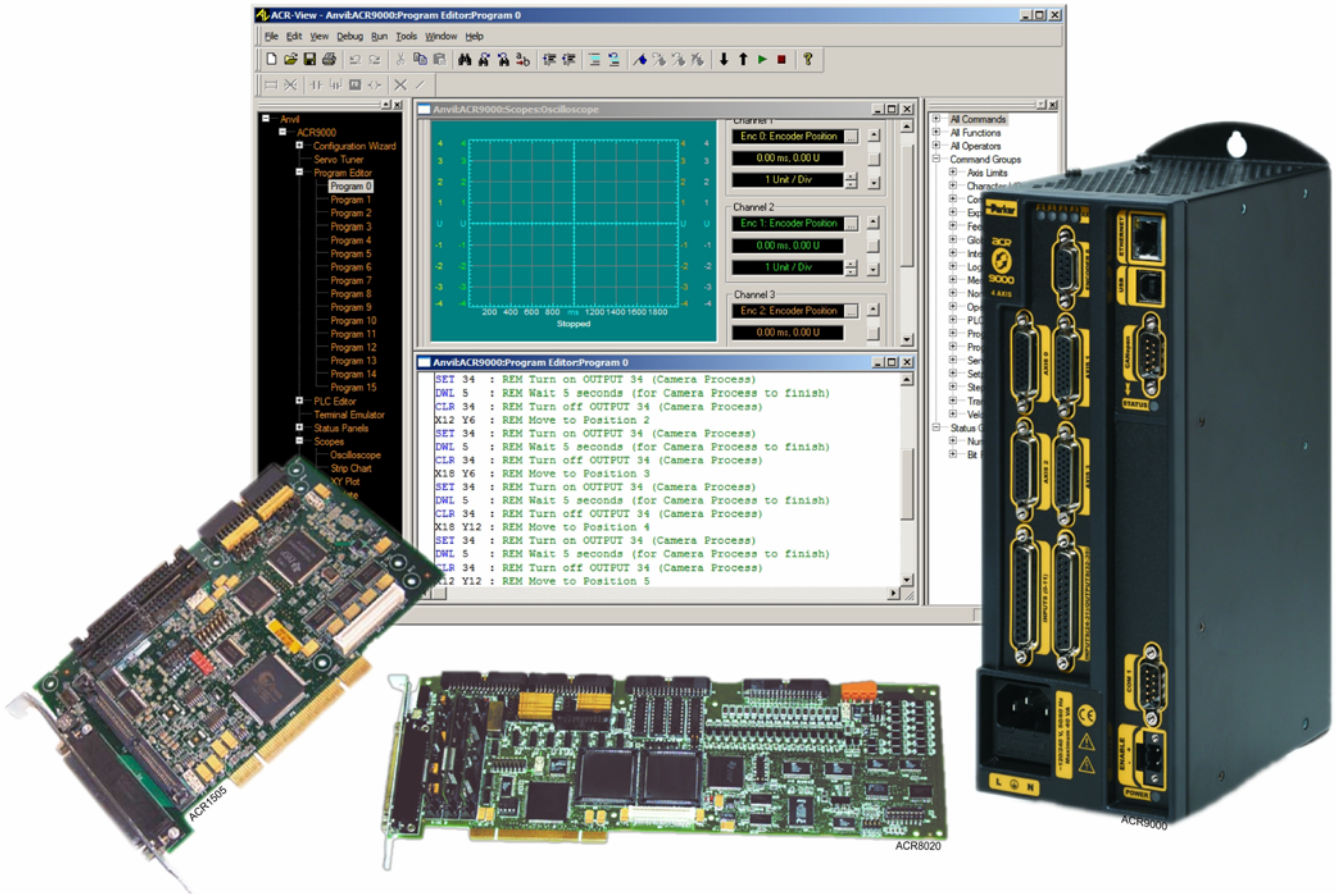
Parker Automation

p/n 88-028039-01A

Motion COMponents .NET User's Guide for ACR Series Products

Effective: July 2006



User Information

Warning — ACR Series products are used to control electrical and mechanical components of motion control systems. You should test your motion system for safety under all potential conditions. Failure to do so can result in damage to equipment and/or serious injury to personnel.

ACR series products and the information in this guide are the proprietary property of Parker Hannifin Corporation or its licensers, and may not be copied, disclosed, or used for any purpose not expressly authorized by the owner thereof.

Since Parker Hannifin constantly strives to improve all of its products, we reserve the right to change this guide, and software and hardware mentioned therein, at any time without notice.

In no event will the provider of the equipment be liable for any incidental, consequential, or special damages of any kind or nature whatsoever, including but not limited to lost profits arising from or in any way connected with the use of the equipment or this guide.

© 2003-2005 Parker Hannifin Corporation
All Rights Reserved

.NET is a trademark of Microsoft Corporation.

Technical Assistance
Contact your local automation technology center (ATC) or distributor.

North America and Asia
Parker Hannifin
5500 Business Park Drive
Rohnert Park, CA 94928
Telephone: (800) 358-9070 or (707) 584-7558
Fax: (707) 584-3793
Email: emn_support@parker.com
Internet: http://www.parkermotion.com

Germany, Austria, Switzerland
Parker Hannifin
Postfach: 77607-1720
Robert-Bosch-Str. 22
D-77656 Offenburg
Telephone: +49 (0) 781 509-0
Fax: +49 (0) 781 509-176
Email: sales.hauser@parker.com
Internet: http://www.parker-emd.com

Europe (non-German speaking)
Parker Hannifin plc
Electromechanical Automation, Europe
Arena Business Centre
Holy Rood Close
Poole
Dorset, UK
BH17 7BA
Telephone: +44 (0) 1202 606300
Fax: +44 (0) 1202 606301
Email: support.digiplan@parker.com
Internet: http://www.parker-emd.com

Italy
Parker Hannifin
20092 Cinisello Balsamo
Milan, Italy via Gounod, 1
Telephone: +39 02 6601 2478
Fax: +39 02 6601 2808
Email: sales.sbc@parker.com
Internet: http://www.parker-emd.com

Parker Automation

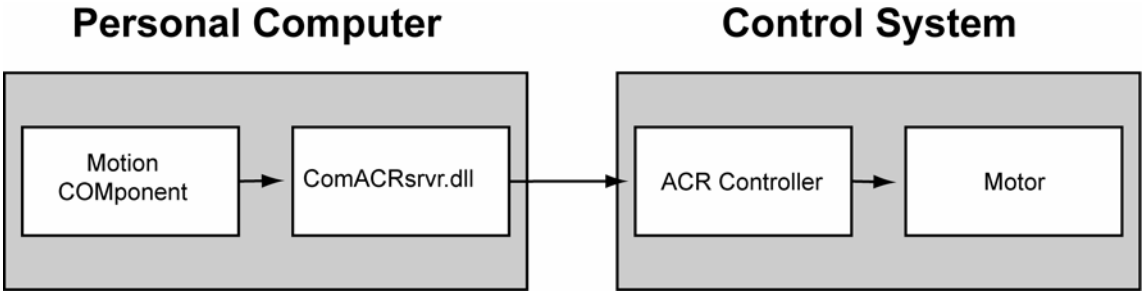
Technical Support E-mail
emn_support@parker.com

Table of Contents

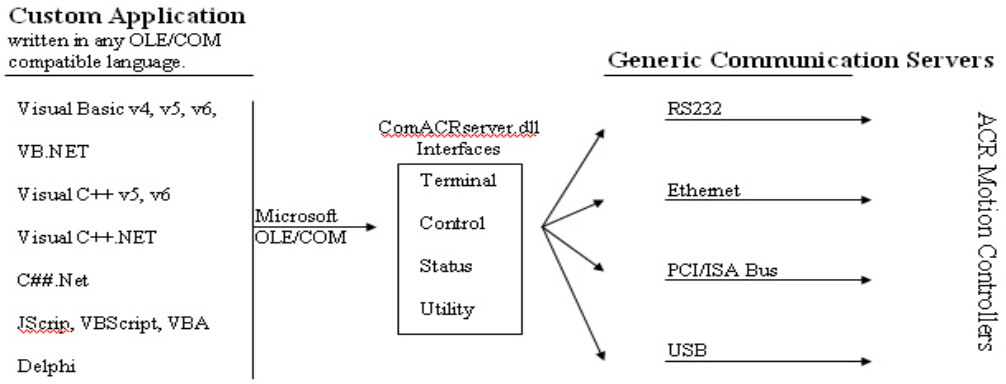
Overview

The .NET version of Motion COMponents for ACR series motion controllers acts as a wrapper for the ComACRsrvr.dll, which you can use with your motion control applications. For more information about the ComACRsrvr.dll, see the

The following illustrates the interaction between a personal computer and control system.



The Motion COMponents let you access the ComACRsrvr.dll using many different Windows based programming environments. The following diagram illustrates this concept.



Connection Control Properties and Methods

The Connection Control is the primary control responsible for creating the communication connection with the ACR controller from the application. It will have a group of properties and methods to facilitate communication.

Properties

- **Bool** **OnConnectTest**
- **String** **ComVersion**
- **Long** **Port**
- **Long** **BPS**
- **Long** **Bus**
- **Long** **Card**
- **String** **IPAddr**
- **Boolean** **IsOffline**
- **Long** **Transport**
- **String** **USBSerialNumber**
- **Long** **nDevice**

Methods

- **Void** **Connect (long nTransport, long nIndex)**
- **Boolean** **TestConnect ()**
- **Void** **SetWatchDog (long nInterval, long nRetries)**
- **Void** **Disconnect ()**

Properties

OnConnectTest

Description	A wrapper Boolean property that determines if the Connection is verified after it is established. If it is true then both Connect method and TestConnect method will be called. If it is False then only the Connect method will be called.
Property	**OnConnectTest**
Return Type	bool
Range	N/A
Default	True
Example	MotionComponents.ACRConnectionControl Connection1 = new MotionComponents.ACRConnectionControl (); Connection1.OnConnectTest=true;

ComVersion

Description A wrapper read only string that holds the version number of the comACRSrvr.dll file.

Property **ComVersion**
Return Type string
Range N/A
Default N/A
Example

```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();
string ver;
ver = Connection1.ComVersion;
```

Port

Description A wrapper long used to set the communications port of the computer to which the serial ACR device is connected. Only used for RS232 Serial connections.

Property **Port**
Return Type long
Range 1 – 256
Default 1
Example

```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();
Connection1.Port = 1; //For COM1 port
```

BPS

Description A wrapper long used to set the speed of the serial port in Bits per Second for the serial port specified by Port. Only used for RS232 Serial communication.

Property **BPS**
Return Type long
Range 9600, 19200, 38400
Default 38400
Example

```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();
Connection1.BPS = 38400;
```

Bus

Description A wrapper long is indicating the type of Bus Card being used. A Value of 0 indicates an ACR PCI Bus Card and a value of 1 indicates an ACR ISA Bus Card. Only used for Bus communication.

Property **Bus**
Return Type long
Range 0-1
Default 0 (PCI)
Example

```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();
Connection1.Bus=0;          //For PCI Bus.
```

Card

Description A wrapper long to set the index number of the ACR Controller. Only used for Bus communication.

Property **Card**
Return Type long
Range N/A
Default 0
Example

```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();
Connection1.Card = 1;
```

IPAddr

Description A wrapper string representing the IP address of an ACR Controller. Only used for Ethernet communication.

Property **IPAddr**
Return Type string
Range N/A
Default 192.168.10.40
Example

```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();
Connection1.IPAddr = ”172.20.22.10”;
```

IsOffline

Description A wrapper read only Boolean indicating whether the connection is Offline. It will set to false once the connection is successful.

Property **IsOffline**
Return Type boo
Range N/A
Default TRUE
Example
```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();
bool isoff = Connection1.IsOffline;
```

Transport

Description A long indicating the physical communication layer being used.

Property **Transport**
Return Type long
Range 0-4
Default N/A
Example
```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();
Connection1.Transport=3;      //Ethernet connection
```

USBSerialNumber

Description A string holding the serial number of the USB port.

Property **USBSerialNumber**
Return Type string
Range N/A
Default N/A
Example
```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();
String usbSNo;
usbSNo = Connection1.USBSerialNumber;
```

---

nDevice

Description Specifies the ACR model.

Property **nDevice**
Return Type Long
Range 0-8
Default 8
Example
```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();
Long lngnDevice = ACRConnection1.nDevice;
```

Methods

Connect

Description Establish a connection of type transport to an ACR Controller.

Signature **Connect** (long nTransport, long nIndex)

Return Type N/A

Parameters

nTransport Indicates the physical communication layer being used, or no layer when offline. (0-Offline, 1- Bus, 2- Serial, 3-Ethernet, 4-USB)

nIndex Transport type dependant data

All Interfaces initially come up with transport = Offline. Each transport type has its own data requirements for connecting.

Transport	**Connection Requirements**
Offline	The nIndex value can be any value.
Bus	The nIndex value must be the card index assigned during installation. To find this card number see parameter P7041, or DIP switch setting on some cards. The Bus property must be set to ISA or PCI depending on the card type.
Serial	The nIndex value is the index of the card, which is typically zero. In a daisy chain configuration, this number identifies the specific controller. The Port must be set to the PC communications port that will be used, and the BPS must be set to the desired bits per second rate.
Ethernet	The nIndex value can be any value. The IPAddrproperty must be set to the IP address of the ACR controller.
USB	The nIndex value is the unique Serial ID of the ACR device. If this is set to zero, the first ACR USB device found will be connected. Any transport specific properties (i.e. IPAddr for Ethernet, etc.) should be set prior to calling Connect method.

Example

```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();

Connection1.IPAddr=”192.9.200.58”;
Connection1.Transport=3;
Connection1.Connect (Connection1.Transport, 0);
```

TestConnect

Description Verifies that an ACR Controller is connected.

Signature **TestConnect** ()

Return Type bool

Parameters N/A

Return A command is sent and the return value verified. If this process succeeds, an ACR's presence is presumed and TRUE is returned. Otherwise FALSE is returned. When the transport type = Offline, this method always returns FALSE.

Example

```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();

bool tstConnect;
tstConnect=Connection1.TestConnect( );
```

SetWatchDog

Description Modifies the Watchdog values.

Signature **SetWatchDog** (long nInterval, long nRetries)

Return Type N/A

Parameters

nInterval — The time in milliseconds, between sending test keep-alive strings to the ACR device.

nRetries — The number of times the keep-alive test string message is sent to the ACR device, with no valid reply, before attempting to reconnect to the ACR device.

Return The Ethernet transport currently has Watchdog functionality. The ACR controller uses a separate port to receive a coded command string (keep-alive message), and echoes it back to the sender. If the Communications Server fails to get a response to a keep-alive message in nInterval*nRetries milliseconds, the Communications Server attempts to reconnect

Example

```
MotionComponents.ACRConnectionControl Connection1 = new
MotionComponents.ACRConnectionControl ();

long interval;
long retries;
interval=2000;
retries=4;
Connection1.SetWatchDog (interval, retries);
```

Disconnect

Description	A wrapper method is used to disconnect the current communication transport.
Signature	**Disconnect** ()
Return Type	N/A
Parameters	N/A
Return	Implicitly calls Connect (0, 0) to switch to Offline mode.
Example	MotionComponents.ACRConnectionControl Connection1 = new MotionComponents.ACRConnectionControl (); Connection1.Disconnect ();

Terminal Control Properties, Methods and Events

The Terminal Control is dual pane multi-line edit control. The right pane is used as an editor and the left pane is used as a terminal where it displays the output of the ACR Controller as well as inputting and sending command to the ACR Controller. There are also buttons below the editor and terminal used to download and upload programs, downloading OS etc.

Terminal Control contains the following Properties and Methods.

Properties

- **int** **DataWaitRate**
- **Color** **TerminalBackcolor**
- **Color** **TerminalForecolor**
- **Color** **EditorBackcolor**
- **Color** **EditorForecolor**
- **User Control** **Connectioncontrol**

Methods

- **string** **DataRead ()**
- **void** **DataWrite (string send)**
- **void** **DownloadFile (string bstrPrg, string bstrFile)**
- **void** **DownloadOS (int nDevice, string bstrFile)**
- **int** **GetDownloadStatus (out int nTotal, out int nBytes)**
- **void** **UploadFile (string bstrPrg, string bstrFile)**
- **void** **StopDownload ()**

Events

- **void** **Datawaiting ()**

Properties

DataWaitRate

Description This property is a wrapper property setting the minimum time between status alerts in milliseconds. The default value is 50 ms.

Property **DataWaitRate**
Return Type int
Range N/A
Default 50 ms
Example

```
MotionComponents.ACRTerminalcontrol Terminalcontrol1 = new
MotionComponents.ACRTerminalcontrol ();

Terminalcontrol1.DataWaitRate=100;
```

TerminalBackColor

Description This property is used to set the Back Color of Terminal Window.

Property **TerminalBackColor**
Return Type Color
Range N/A
Default Blue
Example

```
MotionComponents.ACRTerminalcontrol Terminalcontrol1 = new
MotionComponents.ACRTerminalcontrol ();

Terminalcontrol1.TerminalBackColor = Color.SteelBlue;
```

TerminalForeColor

Description This property is used to set the Fore Color of Terminal Window.

Property **TerminalForeColor**
Return Type Color
Range N/A
Default White
Example

```
MotionComponents.ACRTerminalcontrol Terminalcontrol1 = new
MotionComponents.ACRTerminalcontrol ();

Terminalcontrol1.TerminalForeColor= Color.White;
```

EditorBackColor

Description	This property is used to set the Back Color of Terminal Window.
Property	**EditorBackColor**
Return Type	Color
Range	N/A
Default	White
Example	MotionComponents.ACRTerminalcontrol Terminalcontrol1 = new MotionComponents.ACRTerminalcontrol (); Terminalcontrol1.EditorBackColor = Color.White;

EditorForeColor

Description	This property is used to set the Fore Color of Editor Window
Property	**EditorForeColor**
Return Type	Color
Range	N/A
Default	Black
Example	MotionComponents.ACRTerminalcontrol Terminalcontrol1 = new MotionComponents.ACRTerminalcontrol (); Terminalcontrol1.EditorForeColor = Color.black;

Connectioncontrol

Description	This property will hold the reference of the instance of connection control.
Property	**ConnectionControl**
Return Type	Connection Control
Range	N/A
Default	N/A
Example	MotionComponents.ACRTerminalcontrol Terminalcontrol1 = new MotionComponents.ACRTerminalcontrol (); Terminalcontrol1.Connectioncontrol = this.Connectioncontrol;

Methods

DataRead

Description A wrapper method used to get any ASCII data from the ACR controller. This method wraps the method of Read available in the Terminal interface of ComACRsrvr.dll. This method is used to read the Data available in the buffer of ACR Controller. This method is mainly used to display the output of ACR Controller in the left-pane of Terminal controller.

Signature **DataRead** ()

Return Type string

Parameters N/A

Return The output of the controller will be stored in a string variable.

Example

```
string outputData;

MotionComponents.ACRTerminalcontrol Terminalcontrol1 = new
MotionComponents.ACRTerminalcontrol ();

outputData = Terminalcontrol1.DataRead ();
```

DataWrite

Description This method wraps the method of Write available in the Terminal interface of ComACRsrvr.dll. This method is used to send the command to the ACR Controller.

Signature **DataWrite** (string send)

Return Type N/A

Parameters

send The data used to send to the ACR Controller.

Return N/A

Example

```
String InputData;

MotionComponents.ACRTerminalcontrol Terminalcontrol1 = new
MotionComponents.ACRTerminalcontrol ();
```

Example 1:

```
InputData  = “AXIS0 JOG FWD X”;
Terminalcontrol1.DataWrite (InputData);
```

Example 2:

```
InputData  = “VER”;
Terminalcontrol1.DataWrite (InputData);
```

DownloadFile

Description This method wraps the method available in the Terminalcontrol1.Downloadfile. This method is used to transfer a .8K file to the controller.

Signature **DownloadFile** (string bstrPrg, string bstrFile)

Return Type N/A

Parameters

bstrPrg	holds the Program location in Controller (Optional Parameter) either Blank or Program Location.
bstrFile	holds the path of the downloaded text file.

Return N/A

Example

```
MotionComponents.ACRTerminalcontrol Terminalcontrol1=new
MotionComponents.ACRTerminalcontrol ();
```

Example 1:

```
String filePath = “C:\DownloadData\PROG0.8K”
Terminalcontrol1.DownloadFile (“”, filePath);
```

Example 2:

```
Terminalcontrol1.DownloadFile (“PROG1”, filePath);
```

DownloadOS

Description This method wraps the method of DownloadOS available in the Utility interface of ComACRsrvr.dll. This method is used to download the new Operating system to the controller.

Signature **DownloadOS** (int ndevice, string bstrfile)

Return Type N/A

Parameters

nDevice	Specifies the type of ACR Controller Modal.
bstrFile	Specifies the fully qualified file name of the new Operating System.

Return N/A

Example

```
MotionComponents.ACRTerminalcontrol Terminalcontrol1=new
MotionComponents.ACRTerminalcontrol ();

int nDevice = 8;
String filePath = “C:\DownloadData\NEWOS.OPS”;
Terminalcontrol1.DownloadOS (nDevice, filePath);
```

GetDownloadStatus

Description This method wraps the method of GetDownloadStatus available in the Utility interface of ComACRsrvr.dll. This method is used to get the Current Status of Active download.

Signature **GetDownloadStatus** (out int nTotal, out int nBytes)

Return Type int

Parameters

nTotal	Specifies the Total number of bytes to be transferred.
nBytes	Specifies the Total number of bytes transferred so far.

Return Return value is the status of the active Download.

Example

```
MotionComponents.ACRTerminalcontrol Terminalcontrol1=new
MotionComponents.ACRTerminalcontrol ();

int i =0;
int j = 0;
int k;

k = Terminalcontrol1.GetDownloadStatus (out i ,out j)

MessageBox.show (“Total Bytes to be transferred so far:” +i. ToString());

MessageBox.show (“Total number of bytes transferred so far:”
                                                    +j. ToString ());

MessageBox.show (“Current Active Status Download” + k. ToString ());
```

Uploadfile

Description This method wraps the method of “Uploadfile” available in the Utility interface of ComACRsrvr.dll. This method used to the specified program from the controller to specified .8k file available in the System.

Signature **UploadFile** (string bstrPrg, string bstrFile)

Return Type N/A

Parameters

bstrPrg	Specifies the location to which files are uploaded.
bstrFile	Specifies the fully qualified path and name of the file to download.

Return N/A

Example

```
MotionComponents.ACRTerminalcontrol Terminalcontrol1=new
MotionComponents.ACRTerminalcontrol ();
```

Example 1:

```
String filePath = “C:\UploadData\PROG0.8k”;
Terminalcontrol1.UploadFile (“PROG 0”, filePath);
```

Example 2:

```
String filePath = “C:\UploadData\PROG2.8k”;
Terminalcontrol1.UploadFile (“PROG 1”, filePath);
```

StopDownload

Description This method wraps the method of “StopDownload” available in the Utility interface of ComACRsrvr.dll. This method aborts the file transfer to the Controller.

Signature **StopDownload** ()

Return Type N/A

Parameters N/A

Return N/A

Example

```
MotionComponents.ACRTerminalcontrol Terminalcontrol1 = new
MotionComponents.ACRTerminalcontrol ();

Terminalcontrol1.StopDownload ();
```

Events

DataWaiting

Description	A wrapper callback method is fired whenever the Data available in the Buffer of ComACRsrvr.dll. Using this event the terminal emulator displays the result.
Signature	**void DataWaiting** ()
Return Type	N/A
Parameters	N/A
Return	N/A

Example	Terminalcontrol1.DataWaiting +=new MotionComponents.ACRTerminalcontrol.Datawaiting (Terminalcontrol1_DataWaiting);
	
	private void Terminalcontrol1_DataWaiting ()
	{
	//
	}

Bit Status Control Properties, Methods and Events

The Bit Status Control will be an array of up to 32-bit indicators with labels. The number of bit indicators is dependent on the value of the property BitMask. Bit Data are retrieved in 32-bit blocks. The Bitlabels will constantly update the values of the bits with a color for TRUE and a different color for FALSE.

BitStatus Control contains the following Properties, Methods, and Events.

Properties

- **long** **BitSelect**
- **long** **BitPlacement**
- **long** **Pollrate**
- **Color** **TrueColor**
- **Color** **FalseColor**
- **long** **BitMask**
- **string** **BitMaskCSV**
- **bool** **AutoSize**
- **usercontrol** **ConnectionControl**

Methods

- **void** **BitLabel (string strBitName)**
- **void** **SetBit ()**
- **void** **ClearBit ()**
- **bool** **GetValue ()**
- **bool** **IsFlagSet (int nFlgGrp, int nFlgNdx)**

Events

- **void** **DataChanged (**int msgID, int error)

Properties

BitSelect

Description Represents the user-selected bit number.

Property **BitSelect**
Return Type long
Range N/A
Default N/A
Example

```
MotionComponents.ACRBitStatusControl BitStatus1=new
MotionComponents.ACRBitStatusControl ();
BitStatus1.BitSelect = 550
```

BitPlacement

Description Represents the position of the bit within the 32-bit response. Range 0-31. This is read only property.

Property **BitPlacement**
Return Type long
Range N/A
Default N/A
Example

```
long placement;
MotionComponents.ACRBitStatusControl BitStatus1=new
MotionComponents.ACRBitStatusControl ();

placement = BitStatus1.BitPlacement;
//If BitSelect=34 then result is 2
```

Pollrate

Description Refreshing of bits with ACR Controller will take place based on the Pollrate. Default is set to 100 ms.

Property **Pollrate**
Return Type long
Range N/A
Default 10 ms
Example

```
MotionComponents.ACRBitStatusControl BitStatus1=new
MotionComponents.ACRBitStatusControl ();

BitStatus1.PollRate = 100;   //100 milliseconds
```

TrueColor

Description Color data type has to be set for this property. Default color is Green. If the bit is set to true then the LED color is TrueColor.

Property **TrueColor**
Return Type Color
Range N/A
Default Green
Example

```
MotionComponents.ACRBitStatusControl BitStatus1=new
MotionComponents.ACRBitStatusControl ();
private System.Windows.Forms.ColorDialog cdTrueColor ;
if(cdTrueColor.ShowDialog () == DialogResult OK)
{
        If Bit set, LED color is Green
        BitStatus1.TrueColor = cdTrueColor.Color;
}
```

FalseColor

Description Color datatype has to be set for this property. If the bit is set to false then the LED color is FalseColor

Property **FalseColor**
Return Type Color
Range N/A
Default Grey
Example

```
MotionComponents.ACRBitStatusControl BitStatus1=new
MotionComponents.ACRBitStatusControl ();
private System.Windows.Forms.ColorDialog cdFalseColor;
if(cdFalseColor.ShowDialog() == DialogResult.OK)
{
        //If Bit not set, LED color is Grey
        itStatus1.FalseColor = cdFalseColor.Color;
}
```

BitMask

Description This is a variant value representing the mask of 32 bits. The control will show only the bits that are specified in the mask. Input should be Hexadecimal format.

Property **BitMask**
Return Type long
Range 0 - FFFFFFFF
Default N/A
Example

```
MotionComponents.ACRBitStatusControl BitStatus1=new
MotionComponents.ACRBitStatusControl ();
BitStatus1.BitMask = “4294967295”
//it will show all 32 bits
```

BitMaskCSV

Description This is a comma-separated value of string type. The control will show only the bits that are specified in the BitMaskCSV. Comma separated values can be any value between 0 and 31.

Property **BitMaskCSV**
Return Type string
Range 0-31
Default NA
Example

```
MotionComponents.ACRBitStatusControl BitStatus1=new
MotionComponents.ACRBitStatusControl ();

BitStatus1.BitMaskCSV = “0,1,3”; //it will show bit 0,1, and 3.
```

AutoSize

Description If the Autosize property is true then the size of the control has to be dynamically changed.
This will be based on
(i) Number of bits selected by masking the BitMask property, or
(ii) Number of bits selected in BitMaskCSV property.

Property **AutoSize**
Return Type bool
Range NA
Default TRUE
Example

```
MotionComponents.ACRBitStatusControl BitStatus1=new
MotionComponents.ACRBitStatusControl ();
BitStatus1.AutoSize = true;
```

ConnectionControl

Description This property will hold the reference of the instance of connection control.

Property **ConnectionControl**
Return Type ConnectionControl
Range NA
Default NA
Example

```
MotionComponents.ACRBitStatusControl BitStatus1=new
MotionComponents.ACRBitStatusControl ();
BitStatus1.Connectioncontrol = this.Connectioncontrol;
```

Methods

BitLabel

Description Bit Label has to be captured from the user and has to be assigned to the BitSelect.

Signature **BitLabel** (string strBitName)

Return Type N/A

Parameters

strBitName String value given by user.

Return N/A

Example

```
MotionComponents.ACRBitStatusControl BitStatus1=new
MotionComponents.ACRBitStatusControl ();
BitStatus1.BitSelect = 516;
BitStatus1.BitLabel (“Bit Five One Six”);
```

SetBit

Description This method will enable the corresponding bit of the property BitSelect.

Signature **SetBit** ()

Return Type N/A

Parameters N/A

Return This method will call the SetFlag() method to enable the bit.

Example

```
MotionComponents.ACRBitStatusControl BitStatus1=new
MotionComponents.ACRBitStatusControl ();
BitStatus1.SetBit ();
```

ClearBit

Description This method will disable the corresponding bit of the property BitSelect.

Signature **ClearBit** ()

Return Type N/A

Parameters N/A

Return This method will call the SetFlag() method to disable the bit.

Example MotionComponents.ACRBitStatusControl BitStatus1=new MotionComponents.ACRBitStatusControl ();
BitStatus1.ClearBit();

GetValue

Description This method has to get the status of the bitselect property.

Signature **GetValue** ()

Return Type bool

Parameters N/A

Return Returns TRUE if BitSelect bit is 1, and returns FALSE when the BitSelect bit is 0.

Example MotionComponents.ACRBitStatusControl BitStatus1=new MotionComponents.ACRBitStatusControl ();

bool bitValue;
bitValue = BitStatus1.GetValue ();

IsFlagSet

Description Utility for identifying a bit in a 32-bit long.

Signature **IsFlagSet** (int nFlgGrp, int nFlgNdx)

Return Type bool

Parameters

nFlgGrp	A value of type Long containing flags (as bits.).
nFlgNdx	Index of the flag.

Return Returns TRUE if bit at nFlagNdx is 1, returns FALSE when the bit is 0.

Example

```
Object[] rtnStat;
bool bit128;
rtnStat = GetACRCustom ("P4100");
bit128 = BitStatus1.IsFlagSet (rtnStat(0), 0);
```

Events

DataChanged

Description This event will fire when the data requested is changed.

Signature **DataChanged** (int msgID, int error)

Return Type N/A

Parameters N/A

Return N/A

Example

```
private void BitStatus1_DataChanged (int msgID, int error)
{
}
```

Numeric status Control Properties, Methods and Events

Numeric Status control will get an input parameter from the user (ParameterSelect property has to be used) and it retrieves an array of 8 parameters, which are belongs to same group, from ACR controller.

Numeric status control contains following Properties, Methods and Events.

Properties

- **long** **ParameterSelect**
- **long** **ParameterPlacement**
- **long** **IndexMask**
- **long** **PollRate**
- **bool** **Autosize**
- **string** **IndexMaskCSV**
- **usercontrol** **ConnectionControl**

Methods

- **void** **ParameterLabel (string bstrParamName)**
- **void** **SetLong (int nPparm, int nValue, bool bFast)**
- **void** **SetReal (int nPparm, float fValue, bool bFast)**
- **void** **SetGlobal (int Card, int nGlobal, float dValue, bool bFast)**
- **Object** **GetValue**
- **long** **GetParamType (int nParameter)**
- **bool** **GetParamInfo (int nParameter, out int nType, out int nCode, out int nIndex, out string bstrCatagory, out string bstrDesc)**
- **long** **GetParamAddr (int nParameter)**
- **long** **GetLocalAddr (int nProg, int nType, out int nSize)**
- **long** **GetLocalArrayAddress (int nProg, int nType, int nArray, out int nSize)**
- **Object []** **GetStatus (int nMsgID)**
- **Object** **GetACRMemory (int nType, int nAddress, int nCount)**
- **Object []** **GetACRGroup (string bstrRequest)**
- **Object []** **GetACRCustom (string bstrRequest)**

- **Object GetACRGroupRaw (int nType, int nCode, int nIndex)**
- **void SetACRMemory (int nType, int nAddress, Object Values)**
- **void SetACRMemoryMask (int nAddress, int nNAND, int nOR)**
- **void SetParamLongMask (int nPparm, int nNAND, int nOR)**
- **void InitPerformance ()**
- **void GetPerformance ()**
- **long AddACRGroup (string bstrRequest)**
- **long AddACRGroupRaw (int nType, int nCode, int nIndex)**
- **long AddACRCustom (string bstrRequest)**
- **long AddACRMemory (int nType, int nAddress, int nCount)**
- **void DelStatus (int nMsgID)**

Events

- void **DataChanged (**int msgID, int error**)**

Properties

ParameterSelect

Description This property represents the parameter selected.

Property **ParameterSelect**
Return Type long
Range N/A
Default N/A
Example
```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new MotionComponents.ACRNumericStatusControl ();
NumericStatus1.ParameterSelect = 6916;
```

ParameterPlacement

Description This property represents the index of the array of 8 parameters that the parameter selected by ParameterSelect resides.

Property **ParameterPlacement**
Return Type long
Range N/A
Default N/A
Example
```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new MotionComponents.ACRNumericStatusControl ();
long parmPlacement;
NumericStatus1.ParameterSelect=6916;
parmPlacement =NumericStatus1.ParameterPlacement;
//it will give 5 as result
```

IndexMask

Description This property represents the mask of parameters.

Property **IndexMask**
Return Type long
Range N/A
Default N/A
Example
```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new MotionComponents.ACRNumericStatusControl ();
NumericStatus1.IndexMaskCSV = ”1,2,3”;
```

PollRate

Description This property indicating the period in milliseconds to poll for the status. This value must be set assigned to Connection1. ObjStatus.nStatusWaitRate before calling Connect method. Connection1 is the name of connection control in the form.

Property **PollRate**
Return Type long
Range N/A
Default N/A
Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();
NumericStatus1.PollRate = 100;
```

Autosize

Description If true the control will be automatically resized based on the number of parameters selected with the IndexMask, IndexMaskCSV.

Property **AutoSize**

Return Type bool
Range N/A
Default N/A
Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new MotionComponents.ACRNumericStatusControl ();
NumericStatus1.AutoSize = true;
```

IndexMaskCSV

Description This property represents the mask of parameters with comma separated.

Property **IndexMaskCSV**
Return Type string
Range N/A
Default N/A
Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new MotionComponents.ACRNumericStatusControl ();
NumericStatus1.IndexMaskCSV = ”1,2,3”;
```

ConnectionControl

Description	This property will hold the reference of the instance of connection control.
Property	**ConnectionControl**
Return Type	ConnectionControl
Range	N/A
Default	N/A
Example	MotionComponents.ACRNumericStatusControl NumericStatus1 = new MotionComponents.ACRNumericStatusControl (); NumericStatus1.Connectioncontrol = this.Connectioncontrol;

Methods

ParameterLabel

Description Method used to set the label for a parameter

Signature **ParameterLabel** (string bstrParamName)

Return Type N/A

Parameters

parameterlabel set the label for a parameter

Return N/A

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new  MotionComponents.ACRNumericStatusControl ();
NumericStatus1.ParameterLabel (“P6916”);
```

SetLong

Description Method used to send a long Parameter value to the ACR Controller

Signature **SetLong** (int nPparm, int nValue, bool bFast)

Return Type N/A

Parameters

nPparm Parameter number that is to be changed

nValue Value to assign p-Parameter.

bFast To send the command

Value	Description
TRUE	Binary
FALSE	ASCII

Return N/A

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new MotionComponents.ACRNumericStatusControl ();
int parm=6916;
int nValue=1;
bool bFast=true;
NumericStatus1.SetLong (parm, nValue, bFast);
```

SetReal

Description Method used to send a Real Parameter value to the ACR Controller

Signature **SetReal** (int nPparm, float fValue, bool bFast)

Return Type N/A

Parameters

nPparm Parameter number that is to be changed.

fValue To assign p-Parameter.

bFast To send the command:

Value	Description
TRUE	Binary
FALSE	ASCII

Return N/A

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();

int parm=13370;
int nValue=1.0;
bool bFast=true;

NumericStatus1.SetReal (parm, nValue, bFast);
```

SetGlobal

Description Method used to send a Real Global variable to the ACR Controller.

Signature **SetGlobal** (int nCard, int nGlobal, float dValue, bool bFast)

Return Type N/A

Parameters

ncard value for type of card. This information is needed if using a binary command (bFast=TRUE) to find the memory address. Use zero if using ASCII (bFast=FALSE).

Device Types	
Value	**Controller**
0	ACR1200
1	ACR1500
2	ACR2000
3	ACR8010
4	ACR8020
5	ACR8020 (16-axis)
6	ACR1505
7	Reserved
8	ACR9000

nGlobal p-Parameter number that is to be changed

dValue To assign p-Parameter.

bfast To send the command:

Value	Description
TRUE	Binary
FALSE	ASCII

Return N/A

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();
NumericStatus1.SetGlobal (0, 4095, 100, false);
```

GetValue

Description Method used to get the parameter value.

Signature **GetValue ()**

Return Type Object

Parameters **N/A**

Return Returns the selected numeric parameter status.

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();
NumericStatus1.GetValue ();
```

GetParamType

Description Method used to get the parameter value.

Signature **GetParamType** (int nParameter)

Return Type long

Parameters

nParameter A numeric p-Parameter number

Return Return the data type of the p-Parameter:

Value	Description
0	Int
1	Float
2	Float

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new MotionComponents.ACRNumericStatusControl ();
NumericStatus1.GetParamType (6916);
```

GetParamInfo

Description A wrapper method used to get specific information on a parameter.

Signature **GetParamInfo** (int nParameter, out int nType, out int nCode, out int nIndex, out string bstrCategory, out string bstrDesc)

Return Type bool

Parameters

nParameter A numeric p-Parameter.

nType The data type of the values being read:

Value	Description
0	Int
1	Float
2	Float

nCode The ACR Group Code as documented in the ACR-View online help.

nIndex The ACR Group Index as documented in the ACR-View online help.

bstrCategory A textual description of the category a p-Parameter is in.

bstrDesc A textual description of the p-Parameter.

Return Return TRUE if p-Parameter found.

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new  MotionComponents.ACRNumericStatusControl ();

int i;
int j;
int k;
string catgry;
string catgrydesc;
NumericStatus1.GetParamInfo (4096, out i, out j, out k, out catgry, out
catgrydesc);
```

GetParamAddr

Description A wrapper method used to retrieve the memory address location of a parameter

Signature **GetParamAddr** (int nParameter)

Return Type long

Parameters

nParameter A numeric p-Parameter.

Return Return the address of the p-Parameter.

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();

NumericStatus1.GetParamAddr (6916);
```

GetLocalAddr

Description A wrapper method used to retrieve the memory address of a local variable in a specific program

Signature **GetLocalAddr** (int nProg, int nType, out int nSize)

Return Type long

Parameters

Nprog Provide the program number—local variables are dimensioned in a program space.

NType The data type of the values being read:

Value	Description
0	Int
1	Float
2	Float

NSize After the call, this parameter holds the number of dimensioned variables available.

Return The return value is a valid ACR memory address (or zero if no memory is dimensioned for the requested variable type.)

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new MotionComponents.ACRNumericStatusControl ();
int i;
NumericStatus1.GetLocalAddr (0, 0, out i);
```

GetLocalArrayAddress

Description A wrapper method used to retrieve the memory address if a local variable array in a specific program.

Signature **GetLocalArrayAddress** (int nProg, int nType, int nArray, out int nSize)

Return Type long

Parameters

nProg Provide the program number—local variables are dimensioned in a program space.

nType The data type of the values being read:

Value	Description
0	Int
1	Float
2	Float

nArray The specific array being looked for.

nSize After the call, this parameter holds the number of dimensioned variables available.

Return The return value is a valid ACR memory address (or zero if no memory is dimensioned for the requested variable type.)

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();
int k;
NumericStatus1.GetLocalArrayAddress (0, 0, 0, out k);
```

GetStatus

Description A wrapper method used to retrieve the specific status information.

Signature **GetStatus** (int nMsgID)

Return Type object[]

Parameters

nMsgID — The key to a specific status request as returned by one of the Add routines.

Return The returned array can be any size. It holds the values in Variants, either type long or float.

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new MotionComponents.ACRNumericStatusControl ();
int msgid;
msgid = numericStatusControl1.AddACRGroup (cpar);
NumericStatus1.GetStatus (msgid);
```

GetACRMemory

Description A wrapper method used to retrieve values requested from the specific memory location.

Signature **GetACRMemory** (int nType, int nAddress, int nCount)

Return Type object[]

Parameters

nType — The data type of the values being read:

Value	Description
0	Int
1	Float
2	Float

nAddress — The starting physical memory address.

nCount — The number of values to read (starting at the memory location.) The values of each memory location will be placed in a corresponding position in the returned array.

Return Return the object value.

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new MotionComponents.ACRNumericStatusControl ();
NumericStatus1.GetACRMemory (0,0,5);
```

GetACRGroup

Description A wrapper method used to get the requested parameter group.

Signature **GetACRGroup** (string bstrRequest)

Return Type object[]

Parameters

bstrRequest — String of up to 4 p-Parameters, comma delimited. These parameters are used to look up the group, which is then used to return the 8 p-Parameter values for each group. Any p-Parameter in a group can be used to identify a group. Up to 4 groups can be requested and any undocumented / reserved items in a group are returned as zero (for example P6144 would return 8 values starting with the encoder position for Axis0).

Return Return an array containing up to 32 Variants, each of which are of type long or float. Each p-Parameter in the request results in a group of 8 values of the same type

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();
NumericStatus1.GetACRGroup (6916);
```

GetACRCustom

Description A wrapper method used to get the requested Parameters.

Signature **GetACRCustom** (string bstrRequest)

Return Type object[]

Parameters

bstrRequest — String of up to 32 p-Parameters, comma delimited. These parameters are used to look up the individual, or custom, p- Parameter values (for example P6144, P6160 would return the encoder positions for Axis0 and Axis1).

Return The **GetACRCustom** method Return an array of up to 32 Variants (return type: long or float). Each p-Parameter in the request Return the values of the type as defined in the Parameters Reference section of the ACR User's Guide-View online help.

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();
NumericStatus1.GetACRCustom ("P6916");
```

GetACRGroupRaw

Description A wrapper method used to get the requested parameter group from the index and code.

Signature **GetACRGroupRaw** (int nType, int nCode, int nIndex)

Return Type Object

Parameters

nType The data type of the values being read:

Value	Description
0	Int
1	Float
2	Float

nCode The ACR Group Code as documented in the ACR-View online help.

nIndex The ACR Group Index as documented in the ACR-View online help.

Return Return an array containing up to 8 Variants, all of which are of type long or float.

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();
NumericStatus1.GetACRGroupRaw (6916);
```

SetACRMemory

Description A wrapper method used to changes the value at a specific memory location

Signature **SetACRMemory** (int nType, int nAddress, object Values)

Return Type N/A

Parameters

nType The data type of the values being read:

Value	Description
0	Int
1	Float

2	Float

nAddress — The starting physical memory address on the ACR product values the data to be placed in memory starting at the address.

Values — The data to be placed in memory starting at the address.

Return N/A

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();
object[] values1={10,20,30};
NumericStatus1.SetACRMemory (0, 0, values1);
```

SetACRMemoryMask

Description A wrapper method used to change the bit value(s) of a specific memory location.

Signature **SetACRMemoryMask** (int nAddress, int nNAND, int nOR)

Return Type N/A

Parameters

nAddress — The starting physical memory address.This address must point to a variable of type long for the mask to properly work.nNAND Used to clear bits.nOR Used to set bits.

nNAND — Used to clear bits.

nOR — Used to Set bits.

Return N/A

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();
NumericStatus1.SetACRMemoryMask (0, 10, 5);
```

SetParamLongMask

Description A wrapper method used to change the long value through bitwise operations.

Signature **SetParamLongMask** (int nPparm, int nNAND, int nOR)

Return Type N/A

Parameters

nPparm	The parameter on the ACR product. This address must point to a variable of type long for the mask to properly work.nNAND Used to clear bits.nOR Used to set bits.
nNAND	Used to clear bits.
nOR	Used to Set bits.
Return	N/A
Example	MotionComponents.ACRNumericStatusControl NumericStatus1 = new MotionComponents.ACRNumericStatusControl (); NumericStatus1.SetParamLongMask (6916, 10, 5);

InitPerformance

Description	A wrapper method used to change the long value through bitwise operations.
Signature	**InitPerformance()**
Return Type	N/A
Parameters	N/A
Return	N/A
Example	MotionComponents.ACRNumericStatusControl NumericStatus1 = new MotionComponents.ACRNumericStatusControl (); NumericStatus1.InitPerformance ();

GetPerformance

Description	A wrapper method used to retrieve the performance data of the ISA Cards.
Signature	**GetPerformance ()**
Return Type	N/A
Parameters	N/A
Return	N/A
Example	MotionComponents.ACRNumericStatusControl NumericStatus1 = new MotionComponents.ACRNumericStatusControl (); NumericStatus1.GetPerformance ();

AddACRGroup

Description Method to add a group request into the status queue.

Signature **AddACRGroup** (string bstrRequest)

Return Type long

Parameters

bstrRrequest — String of up to 4 p-Parameters, comma delimited. These parameters are used to look up the group, which is then used to return the 8 p-Parameter values for each group. Any p-Parameter in a group can be used to identify a group. Up to 4 groups can be requested and any undocumented/reserved items in a group are returned as zero (for example P6144 would return 8 values starting with the encoder position for Axis0).

Return A key identifying the request in the queue. The key can be used to retrieve data using **GetStatus()** (for example, when the alert is signaled).

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents.ACRNumericStatusControl ();
NumericStatus1.AddACRGroup ("P6916");
```

AddACRGroupRaw

Description Method to Add a group request into the status queue.

Signature **AddACRGroupRaw** (int nType, int nCode, int nIndex)

Return Type long

Parameters

NType — The data type of the values being read:

Value	Description
0	Int
1	Float
2	Float

NCode — The ACR Group Code as documented in the ACR-View online help.

NIndex — The ACR Group Index as documented in the ACR-View online help.

Return A key identifying the request in the queue. The key can be used to retrieve data using **GetStatus()** (for example, when the alert is signaled).

Example MotionComponents.ACRNumericStatusControl NumericStatus1 = new MotionComponents.ACRNumericStatusControl ();
NumericStatus1.AddACRGroupRaw (0,0,0);

AddACRCustom

Description Add a custom p-Parameter request into the status queue.

Signature **AddACRCustom** (string bstrRequest)

Return Type long

Parameters

BstrRequest — String of up to 32 p-Parameters, comma delimited. These parameters are used to look up individual or custom p-Parameter values (for example P6144, P6160 would return the encoder positions for Axis0 and Axis1).

Return A key identifying the request in the queue. The key can be used to retrieve data using **GetStatus()** (for example, when the alert is signaled).

Example MotionComponents.ACRNumericStatusControl NumericStatus1 = new MotionComponents.ACRNumericStatusControl ();
NumericStatus1.AddACRCustom (“P6916”);

AddACRMemory

Description Add a memory value request into the status queue.

Signature **AddACRMemory** (int nType, int nAddress, int nCount)

Return Type long

Parameters

NType — The data type of the values being read:

Value	Description
0	Int
1	Float
2	Float

NAddress — The starting physical memory address on the ACR product.

NCount — The number of values to read (starting at the memory location.) The values of each memory location will be placed in a corresponding position in the returned array.

Return A key identifying the request in the queue. The key can be used to retrieve data using **GetStatus()** (for example, when the alert is signaled).

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 =
new MotionComponents.ACRNumericStatusControl ();
NumericStatus1.AddACRMemory (0, 0, 2);
```

DelStatus

Description Delete a status request from the status queue.

Signature **DelStatus** (int nMsgID)

Return Type N/A

Parameters

NMsgID — The key to a specific status request as returned by one of the Add routines.

Return N/A

Example

```
MotionComponents.ACRNumericStatusControl NumericStatus1 = new
MotionComponents. ACRNumericStatusControl ();
int msgid;
msgid = numericStatusControl1.AddACRGroup(cpar);
NumericStatus1.DelStatus (msgid);
```

Events

DataChanged

Description	Wrapper callback method to be called when the data is available in the buffer of comACRsrvr.dll. The INumericstatuscontrol.StatusWaiting callback method is to be used.
Signature	**DataChanged** (int msgID, int error)
Return Type	N/A
Parameters	
msgID	holds an integer, which is, correspond to the changed bit.
error	holds an integer that represents the error when error is occurred during the event.
Return	N/A

Example	private void NumericStatus1_DataChanged (int msgID, int error)
	{
	}

Moves Control Properties and Methods

Moves control doesn't have a dialog and primarily will be a wrapper for properties and methods of the ComACRSrvr.dll. With this control we can perform movement in the motor one is linear move and the other is arc move.

Moves Control contains the following Properties and Methods.

Properties

- **int** **MoveProfile**
- **float** **MoveVel**
- **float** **MoveFVel**
- **float** **MoveAcc**
- **int** **MoveMode**
- **bool** **MoveAbsolute**
- **int** **MoveCounter**
- **int** **ArcMode**
- **bool** **ArcAbsolute**
- **bool** **ArcCCW**
- **usercontrol** **ConnectionControl**

Methods

- **void** **Moves (int nmask, Object[] targets)**
- **void** **MoveBatch (int nmask, Object movement)**
- **void** **Arc (int nmask, Object[] targets)**
- **void** **Stop (Bool bDecel)**
- **void** **SendRes (int nmask)**
- **void** **GetMoveCounter (out int nCounter, out int nIncrement)**
- **void** **SetMoveCounter (int nCounter, int nIncrement)**

Properties

MoveProfile

Description	This property specifies the master profile for use to move. Wraps the ComACRsrvr property nMoveProfile.
Property	**MoveProfile**
Return Type	int
Range	0-15
Default	0
Example	MotionComponents.ACRMovesControl MovesControl1=new MotionComponents.ACRMovesControl (); MovesControl1.MoveProfile = 3;

MoveVel

Description This property sets the linear velocity for the next move. Wraps the ComACRsrvr property fMoveVel.

Property **MoveVel**
Return Type float
Range N/A
Default -1 (Use Preset Velocity)
Example
```
MotionComponents.ACRMovesControl MovesControl1 = new
MotionComponents.ACRMovesControl ();
MovesControl1.MoveVel = 10;
```

MoveFVel

Description This property sets the final velocity for the next move. Wraps the ComACRsrvr property fMoveFVel.

Property **MoveFVel**
Return Type float
Range N/A
Default -1 (Use Preset Velocity)
Example
```
MotionComponents.ACRMovesControl MovesControl1 = new
MotionComponents.ACRMovesControl ();
MovesControl1.MoveFVel = 10;
```

MoveAcc

Description This property sets the Acceleration to be used with a linear move. Wraps the ComACRsrvr property fMoveAcc.

Property **MoveAcc**
Return Type float
Range N/A
Default -1 (Use Preset Velocity)
Example
```
MotionComponents.ACRMovesControl MovesControl1=new
MotionComponents.ACRMovesControl ();
MovesControl1.MoveACC = 10;
```

MoveMode

Description This property sets the mode of movement. Wraps the ComACRsrvr property nMoveMode.

Property **MoveMode**
Return Type int
Range N/A
Default 2
Example
```
MotionComponents.ACRMovesControl MovesControl1=new
MotionComponents.ACRMovesControl ();
MovesControl1.MoveMode= 3;
```

MoveAbsolute

Description	This property sets the absolute co-ordinate system or incremental co-ordinate system. Wraps the ComACRsrvr property bMoveAbsolute.
Property	**MoveAbsolute**
Return Type	bool
Range	N/A
Default	TRUE (Absolute Moves)
Example	MotionComponents.ACRMovesControl MovesControl1=new MotionComponents.ACRMovesControl (); MovesControl1.MoveAbsolute = True;

MoveCounter

Description	This property enables the move counter. Wraps the ComACRsrvr property nMoveCounter.
Property	**MoveCounter**
Return Type	int
Range	N/A
Default	1 (ON, Count UP)
Example	MotionComponents.ACRMovesControl MovesControl1=new MotionComponents.ACRMovesControl (); MovesControl1.MoveCounter = 1;

ArcMode

Description	This property determines primary and secondary axes when performing the arc move. Wraps the ComACRsrvr property nArcMode.
Property	**ArcMode**
Return Type	int
Range	0-3
Default	0
Example	MotionComponents.ACRMovesControl MovesControl1=new MotionComponents.ACRMovesControl (); MovesControl1.ArcMode = 1;

ArcAbsolute

Description	This property determines whether the arc centers are treated in Absolute terms or in relative terms. Wraps the ComACRsrvr property bArcAbsolute.
Property	**ArcAbsolute**
Return Type	bool
Range	N/A
Default	TRUE (Arc Absolute Centers)
Example	MotionComponents.ACRMovesControl MovesControl1=new MotionComponents.ACRMovesControl (); MovesControl1.ArcAbsoulte = True;

ArcCCW

Description	This property determines the direction of the arc move. Wraps the ComACRsrvr property bArcCCW.
Property	**ArcCCW**
Return Type	bool
Range	N/A
Default	TRUE (CCW)
Example	MotionComponents.ACRMovesControl MovesControl1=new MotionComponents.ACRMovesControl (); MovesControl1.ArcCCW = True;

Connectioncontrol

Description	This property will hold the reference of the instance of connection control.
Property	**Connectioncontrol**
Return Type	Connectioncontrol
Range	N/A
Default	N/A
Example	MotionComponents.ACRMovesControl MovesControl1=new MotionComponents.ACRMovesControl (); MovesControl1.Connectioncontrol = this.ConnectionControl;

Method

Moves

Description This wrapper method will generate the move.

Signature **Moves** (int nMask, object[] targets)

Return Type N/A

Parameters

nMask	Specifies which axes to use for move.
targets	The target position information for each specified axis.

Return N/A

Example

```
MotionComponents.ACRMovesControl MovesControl1=new
MotionComponents.ACRMovesControl ();
object [] targets = new object [2];
targets[0] = 150;
targets [1] = 150;
MovesControl1.Moves (3,targets);
```

MoveBatch

Description This wrapper method will send a set of fully defined moves for batch processing.

Signature **MoveBatch** (int nmask, object movement)

Return Type N/A

Parameters

nMask	The data type of values being read.
Movement	Data required for completing any number of moves.

Return N/A

Example

```
MotionComponents.ACRTerminalcontrol MovesControl1=new
MotionComponents.ACRMovesControl ();
MYSAFEARRAYLib.IconvertorClass objConvert = new IconvertorClass
                                                    ();
MovesControl1.MoveBatch (0,objConvert. GetArray ());
movement (3) = 3
movement (4) = 0
movement (5) = 0
```

```
movement (6) = 0
movement (7) = 0
movement (8) = 0#
movement (9) = 0#
movement (10) = 0#
movement (11) = 15
movement (12) = 15
movement (13) = 0
movement (14) = 0
movement (15) = 0
movement (16) = 0
movement (17) = 0
movement (18) = 0
movement (19) = 0
movement (20) = 0
movement (21) = 0
movement (22) = 0
movement (23) = 0
movement (24) = 0
movement (25) = 0
movement (26) = 0
movement (27) = 0
movement (28) = 0
movement (29) = 0#
movement (30) = 0#
movement (31) = 0#
MovesControl1.MoveBatch (nMask, fValue)
```

Arc

Description This wrapper method will generate an Arc move.

Signature **Arc** (int nMask, object[] targets)

Return Type N/A

Parameters

nMask	Specifies which axes to use for move.
targets	The arc centers the target position information for each axis.

Return N/A

Example

```
MotionComponents.ACRMovesControl MovesControl1=new
MotionComponents.ACRMovesControl ();
object [] targets = new object [4];
targets [0] = 5;
targets [1] = 0;
targets [2] = 5;
targets [3] = -5;
MovesControl1.Arc (3,targets);
```

Stop

Description This wrapper method stops the commanded motion.

Signature **Stop** (bool bDecel)

Return Type N/A

Parameters

bDecel Determines how to stop the motion.

Value	Description
TRUE	Stop All Moves
FALSE	Kill All Moves

Return When the bDecel parameter is TRUE, a Stop All Moves flag is set using the binary command.
When the bDecel parameter is FALSE, a Kill All Moves flag is set using binary.

Example

```
MotionComponents.ACRMovesControl MovesControl1=new
MotionComponents.ACRMovesControl ();
Bool SendValue = True;
MovesControl1.Stop (SendValue);
```

SendRes

Description This wrapper method resets the position counters of the ACR Controller.

Signature **SendRes** (int nMask)

Return Type N/A

Parameters

nMask Specifies which axes to apply the RES.

Return N/A

Example

```
MotionComponents.ACRMovesControl MovesControl1=new
MotionComponents.ACRMovesControl ();
int nMask;
nMask = 2;
MovesControl1.SendRes (nMask);
```

GetMoveCounter

Description This wrapper method retrieves the move counter from the ACR Controller.

Signature **GetMoveCounter** (out int nCounter, out int nIncrement)

Return Type N/A

Parameters

nCounter	The index value of the move currently active on the controller.
nIncrement	The step used to increment the nCounter.

Return The nCounter and nIncrement are both updated after calling this method.

Example

```
MotionComponents.ACRMovesControl MovesControl1=new
MotionComponents.ACRMovesControl ();
int nCounter = 0;
int nIncrement = 0;
MovesControl1.GetMoveCounter (out nCounter, out nIncrement);
MessageBox.Show (nCounter.ToString() + " " + nIncrement.ToString());
```

SetMoveCounter

Description This wrapper method sets the move counter of the ACR Controller.

Signature **SetMoveCounter** (int nCounter, int nIncrement)

Return Type N/A

Parameters

nCounter	The index value of the move currently active on the controller.
nIncrement	The step used to increment the nCounter.

Return N/A

Example

```
MotionComponents.ACRMovesControl MovesControl1=new
MotionComponents.ACRMovesControl ();
int nCounter = 2;
int nIncrement = 2;
MovesControl1.SetMoveCounter (nCounter, nIncrement);
```

Feedrate Control Properties and Methods

The Feedrate control is used to manipulate the move by either increasing or decreasing the speed of the motor. The motor can also be paused while the motor is in motion and can also be unpaused.

Feedrate Control contains the following Properties and Methods.

Properties

- **int** **MotionProfile**
- **float** **FOV**
- **float** **MinFOV**
- **float** **MaxFOV**
- **usercontrol** **ConnectionControl**

Methods

- **Void** **SetFOV (int nMask, float fValue)**
- **Void** **SetROV (int nMask, float fValue)**

Properties

MotionProfile

Description This property holds the Master number that the control will be using.

Property **MotionProfile**
Return Type int
Range 0 – 7
Default 0
Example
```
private MotionComponents.ACRFeedratecontrol Feedratecontrol1 = new ACRFeedratecontrol ();
Feedratecontrol1.MotionProfile = 0;
```

FOV

Description This property holds the Feedrate override value for the current Master profile. This value will either increase or decrease the speed of the motor. FOV should always be within MinFov and MaxFov.

Property **FOV**
Return Type float
Range N/A
Default 1
Example
```
private MotionComponents.ACRFeedratecontrol Feedratecontrol1 = new ACRFeedratecontrol ();
Feedratecontrol1.FOV = 1.25;
```

MinFOV

Description	This property holds the Minimum Feedrate override value for the current Master profile. This property keeps check on the lower boundary of FOV.
Property	**MinFOV**
Return Type	float
Range	N/A
Default	1
Example	private MotionComponents.ACRFeedratecontrol Feedratecontrol1 = new ACRFeedratecontrol (); Feedratecontrol1.MinFov = 1.0;

MaxFOV

Description	This property holds the Maximum Feedrate override value for the current Master profile. This property keeps check on the upper boundary of FOV.
Property	**MaxFOV**
Return Type	float
Range	N/A
Default	100
Example	private MotionComponents.ACRFeedratecontrol Feedratecontrol1 = new ACRFeedratecontrol (); Feedratecontrol1.MaxFov = 7.50;

ConnectionControl

Description	This property will hold the reference of the instance of connection control.
Property	**Connectioncontrol**
Return Type	Connectioncontrol
Range	N/A
Default	N/A
Example	private MotionComponents.ACRFeedratecontrol Feedratecontrol1 = new ACRFeedratecontrol (); Feedratecontrol1.Connectioncontrol = this.Connectcontrol;

Method

SetFOV

Description This wrapper method will set the Feedrate override value for the current master profile.

Signature **SetFOV** (int nMask, float fValue)

Return Type N/A

Parameters

nMask	Specifies which axes to use for setting the FOV
fValue	Set the FOV value for all the specified axes in nMask.

Return N/A

Example

```
int nMask = 3
float fValue = 1.25
MotionComponents.ACRFeedratecontrol Feedratecontrol1 = new
ACRFeedratecontrol ();
Feedratecontrol1.SetFOV (nMask, fValue);
```

SetROV

Description This wrapper method will set the Rapid Feedrate override value for the current master profile.

Signature **SetROV** (long nMask, double fValue)

Return Type N/A

Parameters

nMask	Specifies which axes to use for setting the ROV
fValue	Set the ROV value for all the specified axes in nMask.

Return N/A

Example

```
int nMask = 3;
float fValue = 1.25;
MotionComponents.ACRFeedratecontrol Feedratecontrol1 = new
ACRFeedratecontrol ();
Feedrate1.SetROV (nMask, fValue);
```

CANopen Control Properties and Methods

The CANopen control is used to configure and start the CANopen I/O Network based on the value of Master Node ID, Bit Rate, Cyclic period and Number of Slave nodes set by the User.

CANopen control contains following Properties, and Methods.

Properties

- **int** **MasterNodeID**
- **int** **BitRate**
- **int** **CyclicPeriod**
- **int** **NumSlaveNodes**
- **UserControl** **ConnectionControl**

Methods

- **Void** **CalcBitRate ()**
- **Void** **CalcCyclicPeriod ()**
- **Void** **SetSlaveNodeID (int[] slavenodes)**
- **int[]** **GetSlaveNodeID ()**
- **Void** **StartCANopen ()**
- **Void** **ResetCANopen ()**
- **Void** **GetCANopenStatus ()**

Properties

MasterNodeID

Description Represents the MasterNodeID of the CANopen network.

Property **MasterNodeId**
Return Type int
Range 1 - 127
Default 5
Example
```
MotionComponents.ACRCANopen CANopen1=new
MotionComponents.ACRCANopen ();
CANopen1.MasterNodeId =5;
```

BitRate

Description Represents the Bit Rate for the CANopen network in kbps.

Property **BitRate**
Return Type int
Range 10,20,50,125,250,500,800,1000
Default 125
Example
```
MotionComponents.ACRCANopen CANopen1=new
MotionComponents.ACRCANopen ();
CANopen1.BitRate =125;
```

CyclicPeriod

Description Represents the Cyclic period of the CANopen network (in ms).

Property **CyclicPeriod**
Return Type int
Range N/A
Default 50
Example
```
MotionComponents.ACRCANopen CANopen1=new
MotionComponents.ACRCANopen ();
CANopen1.CyclicPeriod=50;
```

NumSlaveNodes

Description Holds the value of number of slave nodes on the Network (Range 0- 4).

Property **NumSlaveNodes**
Return Type int
Range 0-4
Default 1
Example
```
MotionComponents.ACRCANopen CANopen1=new
MotionComponents.ACRCANopen ();
CANopen1.NumSlaveNodes =3;
```

ConnectionControl

Description This property will hold the reference of the instance of connection control.

Property **Connectioncontrol**
Return Type userControl
Range N/A
Default N/A
Example
```
MotionComponents.ACRCANopen CANopen1=new
MotionComponents.ACRCANopen ();
CANopen1.ConnectionControl = this.ConnectionControl;
```

Method

CalcBitRate

Description Displays the BitRate Table selection dialog.

Signature **CalcBitRate ()**

Return Type N/A

Parameters N/A

Return N/A

Example

```
MotionComponents.ACRCANopen CANopen1=new
MotionComponents.ACRCANopen ();
CANopen1.CalcBitRate ();
```

CalcCyclicPeriod

Description Displays the CalcCyclicPeriod Table calculator dialog.

Signature **CalcCyclicperiod ()**

Return Type N/A

Parameters N/A

Return N/A

Example

```
MotionComponents.ACRCANopen CANopen1=new
MotionComponents.ACRCANopen ();
CANopen1.CalcCyclicperiod ();
```

SetSlaveNodeID

Description Sends an array of integers that represents the NodeID for each of the Slave Nodes.

Signature **SetSlaveNodeID** (int[] slavenodes)

Return Type N/A

Parameters

Slavenodes	It is an integer array those values are to be set to slave nodes.

Return N/A

Example

```
MotionComponents.ACRCANopen CANopen1=new
MotionComponents.ACRCANopen ();
```

```
int[] slavenodes =new int[]{3, 6, 7, 9};
CANopen1.SetSlaveNodeID (slavenodes);
```

GetSlaveNodeID

Description Retrieves an array of integers that represents NodeID for each of the Slave Nodes.

Signature **GetSlaveNodeID ()**

Return Type Int[]

Parameters N/A

Return This method returns integer array that represents all slave node ids.

Example

```
MotionComponents.ACRCANopen CANopen1=new
MotionComponents.ACRCANopen ();
int[] I=CANopen1.GetSlaveNodeID ();
```

StartCANopen

Description This method activates the CANopen Network.

Signature StartCANopen ()

Return Type N/A

Parameters N/A

Return N/A

Example

```
MotionComponents.ACRCANopen CANopen1=new
MotionComponents.ACRCANopen ();
CANopen1.StartCANopen ();
```

ResetCANopen

Description Method is used to stop and reset the CANopen Network.

Signature **ResetCANopen ()**

Return Type N/A

Parameters N/A

Return N/A

Example	MotionComponents.ACRCANopen CANopen1=new MotionComponents.ACRCANopen (); CANopen1.ResetCANopen ();

GetCANopenStatus

Description	This method is used to request and retrieve the status of the CANopen Network and displays in the Display textbox.
Signature	**GetCANopenStatus ()**
Return Type	N/A
Parameters	N/A
Return	N/A
Example	MotionComponents.ACRCANopen CANopen1=new MotionComponents.ACRCANopen (); CANopen1.GetCANopenStatus ();

TeachPanel Control Properties and Methods

The TeachPanel control is used to play movements in the controller and record the same. Jog Neg and Jog Pos are the 2 different movements, which can be handled, in three different modes. Continuous, Incremental and Whileheld are the three modes. While the drive is rotating we can capture the position and store it in an array. The captured array can be cleared or stored in a CSV file. Through the button named Enable we can enable the drive and with the button named EStop we can stop the motor while in motion.

TeachPanel Control contains the following Properties and Methods.

Properties

- **float** **ActualPos**
- **int** **Axis**
- **bool** **DriveEnable**
- **bool** **DriveFault**
- **bool** **PosLimit**
- **bool** **NegLimit**
- **bool** **Home**
- **int** **TeachArrayIndex**
- **float** **Velocity**
- **float** **Acceleration**
- **float** **TargetPosition**
- **int** **PPU**
- **int** **JogMode**
- **bool** **HomeDirectionPositive**
- **bool** **DisableDriveOnEStop**
- **int** **Pollrate**
- **UserControl** **ConnectionControl**

Methods

- **Void** **JogNeg ()**
- **Void** **JogPos ()**
- **Void** **JogStop ()**
- **Void** **EnableDrive ()**
- **Void** **KillAllMotion ()**
- **Void** **JogHome ()**
- **Void** **ClearPos ()**
- **Void** **CapturePos ()**
- **Void** **ClearArray ()**
- **Void** **SaveArray ()**

Properties

ActualPos

Description This is a read only property, which holds the current position of the motor.

Property **ActualPos**
Return Type float
Range N/A
Default N/A

Axis

Description This property holds the value of the Axis.

Property **Axis**
Return Type int
Range 0-15
Default 0
Example
```
private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl ();
TeachPanelControl1.Axis = 7;
```

DriveEnable

Description This is a read only property, which stores the drive enable status in this property. If the value of this property is true then it represents drive is enabled, and if the value is false it represents drive is disabled.

Property **DriveEnable**
Return Type bool
Range N/A
Default N/A

DriveFault

Description Drive fault will be stored in this read only property. If the value of this property is true then it represents drive is disabled, and if the value is false it represents drive is enabled.

Property **DriveFault**
Return Type bool
Range N/A
Default N/A

PosLimit

Description	Maximum positive limit status will be stored in this read only property. True represents drive has reached positive limit, false represents drive has not yet reached the positive limit.
Property	**PosLimit**
Return Type	bool
Range	N/A
Default	N/A

NegLimit

Description	Maximum negative limit status will be stored in this read only property. True represents drive has reached negative limit, false represents drive has not yet reached the negative limit.
Property	**NegLimit**
Return Type	bool
Range	N/A
Default	N/A

Home

Description	Read only property, which represents the home state of the drive. If true then the position of the drive is in home state.
Property	**Home**
Return Type	bool
Range	N/A
Default	N/A

TeachArrayIndex

Description	This is a read only property which holds the running index of the teach array index. For every click on the CapturePos the index will be increased by one. This will be zero when ClearPos is clicked.
Property	**TeachArrayIndex**
Return Type	int
Range	N/A
Default	0

Velocity

Description	This property is used to make the movement of the motor and its value should be greater than zero.
Property	**Velocity**
Return Type	float
Range	N/A
Default	N/A
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl (); TeachPanelControl1.Velocity = 33.88;

Acceleration

Description	This property increases the speed of the motor and its value should be greater than zero.
Property	**Acceleration**
Return Type	float
Range	N/A
Default	N/A
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl (); TeachPanelControl1.Acceleration = 88.33;

TargetPosition

Description	Property, which makes the jog movement, if JogPos is clicked then the movement will be in positive direction based on the Incremental distance text box. If JogNeg is clicked then the movement will be in negative direction based on the Incremental distance text box.
Property	**TargetPosition**
Return Type	float
Range	N/A
Default	N/A
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl (); TeachPanelControl1.TargetPosition = 20.235;

PPU

Description Pulses Per Unit will be stored in this property.

Property **PPU**
Return Type int
Range N/A
Default N/A
Example
```
private MotionComponents.ACRTeachPanelControl TeachPanelControl1
= new ACRTeachPanelControl ();
TeachPanelControl1.PPU  = 8000;
```

JogMode

Description Mode for the jog operation will be stored in this property.
0 – Continuous
1 – Whileheld
2 – Incremental

Property **JogMode**
Return Type int
Range N/A
Default 0
Example
```
private MotionComponents.ACRTeachPanelControl TeachPanelControl1
= new ACRTeachPanelControl ();
TeachPanelControl1.Jogmode = 1;
```

HomeDirectionPositive

Description Property that determines whether the motion should happen in the positive direction or in negative direction. If the Boolean value is true, then positive direction else negative direction.

Property **HomeDirectionPositive**
Return Type bool
Range N/A
Default TRUE
Example
```
private MotionComponents.ACRTeachPanelControl TeachPanelControl1
= new ACRTeachPanelControl ();
TeachPanelControl1.HomeDirectionPositive = True;
```

DisableDriveOnEStop

Description	Property determines whether the drives should be disabled or not while stopping. If true the drives are disabled while pressing EStop button else drives will not be disabled.
Property	**DisableDriveOnEStop**
Return Type	bool
Range	N/A
Default	TRUE
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl; TeachPanelControl1.DisableDriveonEstop = True;

Pollrate

Description	Delay time between fetching values from the ACR Controller will be stored in this property.
Property	**Pollrate**
Return Type	int
Range	N/A
Default	10
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl (); TeachPanelControl1.Pollrate = 10;

Connectioncontrol

Description	This property will hold the reference of the instance of connection control.
Property	**Connectioncontrol**
Return Type	ConnectionControl
Range	N/A
Default	N/A
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl (); TeachPanelControl1.ConnectionControl = this.ConnectionControl;

Methods

JogNeg

Description This method will make the jog movement in negative direction.

Signature **JogNeg** ()

Return Type N/A

Parameters N/A

Return N/A

Example
```
private MotionComponents.ACRTeachPanelControl TeachPanelControl1
= new ACRTeachPanelControl ();
TeachPanelControl1.JogNeg ();
```

JogPos

Description This method will make the jog movement in positive direction.

Signature **JogPos** ()

Return Type N/A

Parameters N/A

Return N/A

Example
```
private MotionComponents.ACRTeachPanelControl TeachPanelControl1
= new ACRTeachPanelControl ();
TeachPanelControl1.JogPos ();
```

JogStop

Description This method will stop the jog movement.

Signature **JogStop** ()

Return Type N/A

Parameters N/A

Return N/A

Example
```
private MotionComponents.ACRTeachPanelControl TeachPanelControl1
= new ACRTeachPanelControl ();
TeachPanelControl1.JogStop ();
```

EnableDrive

Description	This method will either enable or disable the drive based on the value of the drivestatus string. If the value of Drivestatus string is “On” then the drive is enabled or if the drivestatus string is “Off” then it is disabled.
Signature	**EnableDrive** ()
Return Type	N/A
Parameters	N/A
Return	N/A
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl (); TeachPanelControl1.Enabledrive ();

KillAllMotion

Description	This method will stop the motion of the motor by sending the values Ctrl + Z or Ctrl + X if the caption of the EStop button is “EStop”. If the button caption is “Clear EStop” then Ctrl + Y will be sent to the controller to clear the flags.
Signature	**KillAllMotion** ()
Return Type	N/A
Parameters	N/A
Return	N/A
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl (); TeachPanel1.KillAllMotion ();

JogHome

Description	This method will make the jog movement either in Positive direction or in Negative direction based on the value of the property HomeDirectionPositive. If HomeDirectionPositive is TRUE then the Motor rotates positive direction and vice-versa by sending command using ITerminal Interface of ComACRsrvr.dll.
Signature	**JogHome** ()
Return Type	N/A
Parameters	N/A
Return	N/A
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl (); TeachPanelControl1.HomePos ();

ClearPos

Description	This method will make the AcutalPos property to zero.
Signature	**ClearPos** ()
Return Type	N/A
Parameters	N/A
Return	N/A
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl (); TeachPanelControl1.ClearPos ();

CapturePos

Description	This method will capture the present ActualPos value in an array.
Signature	**CapturePos** ()
Return Type	N/A
Parameters	N/A
Return	N/A
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl (); TeachPanelControl1.CapturePos ();

ClearArray

Description	This method will clear the so far captured positions from the array.
Signature	**ClearArray** ()
Return Type	N/A
Parameters	N/A
Return	N/A
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl; TeachPanelControl1.ClearArray ();

SaveArray

Description	This method will save the so far captured position into a CSV file.
Signature	**SaveArray** ()
Return Type	N/A
Parameters	N/A
Return	N/A
Example	private MotionComponents.ACRTeachPanelControl TeachPanelControl1 = new ACRTeachPanelControl (); TeachPanelControl1.SaveArray ();

PlaybackPanel Control Properties and Methods

The Playback Panel control is used to play various teach point movements and also playback array movements. The movement can be paused and can be resumed. Through this control we can also perform row-by-row movement. This control saves the played points in a CSV file. Events can also be saved against the points in a separate grid. Velocity, Acceleration, Deceleration and StopRamp has to be provided for the movement of the motor.

Playback Panel Control contains the following Properties and Methods.

Properties

- **Long StartingIndex**
- **Long CurrentIndex**
- **Double Velocity**
- **Double Acceleration**
- **Double Deceleration**
- **Double StopRamp**
- **Long MasterProfile**
- **UserControl ConnectionControl**

Methods

- **Void ImportArray (String strCSVFileName)**
- **Void ImportFromTeach (String strCSVFileName)**
- **Void AddEvent ()**
- **Void Playback ()**
- **Void StepPlayback ()**
- **Void SaveArray (String strCSVFileName)**
- **Void PausePlayback ()**
- **Void ResumePlayback ()**
- **Void StopPlayback ()**
- **String GetValue (int lngRow, int lngCol)**
- **Void SetValue (int lngRow, int lngCol, String strValue)**

Properties

StartingIndex

Description	This property holds the starting index value for the StepNext point movement. This property should be greater than zero. This property will be used only for the first time when StepPlayback is clicked.
Property	**StartingIndex**
Return Type	Long
Range	N/A
Default	N/A
Example	MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1 = new ACRPlaybackPanelControl (); PlaybackPanel1.StartingIndex = 3;

CurrentIndex

Description	This read only property holds the index of the current row while playing. This property gets increased one by one while playing.
Property	**CurrentIndex**
Return Type	Long
Range	N/A
Default	0
Example	MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1 = new ACRPlaybackPanelControl (); long lngCurrIndex = 5; PlaybackPanel1.CurrentIndex = lngCurrIndex;

Velocity

Description	Velocity value will be stored in this property. Velocity should be greater than zero for the movement of the motor.
Property	**Velocity**
Return Type	Double
Range	N/A
Default	0
Example	MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1 = new ACRPlaybackPanelControl (); PlaybackPanel1.Velocity = 20;

Acceleration

Description	Acceleration value will be stored in this property. This property increases the speed of the motor.
Property	**Acceleration**
Return Type	Double
Range	N/A
Default	0
Example	MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1 = new ACRPlaybackPanelControl (); PlaybackPanel1.Acceleration = 20;

Deceleration

Description	Deceleration value will be stored in this property. This property decreases the speed of the motor.
Property	**Deceleration**
Return Type	Double
Range	N/A
Default	0
Example	MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1 = new ACRPlaybackPanelControl (); PlaybackPanel1.Deceleration = 20;

StopRamp

Description	StopRamp value will be stored in this property. This property will be used to stop the motor while in motion.
Property	**StopRamp**
Return Type	Double
Range	N/A
Default	0
Example	MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1 = new ACRPlaybackPanelControl (); PlaybackPanel1.StopRamp = 20;

MasterProfile

Description	MasterProfile value will be stored in this property. This represents which master is being used.
Property	**MasterProfile**
Return Type	Long
Range	0 – 15
Default	0
Example	MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1 = new ACRPlaybackPanelControl (); PlaybackPanel1.MasterProfile = 2;

Connectioncontrol

Description	This property will hold the reference of the instance of connection control.
Property	**Connectioncontrol**
Return Type	Object
Range	N/A
Default	N/A
Example	MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1 = new ACRPlaybackPanelControl (); PlaybackPanel1.ConnectionControl = this.ConnectionControl;

Methods

ImportArray

Description This method will load the CSV file into the Grid.

Signature **ImportArray** (String strCSVFilename)

Return Type N/A

Parameters

strCSVFileName Filename of the CSV File.

Return N/A

Example

```
MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1
= new ACRPlaybackPanelControl ();
String strCSVFileName = “c:\Array.CSV”;
PlaybackPanel1.ImportArray (strCSVFileName);
```

ImportFromTeach

Description This method will load the CSV file created through TeachPanel Control.

Signature **ImportFromTeach** (String strCSVFileName)

Return Type N/A

Parameters

strCSVFileName Filename of the CSV File.

Return N/A

Example

```
MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1
= new ACRPlaybackPanelControl ();
String strCSVFileName = “c:\Array.CSV”;
PlaybackPanel1.ImportFromTeach (strCsvFileName);
```

AddEvent

Description This method will add the command string and the added event will be played back.

Signature **AddEvent** ()

Return Type N/A

Parameters	N/A
Return	N/A
Example	MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1 = new ACRPlaybackPanelControl (); PlaybackPanel1.AddEvent ();

Playback

Description	This method will download the file to the controller and execute the playback points.
Signature	**Playback** ()
Return Type	N/A
Parameters	N/A
Return	N/A
Example	MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1 = new ACRPlaybackPanelControl (); PlaybackPanel1.Playback ();

StepPlayback

Description	This method will perform the playback for one step from the current Index.
Signature	**StepPlayback** ()
Return Type	N/A
Parameters	N/A
Return	N/A
Example	MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1 = new ACRPlaybackPanelControl (); PlaybackPanel1.StepPlayback ();

SaveArray

Description This method will save the teach point array and the Event if available.

Signature **SaveArray** (String strCSVFilename)

Return Type N/A

Parameters

strCSVFileName Filename of the CSV file

Return N/A

Example

```
MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1
= new ACRPlaybackPanelControl ();
String strCSVFileName = “c:\Array.csv”;
PlaybackPanel1.SaveArray (strCSVFileName);
```

PausePlayback

Description This method will pause the movement of the motor at the current index.

Signature **PausePlayback** ()

Return Type N/A

Parameters N/A

Return N/A

Example

```
MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1
= new ACRPlaybackPanelControl ();
PlaybackPanel1.PausePlayback ();
```

ResumePlayback

Description This method will resume the movement of the motor from the paused stage.

Signature **ResumePlayback** ()

Return Type N/A

Parameters N/A

Return N/A

Example

```
MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1
= new ACRPlaybackPanelControl ();
PlaybackPanel1.ResumePlayback ();
```

StopPlayback

Description This method will stop the movement of the motor at the current index.

Signature **StopPlayback** ()

Return Type N/A

Parameters N/A

Return N/A

Example

```
MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1
= new ACRPlaybackPanelControl ();
PlaybackPanel1.StopPlayback ();
```

GetValue

Description This function will retrieve the value from the Grid for the specified row and column.

Signature **GetValue** (int lngRow, int lngCol)

Return Type String

Parameters

lngRow	Row number has to be provided.
lngCol	Col number has to be provided.

Return This function returns the value of the cell specified in lngRow and lngCol.

Example

```
MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1
= new ACRPlaybackPanelControl ();
Long lngRow;
Long lngCol ;
String Result;
lngRow = 2;
lngCol = 2;
Result = PlaybackPanel1.GetValue (lngRow, lngCol);
```

SetValue

Description This method will update the grid cell at the specified row and column with the specified value.

Signature **SetValue** (int lngRow, int lngCol, String strValue)

Return Type N/A

Parameters

lngRow	Row number has to be provided.
lngCol	Col number has to be provided.
strValue	Value to be placed in the Grid cell.

Return N/A

Example

```
MotionComponents.ACRPlaybackPanelControl PlaybackPanelControl1
= new ACRPlaybackPanelControl ();
long lngRow;
long lngCol;
String strValue;
lngRow = 2;
lngCol = 2;
strValue = 23.55;
PlaybackPanel1.SetValue (lngRow, lngCol, strValue);
```

StatusPanel Control Properties Methods and Events

The Status Panel Control displays the status of the most commonly used bits and the numeric value of the P-parameters.

Status Panel Control contains following Properties, Methods, and Events.

Properties

- **int** **PollRate**
- **usercontrol** **ConnectionControl**

Methods

- **bool** **GetMasterBitStatus (int Row, int Col)**
- **bool** **GetAxisBitStatus (int Row, int Col)**
- **object** **GetMasterNumericStatus (int Row, int Col)**
- **object** **GetAxisNumericStatus (int Row, int Col)**

Events

- **Void** **DataChanged ()**

Properties

PollRate

Description	This property indicates the period in milliseconds to poll for the status. This value must be set assigned to Connection1.ObjStatus.nStatusWaitRate before calling Connect method.
Property	**PollRate**
Return Type	int
Range	N/A
Default	10
Example	MotionComponents.ACRStatusPanel StatusPanel1=new MotionComponents.ACRStatusPanel (); StatusPanel1.PollRate=100;

ConnectionControl

Description	This property will hold the reference of the instance of connection control.
Property	**ConnectionControl**
Return Type	Connectioncontrol
Range	N/A
Default	N/A
Example	MotionComponents.ACRStatusPanel StatusPanel1=new MotionComponents.ACRStatusPanel (); StatusPanel1.ConnectionControl = this.ConnectionControl;

Methods

GetMasterBitStatus

Description Returns the value from the user interface for a particular label in master bit status display.

Signature **GetMasterBitStatus** (int row, int col)

Return Type bool

Parameters

row — Holds the number of label in master bit status tab.
Range 0-7

0	Moving
1	Accelerating
2	Decelerating
3	Stopping
4	Kill All Moves Request
5	Program Running
6	Program Inhibited
7	Program Dwelling

col — Holds the Master Number (Range 0- 7).

Return The Boolean value for the status of master bit.

Example

```
MotionComponents.ACRStatusPanel StatusPanel1=new
MotionComponents.ACRStatusPanel ();
bool status;

//To get Moving status of Master 0
status=StatusPanel1.GetMasterBitStatus(0, 0);

//To get Moving status of Master 1
status=StatusPanel1.GetMasterBitStatus(0, 1);
```

GetAxisBitStatus

Description Returns the value from the user interface for a particular label in axis bit status display.

Signature **GetAxisBitStatus** (int row, int col)

Return Type bool

Parameters

Row — Holds the number of label in axis bit status tab.
Range 0-9.

0 Drive Enable
1 Drive Fault
2 Kill All Motion Request
3 Jog Active
4 Positive Hard Limit
5 Negative Hard Limit
6 Positive Soft Limit
7 Negative Soft Limit
8 Home Found
9 Maximum Position Error

Col — Holds the Axis Number (Range 0- 7).

Return — The Boolean value for the status of master bit.

Example

```
MotionComponents.ACRStatusPanel StatusPanel1=new
MotionComponents.ACRStatusPanel ();
bool status;
//To get “Drive Enable” status for Axis 0
status=StatusPanel1.GetAxisBitStatus (0, 0);

//To get “Drive Enable” status for Axis 1
status=StatusPanel1.GetAxisBitStatus (0, 1);

//To get “Drive Fault” status for Axis 0
status=StatusPanel1.GetAxisBitStatus (1, 0);

//To get “Drive Fault” status for Axis 1
status=StatusPanel1.GetAxisBitStatus (1, 1);
```

GetMasterNumericStatus

Description — Returns the value from the user interface for a particular label in master numeric status display.

Signature — **GetMasterNumericStatus** (int row, int col)

Return Type — object

Parameters

Row — Holds the number of label in master numeric status tab.
Range 0-4.
0 Vel
1 Acc
2 Distance Into Move
3 Distance To Go
4 Program Line Number

Col — Holds the Master Number (Range 0- 7).

Return This returns the variant value of the corresponding cell from the grid.

Example

```
MotionComponents.ACRStatusPanel StatusPanel1=new
MotionComponents.ACRStatusPanel ();
object values;
//To get “vel” value for axis0
values =StatusPanel1.GetMasterNumericStatus(0,0);
//To get “vel” value for axis1
values =StatusPanel1.GetMasterNumericStatus(0,1);
//To get “Acc” value for axis0
values =StatusPanel1.GetMasterNumericStatus(1,0);
//To get “Acc” value for axis1
values =StatusPanel1.GetMasterNumericStatus(1,1);
```

GetAxisNumericStatus

Description Returns the value from the user interface for a particular label in axis numeric status display.

Signature **GetAxisNumericStatus** (int row, int col)

Return Type object

Parameters

Row — Holds the number of label in axis numeric status tab.
Range 0 - 7.

0	Current Position
1	Target Position
2	Actual Position
3	Jog Offset
4	Jog Vel Setting
5	Jog Vel Current
6	Jog Acc Setting
7	Jog Acc Current

Col — Holds the axis number (Range 0- 7).

Return An Object value for the status of master bit.

Example

```
MotionComponents.ACRStatusPanel StatusPanel1 = new
MotionComponents.ACRStatusPanel ();
object values;
//To get “Current Position” value for Axis 0
values =StatusPanel1.GetAxisNumericStatus(0,0) ;
// To get “Current Position” value for Axis 1
values =StatusPanel1.GetAxisNumericStatus(0,1) ;
//To get “Target Position” value for Axis 0
values =StatusPanel1.GetAxisNumericStatus(1,0) ;
//To “Target Position” value for Axis 1
values =StatusPanel1.GetAxisNumericStatus(1,1) ;
```

Events

DataChanged

Description	This event will fire when the data requested has changed.
Signature	**DataChanged** (int msgID, int error)
Return Type	N/A
Parameters	N/A
Return	N/A
Example	private void StatusPanel1_DataChanged (int msgID, int error) { }

DriveTalk Control Properties and Methods

The DriveTalk Control is the primary mechanism for configuration of DriveTalk communication with DriveTalk enabled drives. This is also the main mechanism for retrieving Drive Status (bits and numeric) from these drives. Also this control allows for sending commands as well as configuration information to these drives.

This control is used to provide the following functionalities:

i) To configure Drive Talk communication with DriveTalk enabled drives.
ii) For retrieving status of the drive from DriveTalk enabled drives.
iii) To send commands and configuration information to DriveTalk enabled drives.

Properties

- **Long** **AxesMask**
- **Bool** **EnableDriveTalk**
- **Long** **DriveDataMask**
- **Long** **DriveTalkMode**
- **UserControl** **ConnectionControl**

Methods

- **Void** **GetDriveDataRequest ()**
- **Void** **GetConfig ()**
- **Void** **SendConfig ()**
- **Void** **GetErrorLog ()**
- **Void** **SendASTFile (String ASTFileName)**

Properties

AxesMask

Description Represents the mask value for number of axes (for Axis 0 –15), which are to be connected to the drive talk.

Property **AxesMask**
Return Type Long
Range 1 - 65535
Default N/A
Example

```
MotionComponents.ACRDriveTalkControl DriveTalk1 = new
MotionComponents.ACRDriveTalkControl ();
DriveTalk1.AxesMask=1;          // Axis 0 to be connected
DriveTalk1.AxesMask=3;          // Axis 0, Axis 1 are to be connected
DriveTalk1.AxesMask=8;          //Axis 3 to be connected
DriveTalk1.AxesMask=65535;      // Axis 0 to 15 are to be connected
```

EnableDriveTalk

Description Represents the enabled / disabled status of the drive talk controller.

Property **EnableDriveTalk**
Return Type Bool
Range N/A
Default FALSE
Example

```
MotionComponents.ACRDriveTalkControl DriveTalk1 = new
MotionComponents.ACRDriveTalkControl ();
DriveTalk1.EnableDriveTalk=True;
```

DriveDataMask

Description DriveDataMask is a 32-bit mask that indicates what data parameters the controller should query the Aries Drive for. The lists of the data parameter types are indicated in Bits10496-10750. So it indicates what information the user is interested in.

Property **DriveDataMask**
Return Type Long
Range N/A
Default N/A
Example

```
MotionComponents.ACRDriveTalkControl DriveTalk1 = new
MotionComponents.ACRDriveTalkControl ();
DriveTalk1.DriveDataMask =128;          //for DriveReset
DriveTalk1.DriveDataMask=1048576;       //for Actual Torque
```

DriveTalkMode

Description Represents the mode of communication with the DriveTalk enabled drives. This property has 3 modes.

1 – Drive Talk
2 – DTalk
3 – TalkTo

Property **DriveTalkMode**
Return Type Long
Range 1 - 3
Default N/A
Example

```
MotionComponents.ACRDriveTalkControl DriveTalk1 = new
MotionComponents.ACRDriveTalkControl ();
DriveTalk1.DriveTalkMode=3;     //‘Mode is set to TalkTo
```

ConnectionControl

Description This property will hold the reference of the instance of connection control.

Property **ConnectionControl**
Return Type Connectioncontrol
Range N/A
Default N/A
Example

```
MotionComponents.ACRDriveTalkControl DriveTalk1 = new
MotionComponents.ACRDriveTalkControl ();
DriveTalk1.ConnectionControl = this.ConnectionControl;
//Connection1 is the name of Connection Control in the
//Current form
```

Methods

GetDriveDataRequest

Description This method used to initiate the controller to query the drive. After the query the data should be available in Parameters P28672-30543. This method will call the IControl.SetParmLong and IControl.SetFlag methods of the ComACRsrvr.dll to change the drive data, based on AxesMask and DriveDataMask properties. This method can be called only if EnableDriveTalk is True and DriveTalkMode is 1.

Signature **GetDriveDataRequest ()**

Return Type N/A

Parameters N/A

Return N/A

Example

```
MotionComponents.ACRDriveTalkControl DriveTalk1 = new
MotionComponents.ACRDriveTalkControl ();
DriveTalk1.AxesMask = 1;    // For Axis 0
DriveTalk1.DriveDataMask = 128;
//The following will assign 128 to parameter P4424
DriveTalk1.GetDriveDataRequest ();
```

GetConfig

Description This method will call the IControl.SetFlag () method of the comACRSrvr.dll to get the drive configuration from the drive. This method will call the SetFlag () method and pass the value for the nBit parameter based on the axis number. (Axis number can be received from AxesMask property). This method can be called only if EnableDriveTalk is True and DriveTalkMode is 1.

nBit for Axis0 is 10498
nBit for Axis1 is 10530
nBit for Axis2 is 10562
nBit for Axis3 is 10594
nBit for Axis4 is 10626
nBit for Axis5 is 10658
nBit for Axis6 is 10690
nBit for Axis7 is 10722
nBit for Axis8 is 10754
nBit for Axis9 is 10786
nBit for Axis10 is 10818
nBit for Axis11 is 10850
nBit for Axis12 is 10882
nBit for Axis13 is 10914

nBit for Axis14 is 10946
nBit for Axis15 is 10978

Signature **GetConfig ()**

Return Type N/A

Parameters N/A

Return N/A

Example

```
MotionComponents.ACRDriveTalkControl DriveTalk1 = new
MotionComponents.ACRDriveTalkControl ();
DriveTalk1.AxesMask=3; // for axis 0 and axis 1
DriveTalk1.GetConfig ();  // it will set bit 10498 and 10530
```

SendConfig

Description This method will call the IControl.SetFlag () method of the comACRSrvr.dll to send the configuration to the drive. This method will call the SetFlag () method and pass the value for the nBit parameter based on the axis number. This method can be called only if EnableDriveTalk is True and DriveTalkMode is 1.

nBit for Axis0 is 10497
nBit for Axis1 is 10529
nBit for Axis2 is 10561
nBit for Axis3 is 10593
nBit for Axis4 is 10625
nBit for Axis5 is 10657
nBit for Axis6 is 10689
nBit for Axis7 is 10721
nBit for Axis8 is 10753
nBit for Axis9 is 10785
nBit for Axis10 is 10817
nBit for Axis11 is 10849
nBit for Axis12 is 10881
nBit for Axis13 is 10913
nBit for Axis14 is 10945
nBit for Axis15 is 10977

Signature **SendConfig ()**

Return Type N/A

Parameters N/A

Return N/A

Example

```
MotionComponents.ACRDriveTalkControl DriveTalk1 = new
MotionComponents.ACRDriveTalkControl ();
DriveTalk1.AxesMask=3; // For axis 0 and axis 1
```

DriveTalk1.SendConfig (); // It will set bit 10497 and 10529

GetErrorLog

Description

This method will call the IControl.SetFlag () method of the comACRSrvr.dll to get the error log from the drive. This GetErrorLog () method should call the SetFlag () method and pass the value for the nBit parameter based on the axis number. This method can be called only if EnableDriveTalk is True and DriveTalkMode is 1.

nBit for Axis0 is 10499
nBit for Axis1 is 10531
nBit for Axis2 is 10563
nBit for Axis3 is 10595
nBit for Axis4 is 10627
nBit for Axis5 is 10659
nBit for Axis6 is 10691
nBit for Axis7 is 10723
nBit for Axis8 is 10755
nBit for Axis9 is 10787
nBit for Axis10 is 10819
nBit for Axis11 is 10851
nBit for Axis12 is 10883
nBit for Axis13 is 10915
nBit for Axis14 is 10947
nBit for Axis15 is 10979

Signature **GetErrorLog ()**

Return Type N/A

Parameters N/A

Return N/A

Example

```
MotionComponents.ACRDriveTalkControl DriveTalk1 = new
MotionComponents.ACRDriveTalkControl ();
DriveTalk1.AxesMask = 3; // for axis 0 and axis 1
DriveTalk1.GetErrorLog (); // It will set bit 10499 and 10531
```

SendASTFile

Description This method calls the IUtility.DownloadFile () to send the config file generated from the Aries Support Tool (.AST file) to the drive. This method can be called only if the property EnableDriveTalk is set to true and DriveTalkMode it is set to 2 or 3.

Signature **SendASTFile** (String strASTFileName)

Return Type N/A

Parameters

strASTFileName	Fully qualified AST file name.

Return N/A

Example

```
MotionComponents.ACRDriveTalkControl DriveTalk1 = new
MotionComponents.ACRDriveTalkControl ();
DriveTalk1.SendASTFile (“Test1.AST”);
```

EStop Control Properties and Methods

This control can be called as Emergency Stop. The control will have the label as “EStop” and “Clear EStop”. EStop will stop the motion of the motor and also it can disable the drive based on the Boolean property “DisableDriveonEStop”. Clear EStop will be in blinking stage and this clears the KAMR bit set and makes the motor ready for motion if the motor is not disabled.

EStop Control contains the following Properties and Methods.

Properties

- **COLOR ClearColor**
- **COLOR StopColor**
- **String ClearLabel**
- **String StopLabel**
- **Bool DisableDriveOnEStop**
- **UserControl ConnectionControl**

Methods

- **Void SendEStop ()**
- **Void ClearEStop ()**

Properties

ClearColor

Description	This property holds the back color of the control and will be displayed only when the control is in “Clear EStop” stage. This will be in blinking stage.
Property	**ClearColor**
Return Type	COLOR
Range	N/A
Default	Green
Example	MotionComponents.ACREStopControl EStop1 = new MotionComponents.ACREStopControl (); EStop1.ClearColor = RGB (75, 75, 75);

StopColor

Description	This property holds the back color of the control and will be displayed only when the control is in “EStop” stage.
Property	**StopColor**
Return Type	COLOR
Range	N/A
Default	Red
Example	MotionComponents.ACREStopControl EStop1 = new MotionComponents.ACREStopControl (); EStop1.StopColor = RGB (75, 75, 75);

ClearLabel

Description	This property holds the text and will be displayed only when the control is in clear EStop stage.
Property	**ClearLabel**
Return Type	String
Range	N/A
Default	Clear EStop
Example	MotionComponents.ACREStopControl EStop1 = new MotionComponents.ACREStopControl (); EStop1.ClearLabel = “Clear EStop”;

StopLabel

Description This property holds the text and will be displayed only when the control is in EStop stage. Default text is “EStop”.

Property **StopLabel**
Return Type String
Range N/A
Default EStop
Example
```
MotionComponents.ACREStopControl EStop1 = new
MotionComponents.ACREStopControl ();
EStop1.StopLabel = “EStop”;
```

DisableDriveOnEStop

Description Property determines whether the drives should be disabled or not while stopping. If true the drives are disabled while pressing EStop button else drives will not be disabled.

Property **DisableDriveOnEStop**
Return Type Bool
Range N/A
Default FALSE
Example
```
MotionComponents.ACREStopControl EStop1=new
MotionComponents.ACREStopControl ();
EStop1.DisableDriveonEstop = True;
```

Connectioncontrol

Description This property will hold the reference of the instance of connection control.

Property **Connectioncontrol**
Return Type Connectioncontrol
Range N/A
Default N/A
Example
```
MotionComponents.ACREStopControl EStop1 = new
MotionComponents.ACREStopControl ();
EStop1.ConnectionControl = this.ConnectionControl;
```

Methods

SendEStop

Description This method will send the “Chr (26)” to the controller, which will stop all the motion of the motors, and the KAMR bit will be set.

Signature **SendEStop ()**

Return Type	N/A
Parameters	N/A
Return	N/A
Example	MotionComponents.ACREStopControl EStop1=new MotionComponents.ACREStopControl (); EStop1.SendEStop ();

ClearEStop

Description	This method will send the “Chr (25)” to the controller, which will clear only the KAMR bit set.
Signature	**ClearEStop ()**
Return Type	N/A
Parameters	N/A
Return	N/A
Example	MotionComponents.ACREStopControl EStop1=new MotionComponents.ACREStopControl (); EStop1.ClearEStop ();