

# SLINK.DLL

## APPLICATION NOTES

### 1. GENERAL DESCRIPTION

SLINK.DLL is a Windows 95/NT dynamic library that implements an Slink3/4 serial communication interface with external devices.

The communication parameters, the port settings and the protocol configuration are fully programmable. SLINK.DLL also allows to share the communication port between several Windows applications; which means that more than one application can be run and simultaneously communicate with the external devices.

### 2. OLE INTERFACE

SLINK.DLL is an OLE Automation server. This feature allows all programs that support automation to communicate with devices that support Slink3/4 protocol.

A typical example of these programs are the applications contained in Microsoft Office or object-based languages such as Microsoft Visual Basic.

The DLL implements an object named **Slink** that manages a simple communication with the devices. Through its methods is possible to configure the communication port and the protocol and then to exchange parameters with the devices.

The object interface identifier is "Slink.Slink.1" and has the following characteristics:

<b>Methods:</b>	Init Config EditPar EditRecord ShowMsg Packet
<b>Properties:</b>	Par ParInt ParFloat ParWord ParDWord MsgOn Active Success ErrTimeOut ErrorCode ErrorString Port Address Option Protocol TimeOut

## 2.1 Methods description

### Init

The **Init** method must be invoked to initialize the **Slink** object before performing any communication procedure.

Syntax:

*object.Init [filename]*

*filename* is an optional string that indicates the pathname of the file where all configuration data are to be stored.

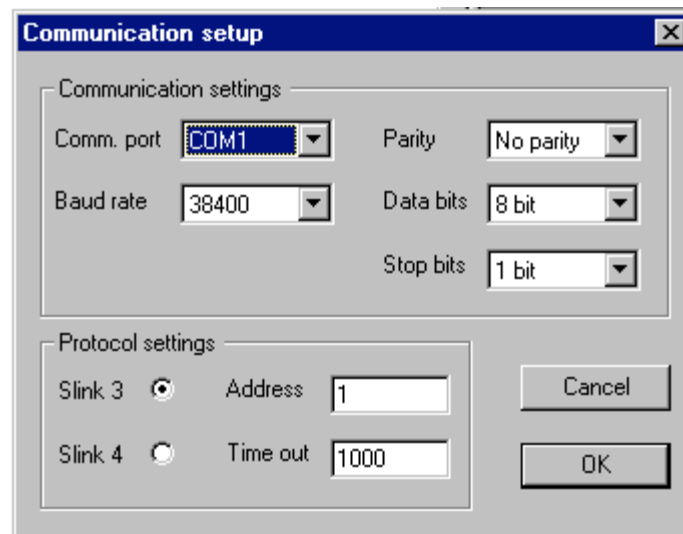
The configuration data are automatically managed by the **Config** method.

### Config

The **Config** method displays the configuration dialog-box shown below. All settings made in the dialog-box are stored in the configuration file (if present) specified with the **Init** method.

Syntax:

*object.Config*



## ShowMsg

The **ShowMsg** method displays a Windows message-box containing the description of the last error occurred during the communication.

Syntax:

*object*.**ShowMsg**

## Packet

The **Packet** method allows to transfer generic data packet to an external device using the Slink3/4 protocol. This command should be used only if the device supports the specific packet format sent using the command itself.

Syntax:

*object*.**Packet** (*CmdId*, *BuffOut*, *LenOut*, *BuffIn*, *LenIn*)

The *CmdId* parameter indicates a command identifier used by the device to identify a specific kind of packet. The *BuffOut* and *LenOut* parameters indicate the data buffer and the length of the packet that is to be sent. The *BuffIn* and *LenIn* parameters indicate the data buffer and the length of the packet that should be received from the device as response of the data sent. A *LenIn* = -1 can be used to indicate an unknown response length.

## 2.2 Properties description

### Par

The **Par** property allows to read or set the value of a parameter by using a VARIANT (see Visual Basic data types) data type.

Syntax: `object.Par (lpa) [= value ]`

The *lpa* parameter indicates the identifier of the parameter.

This functions executes all necessary conversions to obtain a data type compatible with the target device.

Es:

```
Par (701) = 32;    \ Converts 32 in integer and send to device into par 701
Val = Par (604)  \ Reads parameter 604 (that is float) and put the value
                  \ into variable Val as floating point
```

### ParInt

### ParWord

### ParDWord

### ParFloat

The **Parx** properties allows to read or set the value of parameters of a specific data type.

Syntax: `object.Parx (lpa) [= value ]`

The *lpa* parameter indicates the identifier of the parameter.

### MsgOn

**MsgOn** is a BOOLEAN property that enables/disables the automatic issuing of error messages.

Syntax: `object.MsgOn = value`

### Active

**Active** is a BOOLEAN property that indicates if the SLINK interface is active and properly initialized.

Syntax: `object.Active`

### Success

**Success** is a BOOLEAN property that indicates the presence of an error during the last communication command executed.

Syntax: `object.Success`

### ErrorTimeOut

**ErrorTimeOut** is a BOOLEAN property that indicates if the last communication command executed timed out.

Syntax: `object.ErrorTimeOut`

**ErrorCode**

**ErrorCode** is a numeric property that indicates the error code of last communication command executed.

Syntax:        *object*.**ErrorCode**

**ErrorString**

**ErrorString** is a string property that contains a textual description of the error occurred in last communication command executed.

The strings related to each error code are listed in the SLINKDEF.STR file.

Syntax:        *object*. **ErrorString**

**Port  
Address  
Option  
Protocol  
TimeOut**

These properties allows to set and retrieve the communication settings and parameters. These properties refer to the same values displayed by the dialog-box shown by the **Config** method.

Syntax:        *object*.**SettingValue** = *value*

## TYPICAL AUTOMATION MACRO

The following example shows the text of several macros written for Microsoft Excel in its macro language.

```
\
\ Declaration of a variable used for
\ device communication
\

Dim AXV

\
\ This is a macro used to initialize the object Slink
\

Sub Init()

    \ Creation of the object
    Set AXV = CreateObject("Slink.Slink.1")

    \ Initialization of object with configuration file
    \ XLSDEV.CNF
    AXV.Init "XlsDev.cnf"

    \ Activation of configuration dialog-box
    AXV.Config

    \ Enable error messages
    AXV.MsgOn = 1

End Sub

\
\ This macro reads the parameter 601 and puts its value
\ in the active Excel cell.
\ If the communication fails, a default string is stored in the cell
\

Sub GetPar601()

    \ Read the parameter
    par = AXV.ParFloat(601)

    \ Test if the communication failed
    If AXV.Success Then
        \ Cell with the value read
        ActiveCell.Value = par
    Else
        \ Cell with the error value
        ActiveCell.Value = "#####"
    End If

End Sub
```

```
`  
` This macro puts active cell value in the parameter 601 of the device.  
`
```

```
Sub SetPar601()
```

```
    AXV.ParFloat(601) = ActiveCell.Value
```

```
End Sub
```

```
`  
` This macro activate the edit parameter dialog-box.  
` The parameter index is retrieved from the value of the active cell.  
`
```

```
Sub EditPar()
```

```
    AXV.EditPar ActiveCell.Value
```

```
End Sub
```

```
`  
` This macro reads 15 parameters starting from parameter 701 and  
` puts their values in the selected column.  
`
```

```
Sub GetPars()
```

```
    ` Number of the column  
    col = ActiveCell.Column
```

```
    ` Reading loop  
    For i = 1 To 16  
        Cells(i, col).Value = AXV.ParFloat(700 + i)  
    Next
```

```
End Sub
```

## 2.3 FILES NEEDED SLINK INTERFACE

The complete set of files necessary to work with SLINK.DLL is the following:

**SLINK.DLL** is the dynamic link library that features all the Slink API for the devices. It must reside in the Windows system directory or in the application-working directory.

**COMMSERV.EXE** is an application program that manages the serial communications ports. It allows sharing the access to a serial port between several applications.  
It's automatically invoked by SLINK.DLL when the access to a serial port is needed.  
COMMSERV.EXE automatically terminates when no more application programs need to use the serial port it manages.

**SLINKDEF.STR** is the file containing the set of strings used to show the messages and the status of the devices in the dialog-boxes of the OLE interface.

SLINK.DLL also needs that the two MFC (Microsoft Foundation Classes) DLLs: MFC42.DLL and MSVCRT.DLL are available to the system (in the Windows system directories or in the working directory).