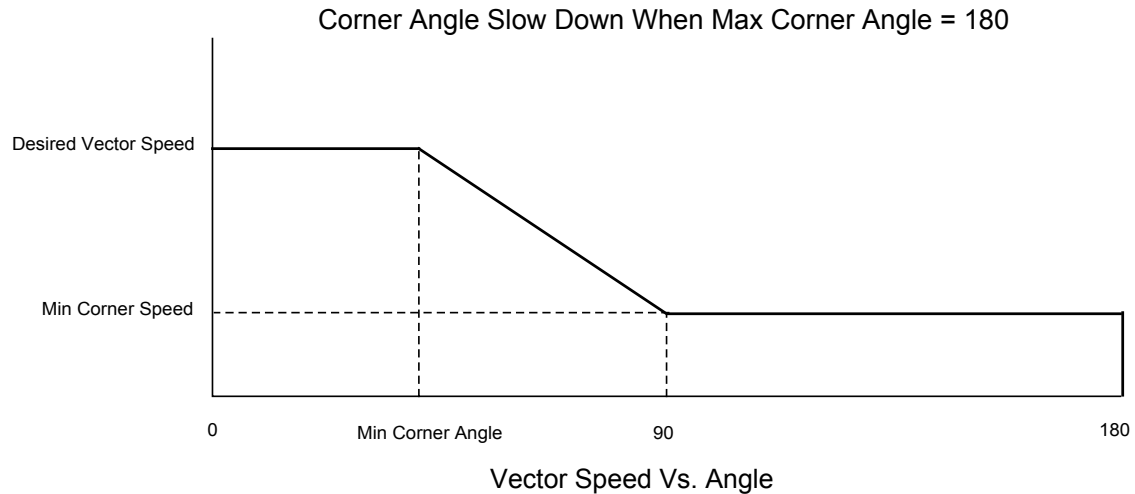
The corner slow down algorithm, applied to the input vector file when the [ConvertFile](#) method is invoked, will either automatically lower the speed on motion segments in the output vector file or force the motion to come to a complete stop when the angle difference between two consecutive motion segments is greater than the minimum corner angle or the arc radius is smaller than the minimum arc radius.

The [DMCCornerSlowDownMode](#) property determines whether the resulting vector motion will be continuous or not. That is, if the [DMCCornerSlowDownMode](#) property is set to 1 or "Stop", a complete stop will be inserted into the output vector file instead of a change in vector speed. A complete stop will also be inserted into the output vector file if the angle difference between two consecutive motion segments is greater than the [DMCMaxCornerAngle](#) property.

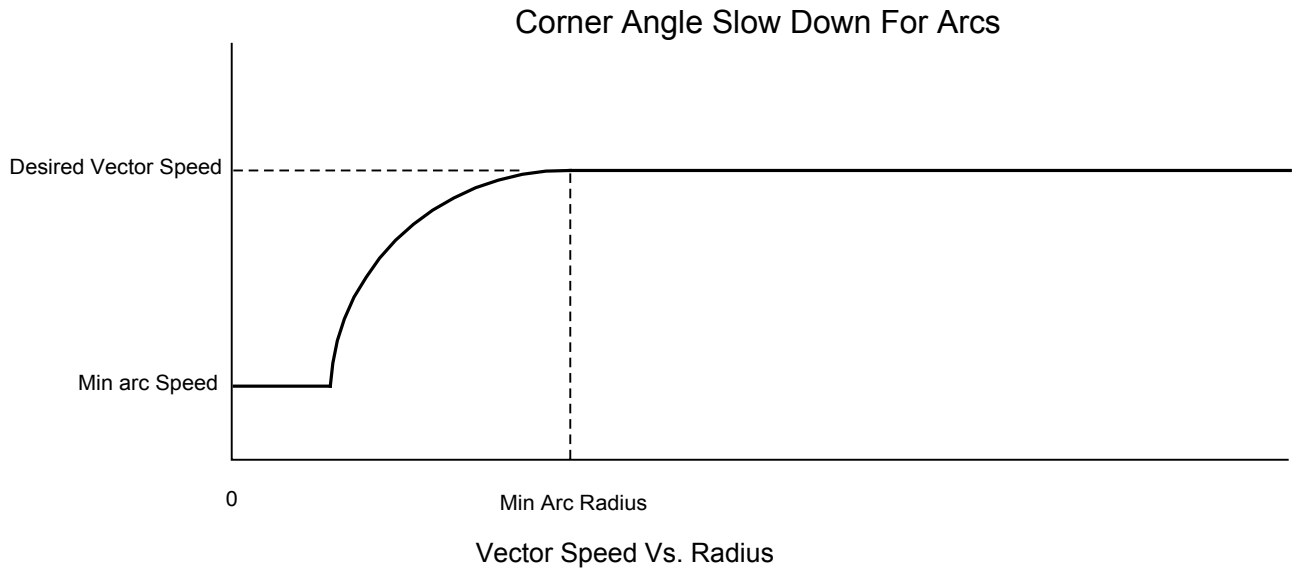
The minimum corner angle is taken from the [DMCMinCornerAngle](#) property. The minimum arc radius is calculated from the vector speed (see the [DMCVectorSpeed](#) property) and the maximum centrifugal acceleration (see the [DMCMaxCentrifugalAcceleration](#) property). The factor by which the speed of a given

motion segment is lowered depends on the [DMCMinCornerSpeed](#), [DMCMinArcSpeed](#), [DMCVectorAcceleration](#), and [DMCVectorDeceleration](#) properties.

The following graphs show how the various corner slow down properties affect the vector speed for a given sequence of motion segments.







If both the `DMCApplYCornerSlowDown` and `DMCApplYToolOffset` properties are set to True, the corner slow down is applied before the tool offset.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Boolean

---

## **DMCApplYToolOffset**

### Description

Used by the `DMCVector` control to determine whether or not to apply tool offset to the input vector file.

### Usage

`[form.][control.]DMCApplYToolOffset [= {True | False}]`

### Remarks

Applying tool offset to the input vector file will result in a new motion path where some of the motion segments may have been trimmed or spanned. The `DMCToolRadius`, `DMCToolOffsetSide`, and `DMCToolOffsetError` properties control how the tool offset is applied.

Special remark statements inserted into the input vector file can change the tool offset `DMCToolOffsetSide` property dynamically. These special remark statements are:

```
REM TOOL_OFFSET_LEFT    DMCToolOffsetSide property is changed to left.
REM TOOL_OFFSET_RIGHT   DMCToolOffsetSide property is changed to right.
```

You should allow at least one motion segment in between switching the `DMCToolOffsetSide` property to allow the tool offset algorithm to properly transition the tool offset. Tool offset can be dynamically turned on or off by using the following special remark statements:

```
REM TOOL_OFFSET_ON      Tool offset is turned on.
```

REM TOOL\_OFFSET\_OFF      Tool offset is turned off.

A lead-in and a lead-out motion segment is also expected when tool offset is turned on or off.

If both the [DMCApplCornerSlowDown](#) and [DMCApplToolOffset](#) properties are set to True, the corner slow down is applied before the tool offset.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Boolean

---

## **DMCCornerSlowDownMode**

### Description

Used to set the corner slow down to either continuous or stop mode.

### Usage

*[form.]**[control.]***DMCCornerSlowDownMode***[= mode]*

### Setting

The DMCCornerSlowDownMode property settings are:

Setting	Description
---------	-------------

0	Continuous. This is the default setting.
1	Stop.

### Remarks

When the DMCCornerSlowDownMode property is set to 0 or "Continuous", the DMCVector control will lower the vector speed according to the acuteness of the angle difference between two consecutive motion segments or the radius of an arc segment. When the DMCCornerSlowDownMode property is set to 1 or "Stop", the DMCVector control will force the motion to come to a complete stop under the same conditions.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCInputFileName**

### Description

Used to set the name of the input vector file (in DMC format) which contains the motion path to be converted, that is, to have corner slow down and/or tool offset applied.

### Usage

*[form.]**[control.]***DMCInputFileName***[= input file name]*

### Setting

The setting for the DMCInputFileName property may be any valid DOS file name, including the path. Note that a file name is a string.

### Remarks

The DMCInputFileName property must be set before invoking the [ConvertFile](#) method.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

String

---

## **DMCMaxCentrifugalAcceleration**

### Description

In conjunction with the [DMCVectorSpeed](#) property, used to calculate the minimum arc radius used by the corner slow down algorithm.

### Usage

[*form.*][*control.*]**DMCMaxCentrifugalAcceleration**[= *acceleration*]

### Setting

The setting for the DMCMaxCentrifugalAcceleration property must be an integer between 1 and 100,000,000. The default value is 100,000. The units for the DMCMaxCentrifugalAcceleration property are encoder counts/second<sup>2</sup>.

### Remarks

The DMCMaxCentrifugalAcceleration property is defined as the vector speed<sup>2</sup>/radius.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCMaxCornerAngle**

### Description

The angle difference between two consecutive motion segments which causes the corner slow down algorithm to automatically transition from continuous mode to stop mode.

### Usage

[*form.*][*control.*]**DMCMaxCornerAngle**[= *angle*]

### Setting

The setting for the DMCMaxCornerAngle property must be an integer between 1 and 180. The default value is 180. The units for the DMCMaxCornerAngle property are degrees.

### Remarks

When the DMCMaXCornerAngle has been exceeded, the corner slow down algorithm will insert a complete stop into the output vector file.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCMinArcSpeed**

### Description

The minimum or slowest vector speed to be applied by the corner slow down algorithm in case the arc radius of a given motion segment is less than the minimum arc radius.

### Usage

[form.][control.]DMCMinArcSpeed[= speed]

### Setting

The setting for the DMCMinArcSpeed property must be an integer between 1 and 8,000,000. The default value is 1,000. The units for the DMCMinArcSpeed property are encoder counts/second.

### Remarks

The DMCMinArcSpeed property is not used if the [DMCCornerSlowDownMode](#) property is set to 1 or "Stop" mode.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCMinCornerAngle**

### Description

The angle difference between two consecutive motion segments which causes the corner slow down algorithm to either lower the vector speed or stop the motion completely depending on the value of the [DMCCornerSlowDownMode](#) property.

### Usage

[form.][control.]DMCMinCornerAngle[= angle]

### Setting

The setting for the DMCMinCornerAngle property must be an integer between 1 and 180. The default value is 30. The units for the DMCMinCornerAngle property are degrees.

### Remarks

When the DMCMinCornerAngle has been exceeded, the corner slow down algorithm will insert a change in vector speed into the output vector file in the case of the [DMCCornerSlowDownMode](#) property being set to 0

or "Continuous" mode, or a complete stop into the output vector file in the case of the [DMCCornerSlowDownMode](#) property being set to 1 or "Stop" mode.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCMinCornerSpeed**

### Description

The minimum or slowest vector speed to be applied by the corner slow down algorithm in case the angle difference between two consecutive motion segments is greater than the minimum corner angle.

### Usage

*[form.][[control.]DMCMinCornerSpeed[= speed]*

### Setting

The setting for the DMCMinCornerSpeed property must be an integer between 1 and 8,000,000. The default value is 1,000. The units for the DMCMinCornerSpeed property are encoder counts/second.

### Remarks

The DMCMinCornerSpeed property is not used if the [DMCCornerSlowDownMode](#) property is set to 1 or "Stop" mode.

### Note

This property may be used at any time. This property is saved if set at design time.

### Data Type

Long

---

## **DMCOutputFileName**

### Description

Used to set the name of the output vector file (in DMC format) which contains the new motion path after the input vector file (also in DMC format) has been converted, that is, had corner slow down and/or tool offset applied.

### Usage

*[form.][[control.]DMCOutputFileName[= output file name]*

### Setting

The setting for the DMCOutputFileName property may be any valid DOS file name, including the path. Note that a file name is a string.

### Remarks

The DMCOutputFileName property must be set before invoking the [ConvertFile](#) method.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

String

---

## **DMCToolOffsetError**

### Description

The action to take in case the tool offset algorithm detects an error. The most likely error to occur is the intersection of one or more motion segments.

### Usage

*[form.]**[control.]*DMCToolOffsetError[= error]

### Setting

The DMCToolOffsetError property settings are:

Setting	Description
0	Notify. This is the default setting.
1	Auto Fix.
2	Ignore.

### Remarks

The "Notify" setting means that if an error is detected, the conversion process will be halted immediately and the [ConvertFile](#) method will return an error. The "Auto Fix" setting means that if the tool offset algorithm detects a case where one or more motion segments would intersect as a result of applying the tool offset, it will automatically remove those segments which cause the intersection. The "Ignore" setting means that all tool offset errors will be ignored. This would be the recommended setting if you know that the motion path has intersecting motion segments and this is acceptable. For example, this might be the case when a cutting tool can be turned on or off by using an I/O port.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCOffsetSide**

### Description

The side on which the tool offset will be applied.

### Usage

*[form.]**[control.]*DMCToolOffsetSide[= side]

### Setting

The DMCOffsetSide property settings are:

Setting	Description
0	Left. This is the default setting.
1	Right.

## Remarks

Special remark statements inserted into the input vector file can change the tool offset DMCToolOffsetSide property dynamically. These special remark statements are:

REM TOOL\_OFFSET\_LEFT     DMCToolOffsetSide property is changed to left.  
REM TOOL\_OFFSET\_RIGHT    DMCToolOffsetSide property is changed to right.

You should allow at least one motion segment in between switching the DMCToolOffsetSide property to allow the tool offset algorithm to properly transition the tool offset. Tool offset can be dynamically turned on or off by using the following special remark statements:

REM TOOL\_OFFSET\_ON        Tool offset is turned on.  
REM TOOL\_OFFSET\_OFF       Tool offset is turned off.

A lead-in and a lead-out motion segment is also expected when tool offset is turned on or off.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCToolRadius**

### Description

The tool radius for tool offset.

### Usage

[form.][[control.]DMCToolRadius[= radius]

### Setting

The setting for the DMCToolRadius property must be an integer between 1 and 1,000,000. The default value is 100. The units for the DMCToolRadius property are encoder counts.

## Remarks

The DMCToolRadius property may not be changed dynamically.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCVectorAcceleration**

### Description

The vector acceleration to use when the input vector file is converted to the output vector file.

## Usage

[form.][control.]**DMCVectorAcceleration**[= acceleration]

## Setting

The setting for the DMCAcceleration property must be an integer between 1,024 and 68,431,360. The default value is 262,144.

## Remarks

The vector acceleration will be rounded down to the nearest factor of 1,024 by the Galil motion controller. The units for the DMCVectorAcceleration property are encoder counts/second<sup>2</sup>.

The DMCVectorAcceleration property will be used as the default vector speed for the output vector file. However, vector speed will be overridden by any VA command which appears in the input vector file.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCVectorDeceleration**

## Description

The vector deceleration to use when the input vector file is converted to the output vector file.

## Usage

[form.][control.]**DMCVectorDeceleration**[= deceleration]

## Setting

The setting for the DMCVectorDeceleration property must be an integer between 1,024 and 68,431,360. The default value is 262,144.

## Remarks

The vector deceleration will be rounded down to the nearest factor of 1,024 by the Galil motion controller. The units for the DMCVectorDeceleration property are encoder counts/second<sup>2</sup>.

The DMCVectorDeceleration property will be used as the default vector speed for the output vector file. However, vector speed will be overridden by any VD command which appears in the input vector file.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCVectorSpeed**

## Description



The vector speed to use when the input vector file is converted to the output vector file.

## Usage

[*form.*][*control.*]**DMCVectorSpeed**[= *speed*]

## Setting

The setting for the DMCVectorSpeed property must be an integer between 0 and 12,000,000. The default value is 8,192.

## Remarks

The vector speed will be rounded down to the nearest factor of 2 by the Galil motion controller. The units for the DMCVectorSpeed property are encoder counts/second.

The DMCVectorSpeed property will be used as the default vector speed for the output vector file. However, vector speed will be overridden by any VS command which appears in the input vector file. In addition, any speed override commands which are placed with a motion segment by using the < character will override the vector speed.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## Galil Events

DMCError

## DMCVector Event Descriptions

### DMCError

## Description

Used by the DMCVector control to notify the application program of an error.

## Syntax

Sub *ctlname*\_DMCError(*ErrorNumber* As Long, *ErrorMessage* As String)

## Remarks

Below is a list of possible errors. A complete list of error numbers can be found in the file DMCCOM40.BAS for Visual Basic programmers and the file DMCCOM.H for C/C++ programmers.

### ErrorNumber

-1  
-2  
-3  
-4  
-5  
-6  
-7

### Description

A time-out occurred.  
Galil language command error.  
Invalid controller number.  
Error opening or saving a file.  
Device driver failure.  
Invalid controller handle.  
Could not load required dynamic link library.

- 8 Out of memory.
- 9 User provided data buffer is almost full.
- 10 User provided data buffer is full and data truncation has occurred.

Invalid controller handle errors most often occur because a connection to a Galil motion controller has not been established. You must use the DMCShell control to establish a communications session.

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCVector control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.
3. The Galil motion controller is waiting for a trip-point, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trip-point is cleared. This is not an error.

## Galil Methods

ConvertFile

## DMCVector Method Descriptions

### ConvertFile

#### Description

Used to apply corner slow down or tool offset to the input vector file creating the output vector file.

#### Usage

RC = [*form.*][*control.*]**ConvertFile**  
 RC is a variable of data type long.

## Remarks

The output vector file (see the [DMCOutputFileName](#) property) is created or re-created each time the ConvertFile method is invoked. Errors are returned through the return code and the [DMCError](#) event. A return code of zero signifies that the output vector file was created successfully.

## Note

This method may only be used at run time.

---

## DMCVector Examples

### Example: Responding to the DMCError Event

In this example, the application program reports on errors as sent by the Galil motion controller.

```
Sub DMCMove1_DMCError (ErrorNumber As Long, ErrorMessage As String)
    Dim RC As Integer
    RC = MsgBox(ErrorMessage, MB_ICONEXCLAMATION, "DMC Error")
End Sub
```

### Example: Using the ConvertFile Method

This example demonstrates the ConvertFile method of the DMCVector control.

```
Dim RC As Long

' Apply corner slow down
'   Note: any of the properties below
'   can be set at design time
DMCVector1.DMCApplyCornerSlowDown = True
DMCVector1.DMCApplyToolOffset = False
DMCVector1.DMCInputFileName = "INPUT.DMC"
DMCVector1.DMCOutputFileName = "OUTPUT.SEN"
DMCVector1.DMCMinCornerSpeed = 100
DMCVector1.DMCMinArcSpeed = 100
DMCVector1.DMCVectorSpeed = 10000
DMCVector1.DMCVectorAcceleration = 200000
DMCVector1.DMCVectorDeceleration = 200000
DMCVector1.DMCMaxCentrifugalAcceleration = 1000000
DMCVector1.DMCMinCornerAngle = 25
DMCVector1.DMCMaxCornerAngle = 100

RC = DMCVector1.ConvertFile

' Apply corner slow down and tool offset
'   Note: any of the properties below
'   can be set at design time
DMCVector1.DMCApplyCornerSlowDown = True
DMCVector1.DMCApplyToolOffset = True
DMCVector1.DMCInputFileName = "INPUT.DMC"
DMCVector1.DMCOutputFileName = "OUTPUT.SEN"
DMCVector1.DMCMinCornerSpeed = 100
DMCVector1.DMCMinArcSpeed = 100
DMCVector1.DMCVectorSpeed = 10000
DMCVector1.DMCVectorAcceleration = 200000
```

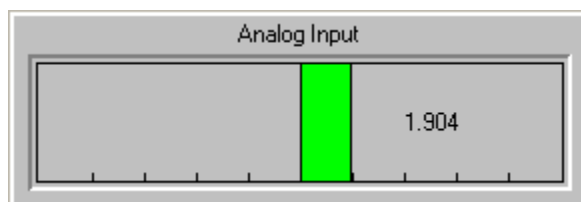
```
DMCVector1.DMCVectorDeceleration = 200000  
DMCVector1.DMCMaxCentrifugalAcceleration = 1000000  
DMCVector1.DMCMinCornerAngle = 25  
DMCVector1.DMCMaxCornerAngle = 100  
DMCVector1.DMCToolRadius = 150  
DMCVector1.DMCToolOffsetSide = 1
```

```
RC = DMCVector1.ConvertFile
```



# Chapter 18 – DMCAnalog Control

This tool is used to display the voltage value on any of the analog inputs of the controller. Values are displayed as a bar graph with optional ranges and colors.



## Galil Properties

DMCAnalogInput  
DMCBevelInner  
DMCBevelOuter  
DMCBevelWidth  
DMCBorderWidth  
DMCCaptionPosition  
DMCController  
DMCIgnoreError  
DMCNegativeColor

DMCOrientation  
DMCPollController  
DMCPollInterval  
DMCPositiveColor  
DMCRange  
DMCShowTickMarks  
DMCShowValue  
DMCStyle  
DMCUseSystemTimer

## DMCAnalog Property Descriptions

### DMCAnalog

#### Description

The analog input to display (graph).

#### Usage

*[form.][control.]DMCAnalogInput[= analog input]*

#### Setting

The DMCAnalogInput property setting is an integer between 1 and 32. The default value is 1.

#### Remarks

Most Galil motion controllers have as standard eight analog inputs numbered from 1 to 8.

#### Note

This property may be used at any time. This property is saved if set at design time.

#### Data Type

Long

---

## **DMCBevelInner**

### **Description**

Used with or without the DMCBevelOuter property to give the DMCAnalog control a 3D appearance.

### **Usage**

[*form.*][*control.*]DMCBevelInner[= *setting*]

### **Setting**

The DMCBevelInner property settings are:

<b>Setting Description</b>	
0	None.
1	Inset. This is the default setting.
2	Raised.

### **Remarks**

The DMCBevelInner property is closely related to the [DMCBorderWidth](#) property. That is, the [DMCBorderWidth](#) property determines the width of the inner bevel.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCBevelOuter**

### **Description**

Used with or without the DMCBevelInner property to give the DMCAnalog control a 3D appearance.

### **Usage**

[*form.*][*control.*]DMCBevelOuter[= *setting*]

### **Setting**

The DMCBevelOuter property settings are:

<b>Setting Description</b>	
0	None.
1	Inset.
2	Raised. This is the default setting.

### **Remarks**

The DMCBevelOuter property is closely related to the [DMCBevelWidth](#) property. That is, the [DMCBevelWidth](#) property determines the width of the outer bevel.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCBevelWidth**

### **Description**

Used to set the width of the bevel used by the DMCBevelInner property to give the DMCAnalog control a 3D appearance.

### **Usage**

*[form.][control.]DMCBevelWidth*[= *width*]

### **Setting**

The DMCBevelWidth property setting must be an integer between 1 and 30 inclusive.

### **Remarks**

The DMCBevelWidth property is closely related to the [DMCBevelOuter](#) property. That is, the DMCBevelWidth property determines the width of the outer bevel. The [DMCBevelOuter](#) property must be set to non-zero in order for the DMCBevelWidth property to have any effect.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCBorderWidth**

### **Description**

Used to set the width of the bevel used by the DMCBevelInner property to give the DMCAnalog control a 3D appearance.

### **Usage**

*[form.][control.]DMCBorderWidth*[= *width*]

### **Setting**

The DMCBorderWidth property setting must be an integer between 1 and 30 inclusive.

### **Remarks**

The DMCBorderWidth property is closely related to the [DMCBevelInner](#) property. That is, the DMCBorderWidth property determines the width of the inner bevel. The [DMCBevelInner](#) property must be set to non-zero in order for the DMCBevelWidth property to have any effect.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCCaptionPosition**

### **Description**

Used to position the display of the caption in the DMCAAnalog control.

### **Usage**

*[form.]**[control.]***DMCCaptionPosition***[= position]*

### **Setting**

The DMCCaptionPosition property settings are:

<b>Setting Description</b>	
0	None.
1	Top. This is the default setting.
2	Left.
3	Bottom.
4	Right.

### **Remarks**

If the Caption property has not been set, the DMCCaptionPosition property will have no effect. To disable the display of the caption, set the DMCCaptionPosition property to 0 (none).

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCController**

### **Description**

The number corresponding to the Galil motion controller registered in the Windows registry database.

### **Usage**

*[form.]**[control.]***DMCController***[= controller]*

### **Setting**

The setting for the DMCController property must be an integer between 1 and 16. The default setting is 1.

### **Remarks**

A Galil motion controller can be registered in the Windows registry database by using the DMCREG 16 or DTERM16 programs for Windows 3.x or the DMCREG32 or DTERM32 programs for Windows 95, 98, NT, or 2000. Galil motion controllers can also be registered using the DMCRegister control (OCX).

### **Note**

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.



## Data Type

Long

---

## **DMCIgnoreError**

### Description

Used to control the notification of errors.

### Usage

[form.][control.]DMCIgnoreError[= {True | False}]

### Setting

The DMCIgnoreError property settings are:

Setting Description	
True	Ignore errors.
False	Do not ignore errors. This is the default setting.

### Remarks

If the DMCIgnoreError property is set to False and an error occurs, the application is notified through the [DMCError](#) event and the [DMCPollController](#) property will be set to False, effectively shutting down communications between the DMCAnalog control and the Galil motion controller.

If the DMCIgnoreError property is set to True, the DMCAnalog control will not notify the application of any errors which may occur. Likewise, polling is not suspended on recognition of an error.

This property should be used with some caution as important information regarding an error condition may not be sent to the application program. The DMCIgnoreError property is best used when the application uses many DMCAnalog or related controls and it is possible or even likely that the DMCAnalog control may experience a communications time-out while waiting for another control to complete its activity with the Galil motion controller.

### Note

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

## Data Type

Integer (Boolean)

---

## **DMCNegativeColor**

### Description

The color to use when the value of the analog input is less than zero. This property is used only when the [DMCStyle](#) property is set to 0 or "Split Bar".

### Usage

[form.][control.]DMCNegativeColor[= color]

## Setting

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

## Remarks

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

## Note

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed, one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

## Data Type

Long

---

## **DMCOrientation**

## Description

Used to display the graph of the analog input in a horizontal or vertical orientation.

## Usage

*[form.]**[control.]***DMCOrientation***[= orientation]*

## Setting

The DMCOrientation property settings are:

Setting Description	
0	Horizontal. This is the default setting.
1	Vertical.

## Remarks

None.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCPollController**

### **Description**

Used to start or stop polling the Galil motion controller.

### **Usage**

[*form.*][*control.*]**DMCPollController**[= {True | False}]

### **Setting**

The DMCPollController property settings are:

<b>Setting Description</b>	
True	Begins polling the Galil motion controller.
False	Ends polling the Galil motion controller. This is the default setting.

### **Remarks**

When the DMCPollController property is set to True, the DMCAAnalog control will poll the Galil motion controller for the value of the selected analog input every n milliseconds where n is equal to the setting for the [DMCPollInterval](#) property.

### **Note**

This property may not be referenced at runtime until a connection to the Galil motion controller has been established.

### **Data Type**

Integer (Boolean)

---

## **DMCPollInterval**

### **Description**

The number of milliseconds between submitting Galil language commands to the Galil motion controller.

### **Usage**

[*form.*][*control.*]**DMCPollInterval**[= *interval*]

### **Setting**

The DMCPollInterval property setting must be an integer within the range 50 through 10,000. The default setting is 500.

### **Remarks**

The DMCPollInterval property should be set to the highest value as is practical for the application. The faster the polling (the lower the setting), the more resources consumed.

### **Note**

This property may be used at any time. This property is saved if set at design time.

### **Data Type**

Long

---

## **DMCPositiveColor**

### **Description**

The color to use for the graph when value of the analog input is greater than zero. This is also the color of the graph when the [DMCStyle](#) property is set to 1 or "Continuous Bar".

### **Usage**

[*form.*][*control.*]**DMCPositiveColor**[= *color*]

### **Setting**

This must be a valid color. This can be chosen by the inspector, or by using one of the standard color definitions.

### **Remarks**

Visual Basic uses the Microsoft Windows environment RGB scheme for colors. Each property has the following ranges of settings:

Normal RGB colors: Colors specified by using the Color palette, or by using the RGB or QBColor functions in code.

System default colors: Colors specified with system color constants from CONSTANT.TXT, a Visual Basic file that specifies system defaults. The Windows environment substitutes the user's choices as specified in the user's Control Panel settings.

### **Note**

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower three bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte is not 0, Visual Basic uses the system colors, as defined in the user's Control Panel and enumerated in CONSTANT.TXT. To display text in the Windows environment, both the text and background colors must be solid. If the text or background colors you've selected are not displayed, one of the selected colors may be dithered that is, comprised of up to three different-colored pixels. If you choose a dithered color for either the text or background, the nearest solid color will be substituted.

### **Data Type**

Long

---

## **DMCRange**

### **Description**

Used to determine which range of values to use for the graph.

### **Usage**

[*form.*][*control.*]**DMCRange**[= *label*]

### **Setting**

The DMCRange property settings are:

<b>Setting Description</b>	
0	-10 to +10. This is the default setting.
1	0 to +10.

**Remarks**

The DMCRange property is used only when the [DMCStyle](#) property is set to 1 or “Continuous Bar”.

**Note**

This property may be used at any time. This property is saved if set at design time.

**Data Type**

Long

---

**DMCShowTickMarks**
**Description**

Used to show or hide the display of tick marks on the graph.

**Usage**

*[form.]**[control.]*DMCShowTickMarks [= {True | False}]

**Setting**

The DMCShowTickMarks property settings are:

<b>Setting Description</b>	
True	Display tick marks. This is the default setting.
False	Do not display tick marks.

**Remarks**

Each tick mark is approximately 2 volts.

**Note**

This property may be used at any time. This property is saved if set at design time.

**Data Type**

Integer (Boolean)

---

**DMCShowValue**
**Description**

Used to show or hide the display the analog input value on the graph.

**Usage**

*[form.]**[control.]*DMCShowValue [= {True | False}]

**Setting**

The DMCShowValue property settings are:

<b>Setting Description</b>	
True	Display value. This is the default setting.
False	Do not display value.

## Remarks

The value displayed has a resolution of three decimal places.

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

## **DMCStyle**

### Description

Used to determine which style of bar graph to use to display the analog input value.

### Usage

[*form.*][*control.*]**DMCStyle**[= *style*]

### Setting

The DMCStyle property settings are:

Setting	Description
0	Split Bar. This is the default setting.
1	Continuous Bar.

## Remarks

The “Split Bar” graph shows .

## Note

This property may be used at any time. This property is saved if set at design time.

## Data Type

Long

---

## **DMCUseSystemTimer Property**

### Applies To

DMCPoll control property under the DMCArray control

### Description

Used to determine which type of timer to use for polling.

### Usage

[*form.*][*control.*]**DMCUseSystemTimer**[= {True | False}]

### Setting

The DMCUseSystemTimer property settings are:

Setting	Description
True	Use a high-resolution system timer for polling.
False	Use a standard windows timer for polling. This is the default setting.

## Remarks

The DMCPoll control by default uses a standard Windows timer to poll the Galil motion controller. The standard Windows timer is not very accurate, but it does not consume much in the way of resources. As an option, the DMCPoll control can use a high-resolution system timer to poll the controller. The high-resolution timer is accurate to within +/- 5ms and will allow you use much faster poll intervals. The trade-off is that this type of timer consumes much more resources. It is not recommended that more than four DMCPoll controls be used simultaneously with the DMCUseSystemTimer property set to True. Most Windows versions will allow up to eight.

## Note

This property may only be used when the DMCPollController property is set to False. This property is saved if set at design time.

## Data Type

Integer (Boolean)

## Galil Events

[DMCError](#)

[DMCValueChanged](#)

## DMCAnalog Event Descriptions

### DMCError

#### Description

Used by the DMCAnalog control to notify the application program of an error.

#### Syntax

Sub *ctlname*\_**DMCError**(*ErrorNumber* As Long, *ErrorMessage* As String)

## Remarks

Below is a list of possible errors. A complete list of error numbers can be found in the file DMCCOM40.BAS for Visual Basic programmers and the file DMCCOM.H for C/C++ programmers.

ErrorNumber	Description
-1	A time-out occurred.
-2	Galil language command error.
-3	Invalid controller number.
-4	Error opening or saving a file.
-5	Device driver failure.
-6	Invalid controller handle.
-7	Could not load required dynamic link library.
-8	Out of memory.
-9	User provided data buffer is almost full.
-10	User provided data buffer is full and data truncation has occurred.

Invalid controller handle errors most often occur because a connection to a Galil motion controller has not been established. You must use the DMCSHELL control to establish a communications session.

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands which are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCAAnalog control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.
3. The Galil motion controller is waiting for a trip-point, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trip-point is cleared. This is not an error.



---

## **DMCValueChanged**

### **Description**

Used by the DMCAnalog control to notify the application program that the value of the analog input has changed. The analog input is defined by the [DMCAnalogInput](#) property.

### **Syntax**

Sub *ctlname*\_**DMCValueChanged**(ByVal *Value* As Single)

### **Remarks**

The Value parameter will be a real number between -10 and +10.

---

## **DMCAnalog Examples**

### **Example: Responding to the DMSError Event**

In this example, the application program reports on errors as sent by the Galil motion controller.

```
Sub DMCAnalog1_DMSError (ErrorNumber As Long, ErrorMessage As String)
    Dim RC As Integer
    RC = MsgBox(ErrorMessage, MB_ICONEXCLAMATION, "DMC Error")
End Sub
```



# Chapter 19 – DMCDiagnosticTests Control

This tool runs a number of diagnostic tests on an axis. The results of these test are displayed in a box. Includes an automatic tuning method.

## Galil Properties

DMCController

DMCModel

DMCIncludeVersionTest

DMCIncludeParameterTest

DMCIncludeStopTest

DMCIncludeMotionTest

DMCIncludeStabilityTest

DMCIncludeTuning

## DMCDiagnosticTests Property Descriptions

### DMCController

#### Description

The number corresponding to the Galil motion controller registered in the Windows registry database.

#### Usage

[form.][[control.]DMCController[= controller]

#### Setting

The setting for the DMCController property must be an integer between 1 and 16. The default setting is 1.

#### Remarks

A Galil motion controller can be registered in the Windows registry database by using the DMCREG 16 or DTERM16 programs for Windows 3.x or the DMCREG32 or DTERM32 program s for Windows 95, 98, NT, or 2000. Galil motion controllers can also be registered using the DMCRegister control (OCX).

#### Note

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

#### Data Type

Long

---

### DMCModel

#### Description

Used to configure the DMCDiagnosticTests control to match the installed Galil motion controller. The exact Galil motion controller must be specifed in order for the Version Test to yield correct results.

## Usage

[form.][[control.]DMCModel[= {model}]]

## Setting

The DMCModel property settings are:

Setting	Description
0	Unknown. This is the default setting.
1	DMC-1010.
2	DMC-1020.
3	DMC-1030.
4	DMC-1040.
5	DMC-1050.
6	DMC-1060.
7	DMC-1070.
8	DMC-1080.
9	DMC-1410.
10	DMC-1411.
11	DMC-1412.
12	DMC-1510.
13	DMC-1520.
14	DMC-1530.
15	DMC-1540.
16	DMC-1550.
17	DMC-1560.
18	DMC-1570.
19	DMC-1580.
20	DMC-1710.
21	DMC-1720.
22	DMC-1730.
23	DMC-1740.
24	DMC-1750.
25	DMC-1760.
26	DMC-1770.
27	DMC-1780.
28	DMC-1210.
29	DMC-1220.
30	DMC-1230.
31	DMC-1240.
32	DMC-1250.
33	DMC-1260.
34	DMC-1270.
35	DMC-1280.
36	DMC-1610.
37	DMC-1620.
38	DMC-1630.
39	DMC-1640.
40	DMC-1650.
41	DMC-1660.
42	DMC-1670.
43	DMC-1680.
44	DMC-1810.
45	DMC-1820.
46	DMC-1830.
47	DMC-1840.

48	DMC-1850.
49	DMC-1860.
50	DMC-1870.
51	DMC-1880.
52	DMC-2010.
53	DMC-2020.
54	DMC-2030.
55	DMC-2040.
56	DMC-2050.
57	DMC-2060.
58	DMC-2070.
59	DMC-2080.
60	DMC-1812.
61	DMC-1822.
62	DMC-1832.
63	DMC-1842.
64	DMC-2110.
65	DMC-2120.
66	DMC-2130.
67	DMC-2140.
68	DMC-2150.
69	DMC-2160.
70	DMC-2170.
71	DMC-2180.
72	DMC-1310.
73	DMC-1320.
74	DMC-1330.
75	DMC-1340.
76	DMC-1415.

## Remarks

The DMCModel property is used as a parameter in the Version Test.

## Note

This property must be set before attempting to connect to a Galil motion controller. This property is saved if set at design time.

## Data Type

Long

---

## **DMCIncludeVersionTest**

## Description

Used to include or exclude the Version Test.

## Usage

[form.][control.]DMCIncludeVersionTest[= {True | False}]

## Setting

The DMCIncludeVersionTest property settings are:

Setting	Description
False	Do not include test.

True                      Include Test. This is the Default setting.

## Remarks

The Version Test uses the value from the DMCMModel property to test if the Galil motion controller is the correct model. The Version Test will also display the firmware revision.

## Note

This property may be used only at design time. It is an error to set this property at run-time. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

## **DMCIncludeParameterTest**

### Description

Used to include or exclude the Parameter Test.

### Usage

[form.][control.]**DMCIncludeParameterTest**[= {True | False}]

### Setting

The DMCIncludeParameterTest property settings are:

Setting	Description
False	Do not include test.
True	Include Test. This is the Default setting.

## Remarks

The Parameter Test checks the value of several key variables against a table of expected values and reports any discrepancies. The variables checked are KP, KD, KI, TL, OF, MT, CE, and TM.

## Note

This property may be used only at design time. It is an error to set this property at run-time. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

## **DMCIncludeStopTest**

### Description

Used to include or exclude the Stop Test.

### Usage

[form.][control.]**DMCIncludeStopTest**[= {True | False}]

### Setting

The DMCIncludeStopTest property settings are:

Setting	Description
False	Do not include test.
True	Include Test. This is the Default setting.

## Remarks

The Stop Test tests the servo system for drift. The Stop Test sets the PID filter to 0 and checks if there is any observed motion within the limits of the specified parameters. Observations are reported back to the user.

## Note

This property may be used only at design time. It is an error to set this property at run-time. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

### **DMCIncludeMotionTest**

#### **Description**

Used to include or exclude the Motion Test.

#### **Usage**

[form.][control.]**DMCIncludeMotionTest**[= {True | False}]

#### **Setting**

The DMCIncludeMotionTest property settings are:

Setting	Description
False	Do not include test.
True	Include Test. This is the Default setting.

## Remarks

The Motion Test tests the servo system for expected motion. The Motion Test applies a specified voltage for a specified time interval to the controller and checks if the resulting motion (if any) is within the limits of the specified parameters. Observations are reported back to the user.

Warning!: This test will attempt to move your motors.

## Note

This property may be used only at design time. It is an error to set this property at run-time. This property is saved if set at design time.

## Data Type

Integer (Boolean)

---

### **DMCIncludeStabilityTest**

#### **Description**

Used to include or exclude the Stability Test.

#### **Usage**

[form.][control.]**DMCIncludeStabilityTest**[= {True | False}]

#### **Setting**

The DMCIncludeStabilityTest property settings are:

Setting	Description
---------	-------------

False	Do not include test.
True	Include Test. This is the Default setting.

### Remarks

The Stability Test tests the servo system for stability. Observations are reported back to the user.

Warning!: This test will attempt to move your motors.

### Note

This property may be used only at design time. It is an error to set this property at run-time. This property is saved if set at design time.

### Data Type

Integer (Boolean)

## **DMCIncludeTuning**

### Description

Used to include or exclude Tuning.

### Usage

[form.][control.]**DMCIncludeTuning**[= {True | False}]

### Setting

The DMCIncludeTuning property settings are:

Setting	Description
False	Do not include test.
True	Include Test. This is the Default setting.

### Remarks

The Tuning function uses the same methodology as in the Galil Servo Design Kit auto-tuning section.

Warning!: This test will attempt to move your motors.

### Note

This property may be used only at design time. It is an error to set this property at run-time. This property is saved if set at design time.

### Data Type

Integer (Boolean)

## Galil Events

DMCError

## DMCDiagnosticTests Event Descriptions

### **DMCError**

## Description

Used by the DMCDiagnosticTests control to notify the application program of an error.

## Syntax

Sub ctlname\_ **DMCError**(ErrorNumber As Long, ErrorMessage As String)

## Remarks

Below is a list of possible errors. A complete list of error numbers can be found in the file DMCCOM40.BAS for Visual Basic programmers and the file DMCCOM.H for C/C++ programmers.

ErrorNumber	Description
-1	A time-out occurred.
-2	Galil language command error.
-3	Invalid controller number.
-4	Error opening or saving a file.
-5	Device driver failure.
-6	Invalid controller handle.
-7	Could not load required dynamic link library.
-8	Out of memory.
-9	User provided data buffer is almost full.
-10	User provided data buffer is full and data truncation has occurred.

Invalid controller handle errors most often occur because a connection to a Galil motion controller has not been established. You must use the DMCSHELL control to establish a communications session.

Galil language command errors occur for three reasons:

1. Unrecognized Galil language command. For example, it was intended to send the Galil language command "BG" to the Galil motion controller, but "BH" (a non-existent Galil language command) was sent to the Galil motion controller instead.
2. Incorrect syntax. One or more arguments are missing or incorrect. Most Galil language commands have "reasonable" defaults, but some do not.
3. The Galil language command is not valid in the current context. For example, Galil language commands that are valid for independent jogging mode are not valid while the Galil motion controller is currently in linear interpolation mode.

For a list of all the Galil language commands and their syntax, see the Technical Reference Guide for the Galil motion controller installed.

Time-outs can occur for three reasons, only one of which is actually an error:

1. Communication to the Galil motion controller has not been established or has been interrupted. This is an error.
2. A Galil language program is currently executing in the Galil motion controller and is using the "IN" command to solicit input from an operator. When the operator responds to the request for input, there is no acknowledgment from the Galil motion controller. Thus, the DMCDiagnosticTests control may detect a time-out condition because the Galil motion controller has not responded to a "command" within the time-out period. This is not an error.
3. The Galil motion controller is waiting for a trip-point, such as AMX (after motion X-axis) or WT1000 (wait 1000 milliseconds) to clear. Communication is suspended until the trip-point is cleared. This is not an error.



---

## Galil Methods

PrintReport

### DMCPrintReport Method Descriptions

#### **PrintReport**

##### **Description**

Used to print the current state of the diagnostic tests performed for all axes.

##### **Usage**

[form.][control.]PrintReport

##### **Remarks**

The PrintReport method will only print data for the tests that have been performed for the axes they have performed on.

##### **Note**

This method may only be used at run time.

---

## Test Parameters

### **DMCDiagnosticTests Control Test Parameters**

Each of the diagnostic tests has a set of parameters that may be customized by the developer. However, the developer should not alter the default value of a parameter except where explicitly noted. Except for the DMCModel property that is a parameter for the Version Test, all parameters must be customized for each axis. Parameters may be modified using the custom property page dialogs available in Visual Basic design mode. Click on the property named (Custom) in the Visual Basic Properties window and then on the ellipsis (...) to activate the custom property pages. The parameters are saved in the file DMCTEST.DAT when the Save button is clicked.

Parameters may also be accessed as normal Visual Basic properties, but only at run-time. Each parameter must be referenced with an index, such that index 0 represents the X axis, index 1 represents the Y axis, and so on. For example:

DMCDiagnosticTests1.DMCExpectedKP(0) = 100

Parameters referenced in this way are not saved, but they will override values stored in the file DMCTEST.DAT.

### **Version Test**

DMCModel Property

## Parameter Test

Parameter	Description	Data Type	Default	Modify?
DMCExpectedKP	Expected value for the Proportional Constant or KP variable.	double	6.0	Yes
DMCExpectedKD	Expected value for the Derivative Constant or KD variable.	double	64	Yes
DMCExpectedKI	Expected value for the Integrator or KI variable.	double	0.0	Yes
DMCExpectedTL	Expected value for the Torque Limit or TL variable.	double	9.998	Yes
DMCExpectedOF	Expected value for the Offset or OF variable.	double	0.0	Yes
DMCExpectedMT	Expected value for the Motor Type or MT variable.	long	1	Yes
DMCExpectedCE	Expected value for the Configure Encoder or CE variable.	long	0	Yes
DMCExpectedTM	Expected value for the Time or TM variable.	long	0	Yes

## Stop Test

Parameter	Description	Data Type	Default	Modify?
DMCStopTestVelocity1	Lower bounds of motion. Any motion less than or equal to this value is considered no motion. Units are in counts per second.	long	100	Yes
DMCStopTestVelocity2	Upper bounds of motion. Any motion greater than or equal to this value is considered excessive motion. Units are in counts per second.	long	10000	Yes

## Motion Test

Parameter	Description	Data Type	Default	Modify?
DMCMotionTestDistance1	Lower bounds of distance motor should move. Units are in counts per second.	long	100	Yes
DMCMotionTestDistance2	Upper bounds of distance motor should move. Units are in counts per second.	long	10000	Yes
DMCMotionTestVolts	Offset pulse to apply in volts.	double	5.0	No
DMCMotionTestTime	Time duration of offset pulse in milliseconds.	long	20	No

## Stability Test

Parameter	Description	Data Type	Default	Modify?
DMCStabilityTestStepSize	Step size for	long	200	No

DMCStabilityTestNumOfPeakTests	step response. Number of iterations to test for peak step.	long	100	No
DMCStabilityTestNumOfDriftTests	Number of iterations to test for drift.	long	50	No
DMCStabilityTestSP	Speed.	long	0	No*
DMCStabilityTestAC	Acceleration.	long	0	No*
DMCStabilityTestDC	Deceleration.	Long	0	No*

\*If the value is left at 0, the Stability Test will use the current value from the controller.

## Tuning

Parameter	Description	Data Type	Default	Modify?
DMCTuningWindow	Test window (in counts) for position error.	long	2	No

Note: Tuning also uses the parameters from the Motion Test, and the SP, AC, and DC values from the Stability Test.

# Appendix 1 - Program Examples

These training exercises should be used to learn the use of the Galil Visual Basic tool kit with Visual BASIC. Each lesson introduces new techniques or different tools. The first 4 lessons introduce the use of the communication tool. Many other tools are demonstrated in the remaining examples.



## Experiment 1- Display Motor Position

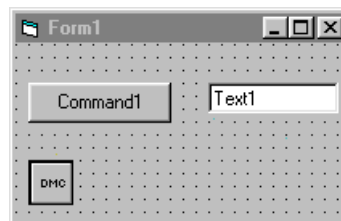
**Purpose:** The design objective is to create a command button and a text box. When you click on the button, the motor position shows in the box.

### Procedure

- Execute the Visual Basic program. This displays Form 1.
- Pull down the Project menu and select Components. Click on the box next to the Control

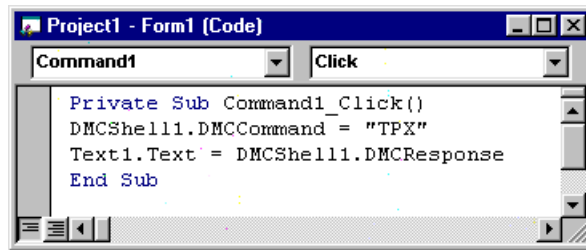
Galil DMCSHELL Control

- Hit the OK button. This adds a DMCSHELL icon to the tool box on the left: 
- Double click on the DMCSHELL icon. This loads it on the form. Drag the icon to the corner.
- Double click on the command button icon in the tools section.  This adds a command button to the form.
- Double click on the text icon. It is located on the left side of the second row and is marked 'ab|'. This loads the text box on the form.
- Select the command button. Find the property list on the right and change the caption property to "TPX".
- Arrange the tools as shown here



- Double click on the command 1 button. This opens command1\_click in the program editor.
- Add the instructions:

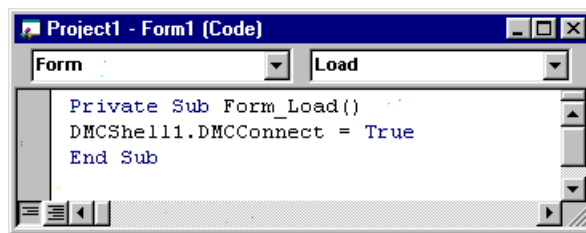
```
DMCSHELL1.DMCCOMMAND = "TPX"  
Text1.TEXT = DMCSHELL1.DMCRESPONSE
```



*Note: Upon clicking, this causes the program to send the command TPX to the controller and to display the response on the Text1 box.*

- Close command1\_click, returning to Form1.
- Double click on Form1 anywhere on the background that is not a control. This opens the load procedure of the object form.
- Add the instruction

DMCSHELL1.DMCCONNECT = True



- Pull down the menu on the right and select 'Unload'.
- In the unload procedure place the command

DMCSHELL1.DMCCONNECT = False

- To run the program, click on Run and select Start.
- Click on Command 1 and the position of the X motor shows in the text box.
- To stop the program, click on Run and select End.

## Experiment 2- Interface to Start/Stop Motor

**Purpose:** The objective is to design a user interface that can start the X motor, stop it, and display its position.

### Procedure

- Start a new project and add the DMCSHELL control
- Draw 3 command buttons.
- Draw one text box.
- Draw one DMCSHELL control .
- To define the procedures, double click on command1 and add the instructions:

DMCSHELL1.DMCCOMMAND = "JG1000"  
DMCSHELL1.DMCCOMMAND = "BGX"

- Close the code window for command1
- Double click on command2 and add the instruction

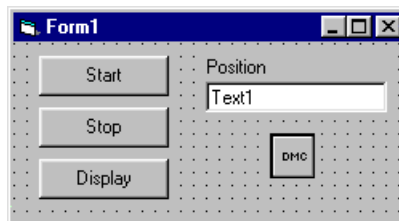
```
DMCShell1.DMCCCommand = "STX"
```

- Close the code window for command2
- Double click on command 3 and add the instruction

```
DMCShell1.DMCCCommand = "TPX"
```

```
text1.text = DMCShell1.DMCResponse
```

- Close the code window of command3
- To label the command buttons, click on command1, then access the properties window on the right. Change the caption property to 'Start'
- Repeat the procedure with command 2 and 3 with the labels STOP and DISPLAY respectively.
- To label the text box, double click on the label icon. This is the icon with the letter A at the upper right-hand corner of the tool box.
- Now select the properties of label 1 and make the caption POSITION.
- Arrange the tools as shown below.



- Double click on the form and add to the load procedure

```
DMCShell1.DMCCConnect = True
```

- Go to the unload procedure and add:


```
DMCShell1.DMCCConnect = False
```

- To run the program, click on Run and select Start.

## Experiment 3- Using Sliders

**Purpose:** Use sliders to change the jog speed of a motor.

### Procedure:

- Start a new project
- Add the DMCShell to the project and add it to the form.
- Draw 2 command boxes
- Draw 1 text box
- Draw 1 horizontal scroll bar 
- Draw 1 caption and change the caption property to "Speed"
- Change the caption of first command button to "Jog" and the second to "Stop"
- Add this code under command1\_click

```
DMCShell1.DMCCCommand = "JG" & Str(hscroll1.value)
```

```
DMCShell1.DMCCCommand = "BGX"
```

- Add this code under command2\_click

DMCShell1.DMCCCommand = "STX"

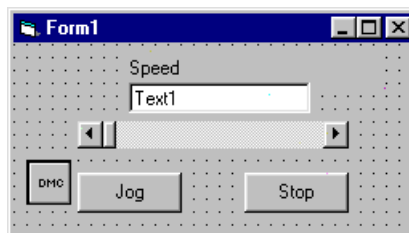
- Double click on the scroll bar and add this code under hscroll1\_change

DMCShell1.DMCCCommand = "JG" & Str(hscroll1.value)  
text1.text = hscroll1.value

- Change the scroll bar properties to

smallchange = 100  
largechange = 1000

- Move the controls as shown below:



- Under form\_load add

DMCShell1.DMCCConnect = True

- Under form\_unload add

DMCShell1.DMCCConnect = False

## Experiment 4- User Inputs Gear Ratio

**Purpose:** The objective is to design a system where the user can set the gear ratio between two motors and display their positions. Error handlers are introduced for the DMCShell.

### Procedure

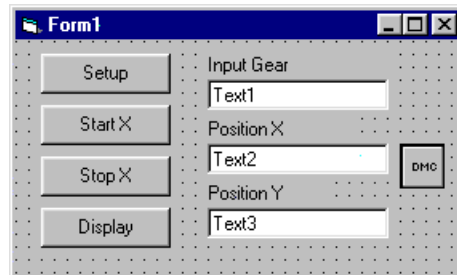
- Start a new project
- Add the DMCShell control.
- Draw 1 DMCShell
- Draw 4 command buttons
- Draw 3 text boxes
- Draw 3 label boxes.
- Now select the properties of the command boxes and make the captions as follows

command 1 - setup  
command 2 - start x  
command 3 - stop x  
command 4 - display

- Now select the properties of the labels and make the captions as follows

Label 1 - Input Gear  
 Label 2 - Position X  
 Label 3 - Position Y

- Position the controls as shown



- Double click each button to define their functions as follows:

#### Command 1 (Setup)

```
DMCShell1. DMCCCommand = "STX"
DMCShell1. DMCCCommand = "AMX"
DMCShell1. DMCCCommand = "DP0,0"
DMCShell1. DMCCCommand = "GAX"
DMCShell1. DMCCCommand = "GR,1"
```

#### Command 2 (Start X)

```
DMCShell1. DMCCCommand = "JG 1000"
DMCShell1. DMCCCommand = "BGX"
```

#### Command 3 (Stop X)

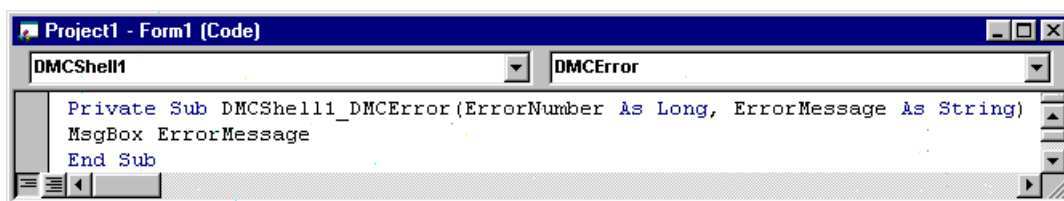
```
DMCShell1. DMCCCommand = "STX"
```

#### Command 4 (Display)

```
DMCShell1. DMCCCommand = "TPX"
text2.text = Val(DMCShell1.DMCResponse)
DMCShell1. DMCCCommand = "TPY"
text3.text = Val(DMCShell1.DMCResponse)
```

- To get error messages from the DMCShell place the following code in the DMCShell1 error event. ( Double click the shell )

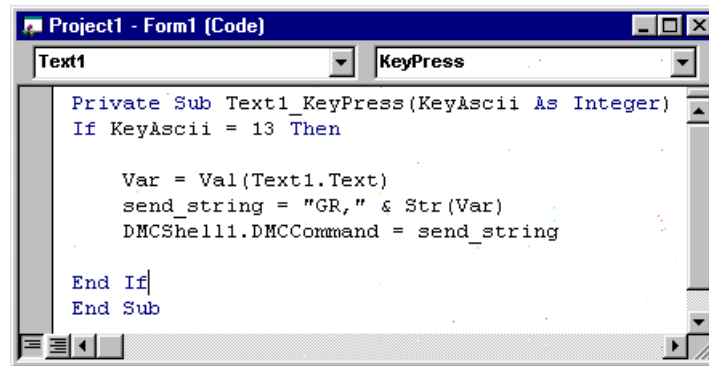
MsgBox ErrorMessage





- To allow data input in the form of new gear ratio, double click on Text 1 box. That opens up the code editor for the object Text 1. Change the procedure to KeyPress and add the instructions:

```
if KeyAscii = 13 then
    var = Val (text1.text)
    send_string = "GR," + Str(var)
    DMCSHell1.DMCCCommand = send_string
end if
```



- Add this code to the form load procedure  
DMCSHell1.DMCCConnect = True
- In the unload procedure place the command  
DMCSHell1.DMCCConnect = False
- Click on Run to start or stop the program. Clicking on Set-up will set the initial parameters. Start X starts the motion of both X and Y. You may change the gear ratio by entering new values to the Text 1 box.

## OTHER TOOLKIT OBJECTS

### Experiment 5- Polling Windows

**Purpose:** Learn to use the polling window to automatically display data from the control card.

#### Procedure:

- Start a new project
- Add the DMCSHell object and draw it on the form
- Add the DMCPoll object and draw two on the form
- Select each one of the polling windows located on the form and go to the properties and change the following:

#### DMCPoll1

DMCPollInterval to 100.  
DMCCCommand to TPX  
DMCFormat to 10.0

#### DMCPoll2

DMCPollInterval to 100.

DMCCommand to TPY  
DMCFormat to 10.0

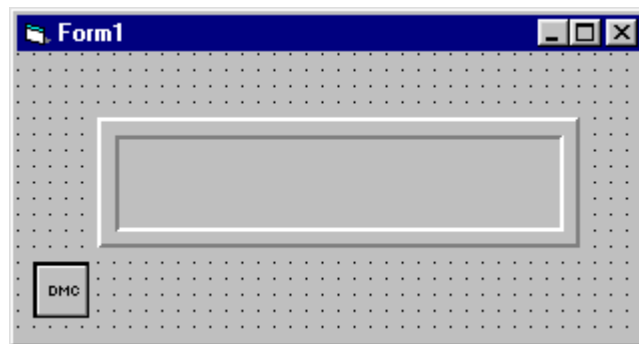
- Enter the commands to establish communications with the controller and activate the polling window. In the 'Form Load' procedure enter the following commands:

```
DMCShell1.DMCCConnect = True  
DMCPoll1.DMCPollController = True  
DMCPoll2.DMCPollController = True
```

- In the form unload procedure place the command

```
DMCShell1.DMCCConnect = False
```

- Run the project then move the X and Y motor by hand to see the values change.
- When finished, end the program by pulling down the "Run" menu and selecting "End"



*Note: There are many other properties that were not touched on here. They mainly affect the appearance of the control. For instance, the polling window can scale the data before it is displayed. Other functions include: offsets, and putting prefixes or suffixes on the data. ( i.e. "1000 counts" ). Many polling windows can be placed on one form but remember that each one takes some overhead from Windows and the controller.*

## Experiment 6- Terminal and Scope Objects

**Purpose:** Create a project using the storage scope and the terminal object. This program will allow users to enter any sequence with the terminal then display the actual motion with the scope.

**Procedure:** This program will have a terminal where the user enters a coordinated motion sequence ( i.e. "CR 1000,0,180" "VP 1000,10000" etc. ) then a "VE". Do not enter a "BGS" to start the motion; a button on the form will send the "BGS" and start the record function at the same time.

- Start a new project
- Add the tools "DMCShell", "DMCTerminal", and "DMCScope"
- Draw a DMCShell, a DMCTerminal, and a DMCScope object on the form
- Select the DMCScope and set the following properties to tell the scope tool to graph Y position in respect to X position and take 700 samples- one every 4 milliseconds:

DMCHorizontalSource	to	1
DMCPointsPerAxis	to	700
DMCSampleTime	to	1- 4ms
DMCVerticalSource	to	1

- Draw a button on the bottom of the screen.
- Select the button and go to the properties window . Set the 'caption' property of the button to "Start Motion".
- Now put code under the button to start recording and start the motion:

```
DMCScope1.DMCCaptureData = True
DMCScope1.DMCCCommand = "BGS"
```

- Enter the following code in the 'form load' procedure;

```
DMCScope1.DMCCConnect=True
```

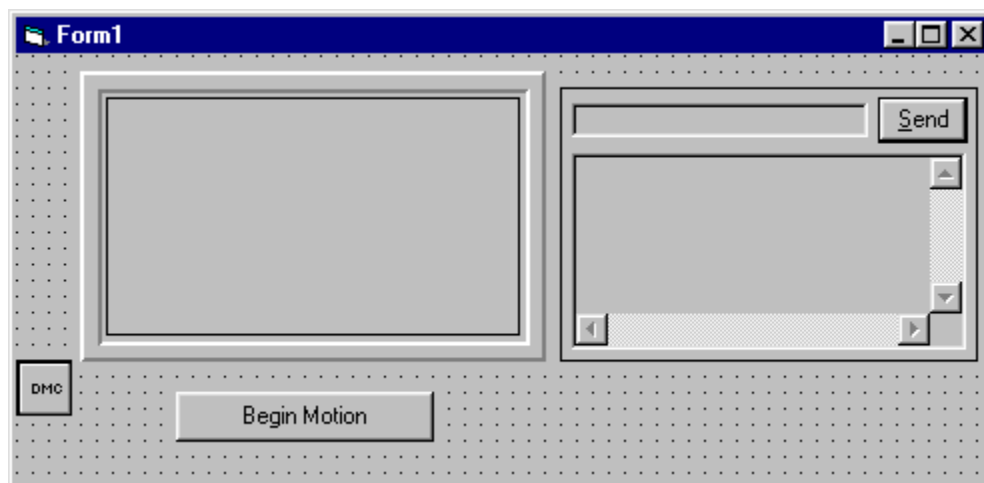
- In the unload procedure place the command

```
DMCScope1.DMCCConnect=False
```

- Run the program.
- Enter any coordinated sequence line by line in the terminal data entry area. Example:

```
VP1000,1000
CR 1000,0,180
VP-1000,500
```

- End the sequence with the "VE" command, **do not enter "BGS"**
- Press the "Start Motion" button. The motors should run and in about 10 seconds the graph will appear.



Because the sample time and number of points have been set at 4 ms and 700 this program will take about 2.8 seconds to gather the data. A few more seconds are required to upload the data and display it.

## Experiment 7- Move Tool

***Purpose:** Use the move tool to display 2-D vector motion before the motors move.*

### Procedure

- Start a new project

- Add the tools 'DMCShell1', 'DMCTerminal', and 'DMCMove' and draw one of each on the form
- Draw a button on the form and change the caption to "Show Move"
- Enter this code under the button

```
DMCMove1.DMCFileName = "c:\dmccox\movefile.dmc"
DMCMove1.Open
```

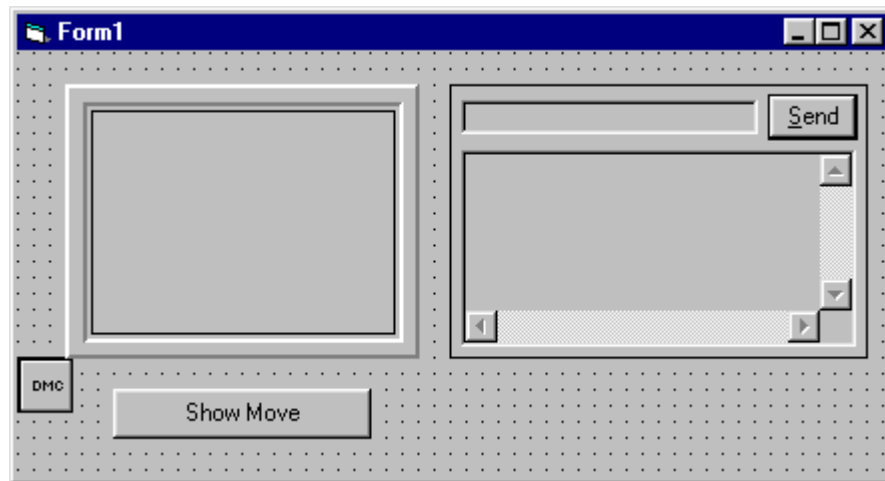
- In the 'form load' procedure enter the code to begin communication:

```
DMCShell1.DMCConnect = True
```

- In the unload procedure place the command

```
DMCShell1.DMCConnect = False
```

- Arrange the controls as follows



- Run the program.
- From the terminal type "ED" and hit return. The program editor will appear. This will be used to create a program.
- Enter any program: start with a label ( i.e. "#A" ) and end with a "EN" and put in any number of VP or CR commands. Here is an example;

```
#A
VP1000,1000
VP2000,1000
VP2000,2000
VP1000,2000
VP1000,1000
VE
BGS
EN
```

- From the "File" menu select "Save File" and save it as "c:\dmccox\movefile.dmc"
- Close the editor window
- Hit the "Show Move" button to see the file in the DMCMove control

## Experiment 8- Teach and Playback

**Purpose:** This lesson will explain the use of the array and playback objects. This program will record motor positions and play them back.


### Procedure

- Start a new project by selecting “New Project” under the “File” menu.
  - Add the controls: DMCArray and DMCPPlayback.
  - Add the DMCSHELL control to the form.
- In the form load procedure put in the code


```
DMCSHELL1.DMCCONNECT=TRUE
```

- In the unload procedure place the command

```
DMCSHELL1.DMCCONNECT=FALSE
```

- Select the array tool and draw it anywhere on the screen: 
- Hit F4 to get the property window for the array tool and change the properties as follows:

DMCFileName	to	"C:\dmcocx\arrayout.txt"
DMCSampleTime	to	2-8 ms

- Select the playback tool and place it anywhere on the screen: 
- Hit F4 to get the property window of the playback tool and change the properties as follows:

DMCOutputFileName	to	"c:\dmcocx\sendout.sen"
DMCInputFileName	to	"c:\dmcocx\arrayout.txt"
DMCPlaybackRate	to	1-4 ms
DMCPlayback Mode	to	1-Linear Interpolation

- Draw a button anywhere and place the following code under it

```
DMCSHELL1.DMCCOMMAND = "MOX"  
DMCSHELL1.DMCCOMMAND = "MOY"  
DMCArray1.DMCAXISTORECORD(0) = 1  
DMCArray1.DMCDATATYPE(0) = 1  
DMCArray1.DMCAXISTORECORD(1) = 2  
DMCArray1.DMCDATATYPE(1) = 1  
DMCArray1.DMCRECORDDATA = 1
```

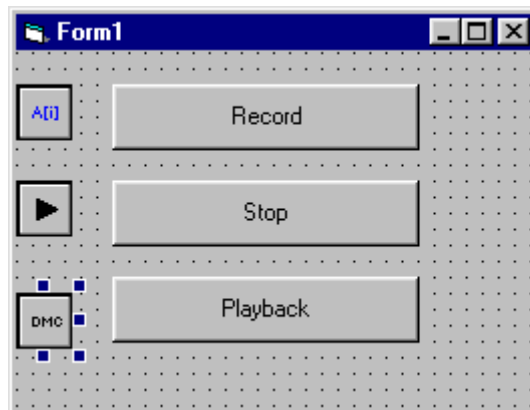
- Change the caption property of the button to “RECORD”
- Draw another button below the first and place the following code under it:

```
DMCArray1.DMCRECORDDATA=0  
DMCSHELL1.DMCCOMMAND="SH"
```

- Change this buttons caption property to “STOP”
- Draw one more button under the other two and place this code under it:

```
DMCPPlayback1.DMCPLAYBACKOPERATION = 1  
DMCPPlayback1.DMCPLAYBACKOPERATION = 2
```

- Change this buttons caption property to “Playback”
- The form should look something like this:



- Run the program.
- Hit the “Record” button then move the motors by hand
- Hit the “Stop” button.
- Hit the “Playback” button. The motors should replay the motion.

**THIS PAGE LEFT BLANK INTENTIONALLY**

# Appendix 2 – Data Record, Error Code, Registry Entry, and other Global Constants

The following global constants can be found in the file DMCCOM40.BAS which may be added to your projects.

**These error codes are returned as a variable into the error event procedure of each ActiveX tool.**

## Data Record Constants

The DMCGetDataRecordByItemId function retrieves a data record item by unique Id while the DMCGetDataRecord function retrieves a data record item by offset. While data record item offsets can change with the firmware revision, the data record item Ids always remain the same.

```
' Constant values for data record data types
Public Const DRTypeUnknown = 0
Public Const DRTypeCHAR = 1
Public Const DRTypeUCHAR = 2
Public Const DRTypeSHORT = 3
Public Const DRTypeUSHORT = 4
Public Const DRTypeLONG = 5
Public Const DRTypeULONG = 6

' Constant values for data record item Ids to be used with the function
' DMCGetDataRecordByItemId
Public Const DRIdSampleNumber = 1
Public Const DRIdGeneralInput0 = 2
Public Const DRIdGeneralInput1 = 3
Public Const DRIdGeneralInput2 = 4
Public Const DRIdGeneralInput3 = 5
Public Const DRIdGeneralInput4 = 6
Public Const DRIdGeneralInput5 = 7
Public Const DRIdGeneralInput6 = 8
Public Const DRIdGeneralInput7 = 9
Public Const DRIdGeneralInput8 = 10
Public Const DRIdGeneralInput9 = 11
Public Const DRIdGeneralOutput0 = 12
Public Const DRIdGeneralOutput1 = 13
Public Const DRIdGeneralOutput2 = 14
Public Const DRIdGeneralOutput3 = 15
Public Const DRIdGeneralOutput4 = 16
Public Const DRIdGeneralOutput5 = 17
Public Const DRIdGeneralOutput6 = 18
Public Const DRIdGeneralOutput7 = 19
Public Const DRIdGeneralOutput8 = 20
Public Const DRIdGeneralOutput9 = 21
Public Const DRIdErrorCode = 22
Public Const DRIdGeneralStatus = 23
Public Const DRIdSegmentCountS = 24
Public Const DRIdCoordinatedMoveStatusS = 25
```



```

Public Const DRIdCoordinatedMoveDistancesS = 26
Public Const DRIdSegmentCountT = 27
Public Const DRIdCoordinatedMoveStatusT = 28
Public Const DRIdCoordinatedMoveDistanceT = 29
Public Const DRIdAnalogInput1 = 30
Public Const DRIdAnalogInput2 = 31
Public Const DRIdAnalogInput3 = 32
Public Const DRIdAnalogInput4 = 33
Public Const DRIdAnalogInput5 = 34
Public Const DRIdAnalogInput6 = 35
Public Const DRIdAnalogInput7 = 36
Public Const DRIdAnalogInput8 = 37
Public Const DRIdAxisStatus = 38
Public Const DRIdAxisSwitches = 39
Public Const DRIdAxisStopCode = 40
Public Const DRIdAxisReferencePosition = 41
Public Const DRIdAxisMotorPosition = 42
Public Const DRIdAxisPositionError = 43
Public Const DRIdAxisAuxillaryPosition = 44
Public Const DRIdAxisVelocity = 45
Public Const DRIdAxisTorque = 46
Public Const DRIdSlotModulePresent = 47
Public Const DRIdSlotModuleType = 48
Public Const DRIdSlotNumberOfIOPoints = 49
Public Const DRIdSlotIODirection = 50
Public Const DRIdSlotAnalogRange = 51
Public Const DRIdSlotRangeType = 52
Public Const DRIdSlotIOData0 = 53
Public Const DRIdSlotIOData1 = 54
Public Const DRIdSlotIOData2 = 55
Public Const DRIdSlotIOData3 = 56
Public Const DRIdSlotIOData4 = 57
Public Const DRIdSlotIOData5 = 58
Public Const DRIdSlotIOData6 = 59
Public Const DRIdSlotIOData7 = 60

```

' Constant values for axis Ids to be used with the function

' DMCGetDataRecordByItemId

```

Public Const DRIdAxis1 = 1
Public Const DRIdAxis2 = 2
Public Const DRIdAxis3 = 3
Public Const DRIdAxis4 = 4
Public Const DRIdAxis5 = 5
Public Const DRIdAxis6 = 6
Public Const DRIdAxis7 = 7
Public Const DRIdAxis8 = 8

```

'Constant values for IOC7007 slot Ids to be used with the function

'DMCGetDataRecordByItemId

'DMCGetDataRecordSlotIDs

```

Public Const DRIdSlot1 = 15
Public Const DRIdSlot2 = 16
Public Const DRIdSlot3 = 17
Public Const DRIdSlot4 = 18
Public Const DRIdSlot5 = 19
Public Const DRIdSlot6 = 20
Public Const DRIdSlot7 = 21

```

## Error Code Constants

```
' Error Codes
Public Const DMCNOERROR = 0
Public Const DMCERROR_TIMEOUT = -1
Public Const DMCERROR_COMMAND = -2
Public Const DMCERROR_CONTROLLER = -3
Public Const DMCERROR_FILE = -4
Public Const DMCERROR_DRIVER = -5
Public Const DMCERROR_HANDLE = -6
Public Const DMCERROR_HMODULE = -7
Public Const DMCERROR_MEMORY = -8
Public Const DMCERROR_BUFFERFULL = -9
Public Const DMCERROR_RESPONSEDATA = -10
Public Const DMCERROR_DMA = -11
Public Const DMCERROR_ARGUMENT = -12
Public Const DMCERROR_DATAARECORD = -13
Public Const DMCERROR_DOWNLOAD = -14
Public Const DMCERROR_FIRMWARE = -15
Public Const DMCERROR_CONVERSION = -16
Public Const DMCERROR_RESOURCE = -17
Public Const DMCERROR_REGISTRY = -18
Public Const DMCERROR_BUSY = -19
Public Const DMCERROR_DEVICE_DISCONNECTED = -20
```

## Registry Entry Constants

```
' Constant values for GALILREGISTRY structure

' Controller Type
Public Const ControllerTypeISABus = 0
Public Const ControllerTypeSerial = 1
Public Const ControllerTypePCIBus = 2
Public Const ControllerTypeUSB = 3

' Device Drivers
Public Const DeviceDriverWinRT = 0
Public Const DeviceDriverGalil = 1

' Serial Handshake
Public Const SerialHandshakeHardware = 0
Public Const SerialHandshakeSoftware = 1

' Data Record Access
Public Const DataRecordAccessNone = 0
Public Const DataRecordAccessDMA = 1
Public Const DataRecordAccessFIFO = 2

' Ethernet Protocol
Global Const EthernetProtocolTCP = 0
Global Const EthernetProtocolUDP = 1
```

```

' Structures

' To add/change/delete registry information
Type GALILREGISTRY
    Model As String * 16
    DeviceNumber As Integer
    DeviceDriver As Integer
    Timeout As Long
    Delay As Long
    ControllerType As Integer
    CommPort As Integer
    CommSpeed As Long
    Handshake As Integer
    Address As Integer
    Interrupt As Integer
    DataRecordAccess As Integer
    DMAChannel As Integer
    DataRecordSize As Integer
    RefreshRate As Integer
    SerialNumber As Integer
    PNPHardwareKey As String * 64
End Type

```

## Global Constants (and Public Constants)

```

Public Const DMC400 = "DMC-400"
Public Const DMC600 = "DMC-600"
Public Const DMC700 = "DMC-700"
Public Const DMC1000 = "DMC-1000"
Public Const DMC1200 = "DMC-1200"
Public Const DMC1410 = "DMC-1410"
Public Const DMC1411 = "DMC-1411"
Public Const DMC1412 = "DMC-1412"
Public Const DMC1417 = "DMC-1417"
Public Const DMC1500 = "DMC-1500"
Public Const DMC1600 = "DMC-1600"
Public Const DMC1700 = "DMC-1700"
Public Const DMC1800 = "DMC-1800"
Public Const DMC1802 = "DMC-1802"
Public Const DMC2000 = "DMC-2000"
Global Const DMC2100 = "DMC-2100"
Global Const DMC90064 = "IOC-90064"

```

# Appendix 3 - Distributing Your Application

NOTES FOR DISTRIBUTING APPLICATIONS BUILT WITH THE GALIL ACTIVEX TOOL KIT

**Visual Basic Setup Wizard**

Microsoft Visual Basic's Setup Wizard uses a dependency file to determine which application components need to be included in the distribution. The file `DEPEND32.INI` should be added to the Setup Wizard's dependency files (`SWDEPEND.INI` for Visual Basic 4.0, `VB5DEP.INI` for Visual Basic 5.0, and `VB6DEP.INI` for Visual Basic 6.0) so that Setup Wizard can automatically include all the files necessary for distribution with applications using the Galil ActiveX Tool Kit. The only exception to this is that the device driver files must be added to the Setup Wizard manually (See Manual Distribution below).

For Visual Basic 5.0/6.0, there is a dependency file (`.dep`) for each Galil ActiveX control (`.ocx`). The `.dep` files are installed in the same directory as the `.ocx` files. Visual Basic 5.0/6.0 should be able to determine the correct dependencies for your project.

## Manual Distribution

When you distribute your application using the Galil Windows Drivers and DLLs or the Galil ActiveX Tool Kit and OLE Custom Control Library, you must also distribute the following files:

### For Windows 98 Second Edition, Windows ME, Windows 2000, Windows XP

For PCI bus controllers, use `GLWDMPCI.SYS`. For ISA bus controllers, use `GLWDMISA.SYS`. For USB, use `GLWDMUSB.SYS`. These files must reside in the `WINDOWS\SYSTEM32\DRIVERS` directory.

These device drivers are normally installed using `.INF` files. You can place `GLWDMPCI.INF`, `GLWDMISA.INF`, `GLWDMUSB.INF` in the `WINDOWS\INF` or `WINNT\INF` directory and the operating system will find the Galil `INF` files when looking for a driver for the controller.

You should also copy `DMC32.DLL`, `dmcreg.ocx` and `DMCBUS32.DLL` (for bus controllers), or `DMCSER32.DLL` (for serial controllers) in the `WINDOWS\SYSTEM` or `Winnt\System32` directory. You will also need to copy and register `dmcreg.ocx` with the communication dlls.

Note: the `WINDOWS` or `WINNT` directory may actually have a different, user-specified name.

### For Windows NT 4.0

If you are using the Galil drivers: For ISA bus controllers, copy `GALIL.SYS` to the `WINNT\SYSTEM32\DRIVERS` directory. For PCI or CompactPCI bus controllers, copy `GALILPCI.SYS` to the `WINNT\SYSTEM32\DRIVERS` directory.

You should also copy `DMC32.DLL`, `dmcreg.ocx` and `DMCBUS32.DLL` (for bus controllers), or `DMCSER32.DLL` (for serial controllers) in the `WINNT\SYSTEM32` directory. You will also need to copy and register `dmcreg.ocx` with the communication dlls.

Note: the `WINNT` directory may actually have a different, user-specified name.

## Microsoft Run-time Files

The following Microsoft run-time files:

`MSVCRT.DLL`

`MFC42.DLL`

NOTE: If your project is built using Visual Basic or Delphi, or a Version of Visual C++ other than 6.0, you will also need to distribute the run-time files for those tools.

# Appendix 4 - Troubleshooting

This product is designed to run under all Windows operating systems. There are, however, various differences between systems that may cause errors during installation or using the custom controls. This section covers the known problems and offers solutions.

## General Tips and Tricks

- When a new version of the tool kit is installed make an effort to first remove all older files. Many older versions of .VBX files were placed in the /Windows/system directory. The files to erase all have "DMC" in their filename and have a .VBX or .DLL extension.
- After a version upgrade old forms may not recognize the Shell object or other objects. This can be solved by first deleting the old tool off the form and adding the new one. In most cases the properties have not changed; no code modification is needed.
- Current versions of the tool kit place the files under the /DMCOCX directory. This may cause errors in current projects. Remove the old tools from the project then add the new tools.
- Make sure there is only one Shell object in the Visual Basic project for each controller in the system. To reference the shell from another form use the syntax *form\_name.DMCShell1.DMCCommand = "TPX"* for example.
- Try not to put too many polling windows on any form. Any more than 8 polling objects running at the same time can slow down the response of Windows. If many parameters need to be polled try using the 'timer' object supplied with Visual Basic and place all the queries under that routine.
- To make sure the DMCCConnect = False executed always stop the program with the 'close' menu item in the upper left hand corner of the active window. It may be easier to draw a button on the form labeled "Quit" that runs the DMCCConnect = False then the 'end' command to stop the program. If the program is stopped in any other way the serial port will be lost, and Windows will have to be restarted. Bus level communication will not be lost the first time this happens, but after a number of these bad stops the communication will hang.
- 'MG' commands within the program running on the controller should be avoided. Although the shell has been set up to handle these unsolicited responses it is not a stable solution. Here is the way it can be done:
  1. Send the command CW1 to the controller and burn it in. This offsets any unsolicited response by 128 ASCII.
  2. Turn on the polling on the DMCSHELL: DMCSHELL1.DMCPollController=True
  3. Set up code in the DMCSHELL response event to read the unsolicited response.

This configuration works well until a master reset is performed on the controller and the CW command is lost. Because this may happen when the designer is not available Galil does not recommend using this configuration.

The preferred method is to set up flag variables within the controllers program and set them to a known value at any place a MG would have been used. Then tell Visual Basic to poll that variable from the controller. When the variable changes value VB can send a message to the screen:

The code in the controller may look like this:

### *Preferred Method*

```
#A  
FLAG1=0  
PR1000  
BGX
```

### *MG Method*

```
#A  
PR1000  
BGX  
AMX
```

```
AMX
FLAG1=1
EN
```

```
MG "Move Complete"
EN
```

Use the timer control provided with VB to query the variable and display a message:

```
DMCShell11.DMCCommand = "FLAG1 = "

If DMCShell11.DMCResponse = 1 then
    msgbox "Move Complete"
    DMCShell11.DMCCommand = "FLAG1 = 0"
EndIf
```

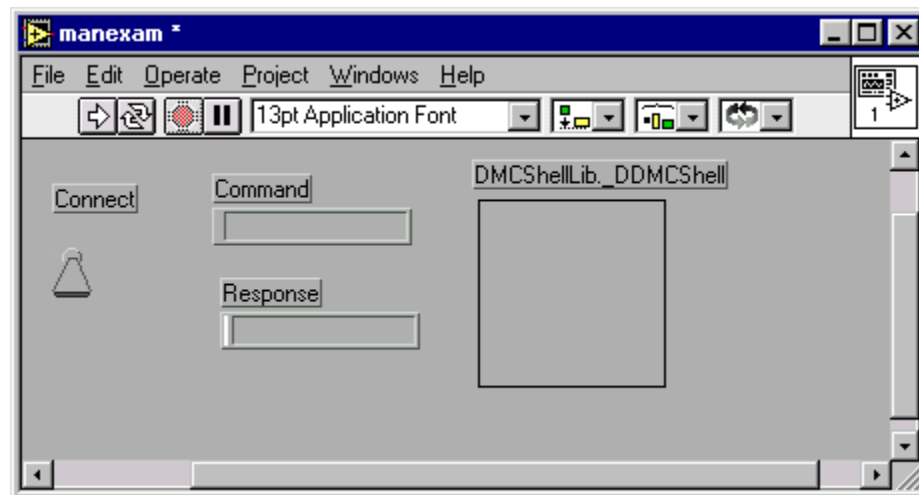
# Appendix 5 - Using the ActiveX Tool Kit with LabVIEW

LabView version 5.0 now supports ActiveX ( OCX ) controls. The standard Galil tool kit can be used without modification. Please note that while properties and methods are supported, 'events' are not. This is a limitation of LabView but should not effect the operation of the tools but it does prevent writing code to respond to error events.

This step by step example demonstrates using the DMCSHELL, DMCTerminal, and DMCPoll in a LabView application:

- Start with a new LabView project
- Add an ActiveX container, found in the controls palette, to the project
- Select the ActiveX container with the left mouse button and select 'Insert ActiveX Object'. Select DMCSHELL from the menu.
- Add one Boolean switch from the controls palette and draw it on the project.
- Add one string control and one string indicator and draw it on the project
- Left mouse click on the string control to bring up a menu. Select 'Limit to single line'.
- Create labels on the controls by left clicking the mouse on the control and selecting 'Show-Label' then typing the label text.

The project should look like this:



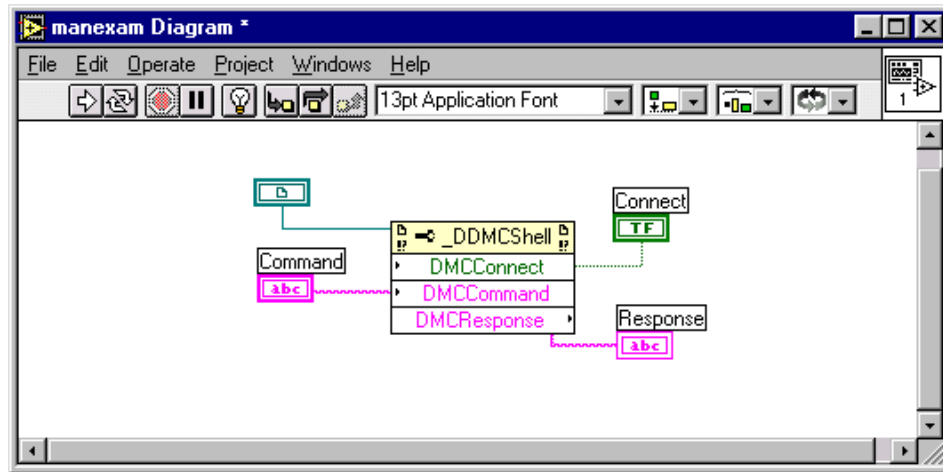
- Show the project diagram by selecting 'Show Diagram' from the Windows Menu.
- Add an ActiveX property node by selecting the Communications icon from the Functions palette then ActiveX from the sub-menu
- Enlarge the icon by dragging a lower corner so that it shows 3 property boxes as shown below



- Select the Connect Wire icon from the Tools palette
- Draw a wire from the DMCSHELL1 icon to the top left corner of the Automation icon show above.

- Change each property on the Automation icon ( which now reads \_DMCShell ) by placing the mouse pointer over a property box and left click to bring up a menu. Select DMCCConnect, DMCCCommand, and DMCCResponse for the three boxes.
- Left mouse click over the DMCCConnect box and select 'Change to Write'. Do the same for DMCCCommand.
- Draw a line using the Connect Wire tool from the DMCCConnect box to the Switch labeled 'Connect' ( a box with TF inside )
- Draw a line from the DMCCCommand box to the string control labeled Command.
- Draw a line from the DMCCResponse box to the string indicator labeled Response

The diagram should look like this:



- To run the program click the Run Continuously icon
- From the project window click the Connect switch to start communication with the controller.
- Type a valid Galil command in the Command string control and hit return
- The response should be displayed in the Response string indicator.

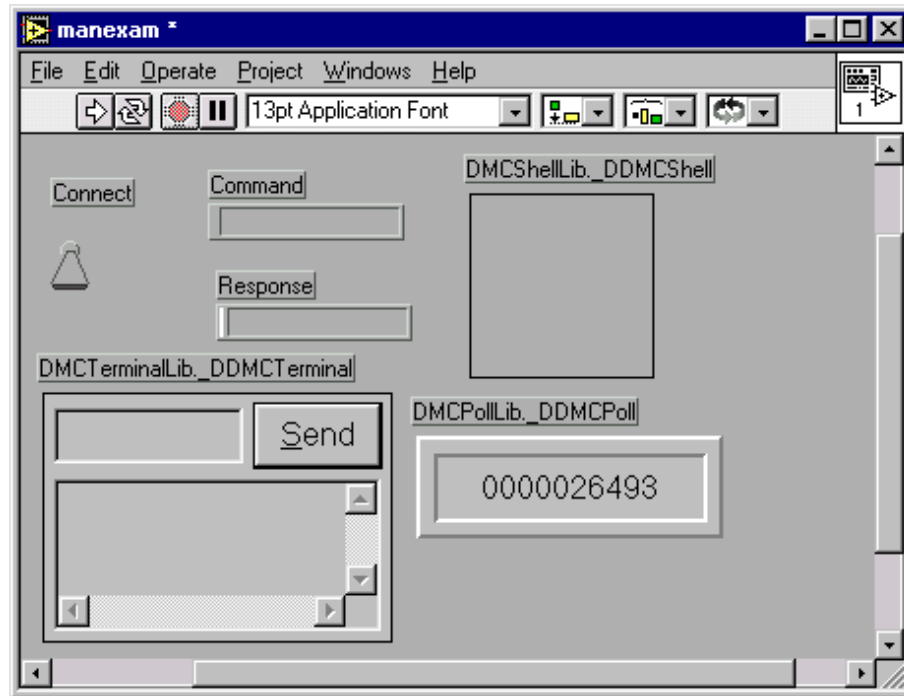
Before stopping the program make sure the controller is not connected: hit the connect switch to turn off the connection.

#### **Adding the terminal and polling windows.**

- Add two more ActiveX containers to the project.
- Make one a DMCTerminal, the other a DMCPoll
- Change the properties of the polling window by left clicking on the polling window then selecting DMCPoll Control Object - Properties. Change DMCCCommand to TPX.

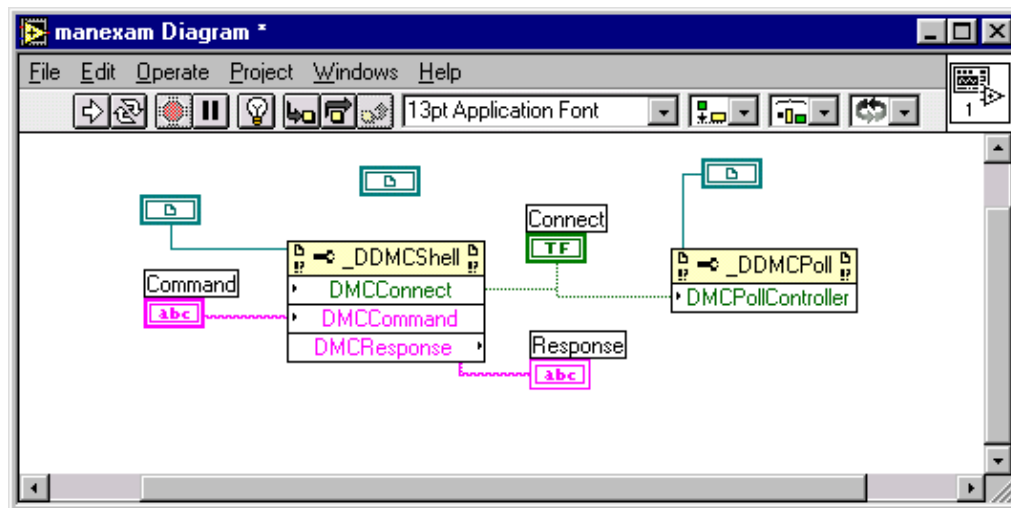
The project should look like this:





- Add a Property Node to the diagram and draw a line between the DMCPoll object and the top left corner of the node
- Change the property in the node to DMCPollController then select Change to Write.
- Draw a line from the DMCPollController box to the Connect Switch

The diagram should now look like this:



When the program is running the polling window will update. Commands can also be sent though the terminal.

**THIS PAGE LEFT BLANK INTENTIONALLY**

# Appendix 6 - Using the ActiveX Toolkit with VB .NET

This section describes the procedure for using the Galil Active-X controls in VisualBasic.NET (VB.NET). VB.NET is the latest Visual Basic design environment that replaces VB6. Overall, VB.NET is very similar in functionality and appearance to VB6, however there are some slight differences in coding and formatting related to the Galil Active-X Toolkit that will be detailed in the following example.

After opening the VB.NET design screen and selecting a new Windows form as the project, the Galil Active-X controls will have to be added to the design environment. Assuming the Galil Active-X toolkit has been successfully installed, select “Add/Remove Toolbox items...” under the “Tools” pull down menu. When the dialog box appears, select the desired controls from the COM Components list with a checkbox and then select “OK”. Figure-1 below shows the “Customize Toolbox” dialog box with the Galil controls checked.

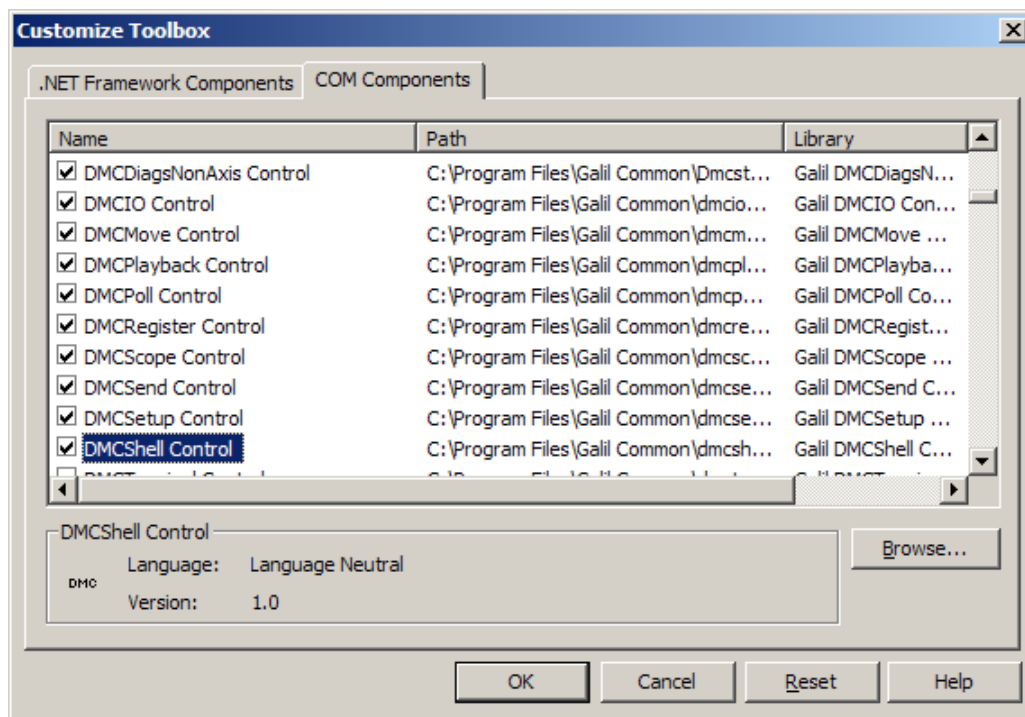


Figure 1 – Customize Toolbox Dialog Box

The Galil Active-X controls will then appear on the toolbox on the left side of the design screen under the toolbox menu “Windows Forms.” One of the main differences between VB.NET and VB6 is that the Active-X controls will only have to be added once to the toolbox. Even when new project is started at a different time, the Galil Active-X controls will already be available in the toolbox. Figure-2 below shows the VB.NET design environment with a sample program using the Galil Active-X controls. Note the Active-X controls in the toolbox on the left.

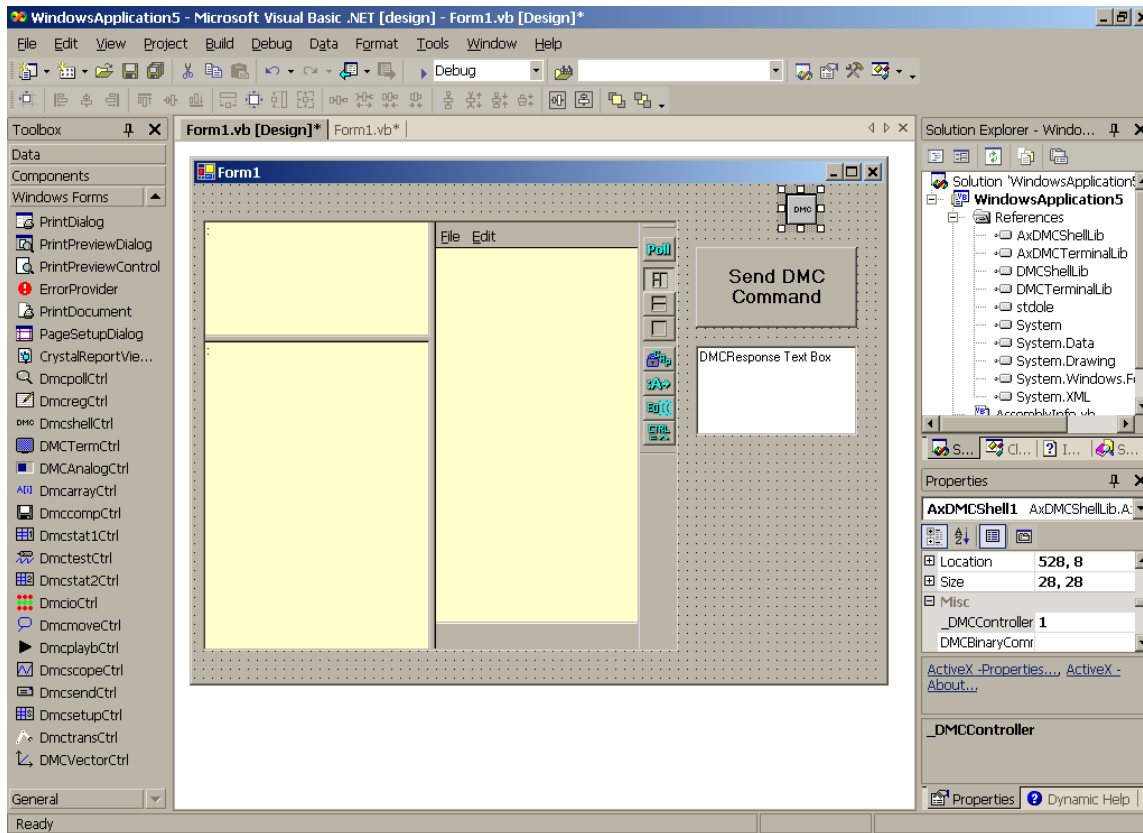


Figure 2 – VB.NET Design Environment

Adding the Galil Active-X controls to the Form is the same as in VB6, simply select (highlight) the desired control from the toolbox and drag it onto the Form. The properties for each control can be set just as in VB6, by selecting the desired property from the Properties dialog located in the lower right corner of the design environment.

Some common properties can also be set by “right clicking” the control on the Form and selecting properties. This action prompts a dialog box to appear with the most common control properties, such as “DMCController.” Figure-3 below shows the Properties dialog box after “right clicking” the DMCShell control on the form.

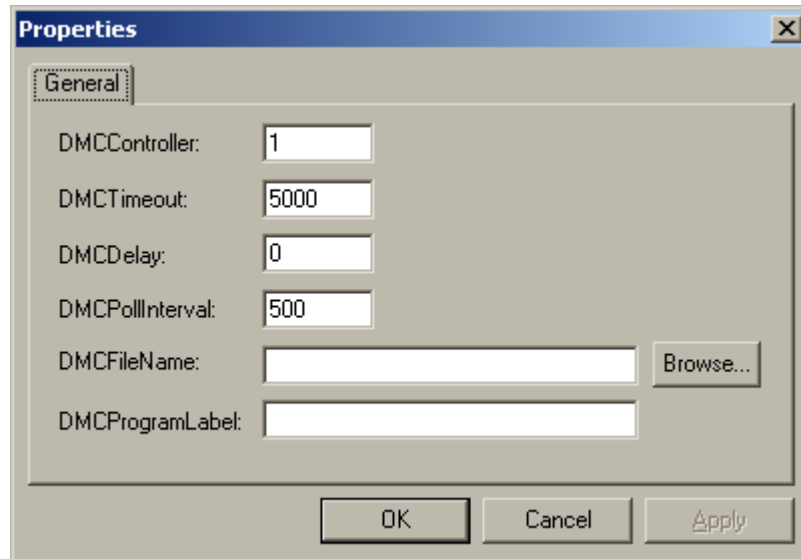


Figure 3 – Properties dialog for DMCSHELL Control

The code window for the Form can be viewed by “clicking” the tab labeled “Form1.vb” at the top of the design window. Just as in VB6, the code window can also be viewed by “double clicking” anywhere on the form, which also automatically brings up the “Form\_Load” procedure.

Figure-4 shows a sample program as it appears in the code window. Note how VB.NET allows the programmer to expand/collapse the subroutines (by clicking the + or – sign).

Note: VB.NET automatically adds the code under the heading “Windows Forms Designer generated code”. This code is added to give the programmer complete access to the code that generates the form itself. For most applications, it is not necessary to modify this code (hence it is grayed out).

Note2: VB.NET also modifies some properties into methods such that they now use the set\_ and get\_ prefixes.

```
AxDMCAArray1.set_DMCAxistoRecord(0)=1
```

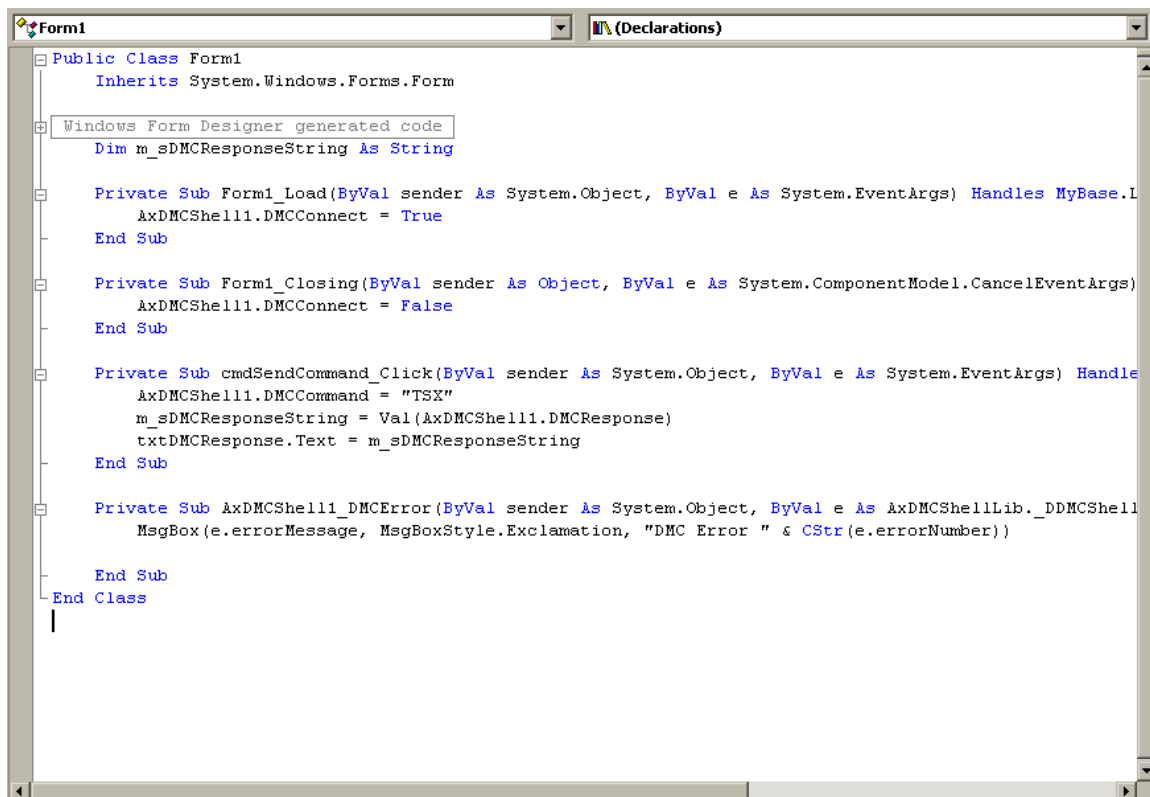


Figure 4 – VB.NET Code Window

VB.NET defines the Active-X controls with the prefix "Ax." Hence, any reference to the Active-X controls in the code environment requires that they begin with "Ax." This is to help with programming clarity and distinguish Active-X controls from other code objects.

Another major difference between VB.NET and VB6 is that the "Form1\_Unload" procedure in VB6 is now referred to as "Form1\_Closing" in VB.NET. Also, to find the Form event procedures, select "(Base Class Events)" from the left pull down menu at the top of the code window, then select the desired Form event from the right pull down menu.

Event procedures can be used just as in VB6, however the argument that gets passed by the routine is an object with associated strings and values (as opposed to explicitly returning a separate message string and error number). The "AxDMCShell1\_DMCErrors" event routine, for example (shown in figure 4) accesses the errorMessage and errorNumber by "dotting" the returned object "e."