Carlo Gavazzi Industri A/S

Dupline Data Access Manual

DuplineDAManual.doc

22-06-2002

Contents

1 Introduction

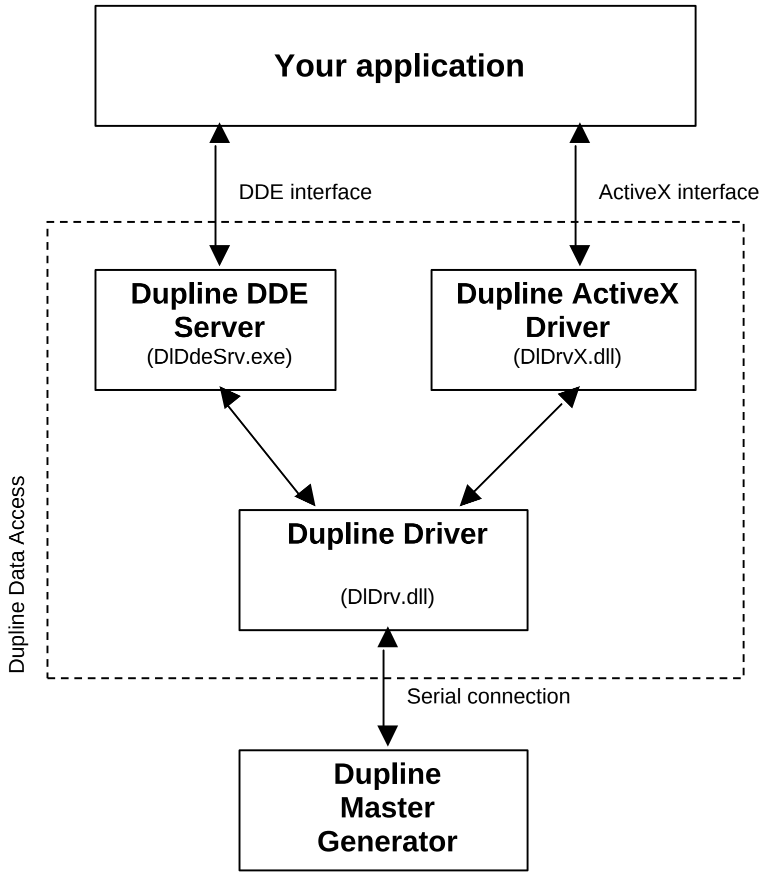
The Dupline Data Acces software package consists of the following two products:

Dupline DDE Server
Dupline ActiveX Driver

The Dupline DDE Server exposes a DDE (Dynamic Data Exchange) interface, which can be used from applications that supports client side DDE such as Microsoft Excel.

The Dupline ActiveX Driver exposes an ActiveX interface (ActiveX = COM = Component Object Model), which is a Microsoft standard for communication between two applications. The Dupline ActiveX driver can be used from any development tool that support client side ActiveX such as Microsoft's Visual Basic and Borland C++ Builder.

The diagram below shows where the Dupline Data Access software fits in the big picture.



2 Dupline DDE Server

2.1 Preface

The Dupline DDE Server makes it easy to develop custom application that communicates with the Dupline Master Generator because it has a user-friendly DDE interface in a Dupline context.

The server is specialized for integration with Microsoft Excel but it can be used with any application that supports DDE (Dynamic Data Exchange).

2.2 Using with Excel

Please refer to the on-line help in the Dupline DDE Server.

2.3 DDE Specifics

2.3.1 Service name

The Dupline DDE Server registers the following DDE service name in the underlying Windows DDE management system:

```
DlDdeSrv
```

2.3.2 Topics and items

The Dupline DDE Server exposes the following four topics:

Digital	topic for communicating with digital Dupline addresses
Counter	topic for communicating with counter Dupline addresses
Analink	topic for communicating with analink Dupline addresses
MuxAnalog	topic for communicating with muxanalog Dupline addresses

Under each of theses topics a number of items are exposes which has the following formats:

Digital	Item format is ‘A1’ -> ‘P8’
Counter	Item format is ‘CNT0-1’ -> ‘CNT31-4’
Analink	Item format is ‘A1’ -> ‘P8’
MuxAnalog	Item format is ‘CD-0’ -> ‘OP-F’

The mentioned items above are all read only i.e. DDE poke is not supported.

2.3.3 Execute

The only items that can be controlled via DDE are the digital and counter items. This is done though the DDE execute mechanism as shown in the Excel VBA code examples below:

2.3.3.1 Digital Excel VBA code

Code for sending a pulse to address A4 on node 1: (this creates the same response as e.g. a pulse from a pushbutton coded for A4- can e.g. be used to toggle a flip-flop channel)

```
Dim ch As Long
ch = Application.DDEInitiate("DlDdeSrv", "Digital")
Application.DDEExecute ch, "[Toggle:1:A4]"
Application.DDETerminate ch
```

Code for continuous activation of A4 on node 2: (this creates the same response as a continuous activation of e.g. a pushbutton coded for A4)

```
Dim ch As Long
ch = Application.DDEInitiate("DlDdeSrv", "Digital")
Application.DDEExecute ch, "[ForceON:2:A4]"
Application.DDETerminate ch
```

Code for removing continuous activation of A4 on node 2:

```
Dim ch As Long
ch = Application.DDEInitiate("DlDdeSrv", "Digital")
Application.DDEExecute ch, "[ForceOFF:2:A4]"
Application.DDETerminate ch
```

2.3.3.2 Counter Excel VBA code

Code for resetting counter CNT22-1 on node 2:

```
Dim ch As Long
ch = Application.DDEInitiate("DlDdeSrv", "Counter")
Application.DDEExecute ch, "[Reset:2:CNT22-1]"
Application.DDETerminate ch
```

3 Dupline ActiveX Driver

3.1 Preface

The Dupline ActiveX Driver makes it very easy to develop custom application that communicates with the Dupline Master Generator because it has a user-friendly API in a Dupline context.

The driver is build on a observer pattern which means that a application tells the driver which Dupline addresses the application is interested in through the Add / Remove methods which are described later. The driver fires events when it for example detects a state change on a digital address. In this way it is not necessary for an application to poll the Dupline Master Generator, it is notified when a change occurs.

3.2 Methods

3.2.1 SetBaudRate

```
HRESULT SetBaudRate([in] BYTE ucPort, [in] long lBaudRate);
```

Set the baud rate for the specified COM port. The call must be made prior to the Open call.

Parameters

ucPort	COM port number
lBaudRate	The baud rate ex. 9600, 57600 etc.

Return value
0 on success.

3.2.2 Open

```
HRESULT Open([in] BYTE ucPort);
```

Opens the specified COM port.

Parameters

ucPort	COM port number

Return value
0 on success.

3.2.3 Close

```
HRESULT Close();
```

Closes the COM port previously opened by a call to the Open method.

Return value
Zero on success.

3.2.4 ToggleDigital

```
HRESULT ToggleDigital([in] BYTE ucNode, [in] BSTR strAddr);
```

Parameters

ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘A1’ -> ‘P8’)

Return value
Zero on success.

3.2.5 ForceDigital

```
HRESULT ForceDigital([in] BYTE ucNode, [in] BSTR strAddr, [in] VARIANT_BOOL bState);
```

Forces a digital address on or off.

Parameters

ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘A1’ -> ‘P8’)
bState	true if the address should be forced on

Return value

Zero on success.

3.2.6 AddDigital

```
HRESULT AddDigital([in] BYTE ucNode, [in] BSTR strAddr);
```

Adds a digital address to the subscription mechanism causing the OnDigital event to fire on a state change.

Parameters

ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘A1’ -> ‘P8’)

Return value

Zero on success.

3.2.7 RemoveDigital

```
HRESULT RemoveDigital([in] BYTE ucNode, [in] BSTR strAddr);
```

Removes a digital address from the subscription mechanism.

Parameters

ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘A1’ -> ‘P8’)

Return value

Zero on success.

3.2.8 AddAnalink

```
HRESULT AddAnalink([in] BYTE ucNode, [in] BSTR strAddr);
```

Adds an analink address to the subscription mechanism causing the OnAnalink event to fire when the analink value changes.

Parameters

ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘A1’ -> ‘P8’)

Return value
Zero on success.

3.2.9 RemoveAnalink

```
HRESULT RemoveAnalink([in] BYTE ucNode, [in] BSTR strAddr);
```

Removes an analink address from the subscription mechanism.

Parameters

ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘A1’ -> ‘P8’)

Return value
Zero on success.

3.2.10 AddMuxAnalog

```
HRESULT AddMuxAnalog([in] BYTE ucNode, [in] BSTR strAddr);
```

Adds a muxanalog address to the subscription mechanism causing the OnMuxAnalog event to fire when the muxanalog value changes.

Parameters

ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘CD-0’ -> ‘OP-F’)

Return value
Zero on success.

3.2.11 RemoveMuxAnalog

```
HRESULT RemoveMuxAnalog([in] BYTE ucNode, [in] BSTR strAddr);
```

Removes a muxanalog address from the subscription mechanism.

Parameters

ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘CD-0’ -> ‘OP-F’)

Return value

Zero on success.

3.2.12 AddCounter

```
HRESULT AddCounter([in] BYTE ucNode, [in] BSTR strAddr);
```

Adds a counter address to the subscription mechanism causing the OnCounter event to fire when the counter value changes.

Parameters

ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘CNT0-1’ -> ‘CNT31-4’)

Return value

Zero on success.

3.2.13 RemoveCounter

```
HRESULT RemoveCounter([in] BYTE ucNode, [in] BSTR strAddr);
```

Removes a counter address from the subscription mechanism.

Parameters

ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘CNT0-1’ -> ‘CNT31-4’)

Return value

Zero on success.

3.2.14 ResetCounter

```
HRESULT ResetCounter([in] BYTE ucNode, [in] BSTR strAddr);
```

Resets the specified counter.

Parameters

ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘CNT0-1’ -> ‘CNT31-4’)

Return value

Zero on success.

3.3 Events

3.3.1 OnDigital

```
HRESULT OnDigital( [in] VARIANT_BOOL bValid, [in] BYTE ucNode,
                   [in] BSTR strAddr, [in] VARIANT_BOOL bState);
```

The OnDigital event is fired when a digital address changes its state. The address must be added to the subscription mechanism with the AddDigital method to receive events. Use the RemoveDigital method to remove the subscription.

An initial event is fired after a call to AddDigital. Furthermore an event is fired if the communication with the Dupline Master Generator fails or regains connection.

Parameters

bValid	true if the communication with the Dupline Master Generator is ok
ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘A1’ -> ‘P8’)
bState	true if the digital address is on

3.3.2 OnAnalink

```
HRESULT OnAnalink( [in] VARIANT_BOOL bValid, [in] BYTE ucNode,
                   [in] BSTR strAddr, [in] BYTE ucRaw,
                   [in] VARIANT_BOOL bAlarm);
```

The OnAnalink event is fired when an analink address changes value. The address must be added to the subscription mechanism with the AddAnalink method to receive events. Use the RemoveAnalink method to remove the subscription.

An initial event is fired after a call to AddAnalink. Furthermore an event is fired if the communication with the Dupline Master Generator fails or regains connection.

Parameters

bValid	true if the communication with the Dupline Master Generator is ok
ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘A1’ -> ‘P8’)
ucRaw	the analink value (0-255)
bAlarm	the analink alarm state, true if the analink alarm is on

3.3.3 OnMuxAnalog

```
HRESULT OnMuxAnalog([in] VARIANT_BOOL bValid, [in] BYTE ucNode,
                    [in] BSTR strAddr, [in] long lValue);
```

The OnMuxAnalog event is fired when a muxanalog address changes value. The address must be added to the subscription mechanism with the AddMuxAnalog method to receive events. Use the RemoveMuxAnalog method to remove the subscription.

An initial event is fired after a call to AddMuxAnalog. Furthermore an event is fired if the communication with the Dupline Master Generator fails or regains connection.

Parameters

bValid	true if the communication with the Dupline Master Generator is ok
ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘A1’ -> ‘P8’)
lValue	the muxanalog value

3.3.4 OnCounter

```
HRESULT OnCounter( [in] VARIANT_BOOL bValid, [in] BYTE ucNode,
                   [in] BSTR strAddr, [in] long lValue);
```

The OnCounter event is fired when a counter address changes value. The address must be added to the subscription mechanism with the AddCounter method to receive events. Use the RemoveCounter method to remove the subscription.

An initial event is fired after a call to AddCounter. Furthermore an event is fired if the communication with the Dupline Master Generator fails or regains connection.

Parameters

bValid	true if the communication with the Dupline Master Generator is ok
ucNode	the Dupline Master Generator node number
strAddr	the Dupline address (‘A1’ -> ‘P8’)
lValue	the counter value

3.4 Error handling

The Dupline ActiveX Driver uses the standard COM IErrorInfo interface for returning errors to the application. In Visual Basic the error information is returned in the Error object which causes VB to report an error as shown in the code below:

```
' Error object helper
Private Sub ReportError()
    Dim strError As String

    strError = "Error:" & vbCrLf & "Number: " & Err.Number & vbCrLf
    strError = strError & "Source: " & Err.Source & vbCrLf
    strError = strError & "Description: " & Err.Description & vbCrLf

    MsgBox strError, vbCritical
End Sub

' Open port handler
Private Sub OpenPort_Click()
    ' install error handler
    On Error GoTo errorhandler

    ' open the specified port
    obj.Open ComPort.ListIndex + 1

    Exit Sub
errorhandler:
    ' on error report it
    ReportError
End Sub
```

For further information see the VB sample installed with the Dupline Data Access software.

3.5 Redistribution

The following files are needed when redistributing the Dupline ActiveX Driver:

DlDrv.dll	C++ native Windows driver
DlDrvX.dll	ActiveX wrapper for the native Windows driver (the Dupline ActiveX Driver)

The DlDrvX.dll must be registered before it can be used. Many installers automatically including the Dupline Data Access installation do this. If you however want to distribute the Dupline ActiveX Driver without creating an installer you can manually register the Dupline ActiveX Driver with the following command from a command prompt:

```
Regsvr32 DlDrvX.dll
```

The DlDrv.dll must be in the OS path or placed in the same sub-dir as the DlDrvX.dll.

3.5.1 Troubleshooting

The Dupline ActiveX Driver depends on the COM (Component Object Model) dll's installed with the Windows OS. If the registration process fails you may need to install a COM update, which is available from Microsoft at the following URLs:

Windows 95	http://www.microsoft.com/com/dcom/dcom95/dcom1_3.asp
Windows 98	http://www.microsoft.com/com/dcom/dcom98/dcom1_3.asp

Under Windows ME and Windows 2000 the COM dll's are up-to-date.
Under Windows NT 4 service pack 4 or later must be installed.

4 System requirements

The Dupline Data Access software can be used with the following versions of Windows:
Windows 95
Windows 98
Windows ME
Windows NT 4, service pack 4 or later
Windows 2000
Windows XP

5 Contact information

For further information please contact:

Carlo Gavazzi Industri A/S
Over Hadstenvej 40
DK-8370 Hadsten
Phn. / fax: +45 89606100 / +45 86981530
www.dupline.com or www.carlogavazzi.dk