Testo Toolbox

testo 300 (version 2018)
testo 320
testo 324
testo 330 (version 2004 and version 2010)
testo 340
testo 350 (version 2011)

Version 1.5.0
18/07/2019

Changes

Version 1.2 (11/14/2014)
First version as Word document

Version 1.3 (03/11/2015)
Support for firing rate in testo 350 added

Version 1.31 (11/13/2015)
Corrections for reading testo 350 protocols

Version 1.32 (11/27/2015)
Function Testo330SetLanguageUS added

Version 1.40 (09/26/2016)
Bluetooth support for testo 340 added

Version 1.41 (08/10/2017)
Function Testo330CloseOnly added

Version 1.42 (13/03/2018)
Support for testo 324 added

Version 1.50 (18/07/2019)
Support for testo 300 (version 2018) added

Functions of the Testo Toolbox

The Testo Toolbox provides simple function to read the current measure values and the stored measure protocols from the supported testo analyzers. In addition, simple control options, such as starting and stopping the pump are supported.

You get the following things with the library

- The Testo Toolbox for supported devices
- The include and library files (*.h, *.lib) for integration into a C/C++ development environment
- The *:DLL files for redistribution

Supported measure analyzers

The following Testo flue gas analyzers are supported

- testo 300 (2018 version)
- testo 320
- testo 324
- testo 330 (2004 version with gray housing and grayscale display)
- testo 330 (2010 version with black housing and color display)
- testo 340
- testo 350 measure box (2011 version)

The testo 330 can be connected to an optional testo 380 particle analyzer.

Only the connection to a single testo 350 measure box is supported. There is no support for systems connected via the Testo bus.

All instruments can be connected via USB to the PC. For this, the installation of the correct USB driver is mandatory. Please install before connecting the analyzer the Testo USB driver from the CD order no. 0501.0152.

If a Bluetooth module is present in the device, a Bluetooth connection can be used from the PC to the analyzer. In this case, the Bluetooth connection has to be established from the Windows system. The pairing PIN is always 1234 for all Testo measure analyzers. It must be set up a serial connection (Bluetooth SPP profile).

Please note that many file names and functions contain the identifier *330.* However, you can use the library for all analyzers referred above. The names are historical, the testo 330 was the first supported measure analyzer.

Supported Operating Systems

The following operating systems were supported

- Microsoft Windows XP (at least SP3), 32 bits
- Microsoft Windows Vista (at least SP2), 32 and 64 bit
- Microsoft Windows 7 (at least SP1), 32 and 64 bit
- Microsoft Windows 8, 8.1, 32 and 64 bit
- Microsoft Windows 10, 32 and 64 bit

Installing the toolbox

For a USB connection, the successful installation of the Testo USB driver is required.

The Testo Toolbox library consists of some. DLL files. These files must be present in a directory, which is found throughout program execution. Use either the directory with your program files or the directory c: \ windows or c: \ windows \ system32. There are the following files:

- Idtostrconverter.dll
- Idtostrconverter_t330.dll
- Idtostrconverter_t324.dll
- Testo3xxE.dll
- Testo3xxT.dll
- Testo330base.dll
- Testoemusb.dll
- TestoRS232.dll
- Testousb.dll

Documentation of function calls

The calls to the Testo Toolbox are included in the file testo330base.dll and are implemented as a classic C interfaces.

In the file testo330base.dll are more interfaces not shown in this document. These are not suitable and tested for toolbox. We must therefore advice against the use of these interfaces.

The toolbox logs to the file *testo330base.txt* in the *%TEMP%* directory the most important steps in communication with the measure analyzer. In case of error important information can be obtained from this file.

Open connection to one analyzer

Testo330Open

This function is used to open the connection between the PC and the instrument. It is checked whether a connection can be established and the analyzer is ready for measuring.

It is automatically recognized which analyzer type is connected. The remaining commands to the analyzer (so as the respective encoder expected) are sent correctly. If it is an analyzer that is not supported the connection is terminated with an error.

For a USB connection the measure analyzer is searched on all USB ports of your PC automatically. It is assumed that just one Testo analyzer is connected. If several analyzers are connected a USB connection cannot be established. If you want to connect to a specific analyzer in the case of multiple connected analyzers via USB please use the functions *Testo330GetNumberOfUsbDevices*, *Testo330GetUsbSerialNumber* and *Testo330OpenWithUSbSerialNumber* instead of *Testo330Open*.

In case of error the function Testo330GetLastError provides additional information why no connection was possible.

The connection has to be closed in any case with the Testo330Close function.

Parameter	Type	Description
uCommType	Integer	Select the desired connection type. If the value is zero, a USB connection is established. If the value is greater than or equal to one, then Bluetooth is trying to open connection using the selected Bluetooth SPP. 1 corresponds to COM1, 2 corresponds to COM2, etc.
bSetSlaveMode	Integer	This parameter determines whether the measuring analyzer is set to slave mode. In slave mode, the meter cannot be operated using the keypad on the device. This avoids conflicts with competing actions. A value of 0 means that the slave mode is not set, the value 1 means that the slave mode is set.
Dummy	Integer	Not used, please set the value to zero
Return value	Integer	Returns a handle to the opened analyzer connection. This handle is used in further calls as the first parameter. If the value is zero, then the connection could not be performed. The reason of the error can be obtained using the function Testo330GetLastError.

Open connection with selection of analyzer

Testo330GetNumberOfUsbDevices

This function checks how many Testo measuring instruments are connected to the USB bus.

Parameter	Type	Description
Return value	Integer	number of Testo measuring devices that are connected via USB

Testo330GetUsbSerialNumber

This function returns the serial number of Testo analyzers connected to the USB bus. The *Testo330GetNumberOfUsbDevices* function must have been called once.

Using these serial numbers you can select the analyzer to which the USB connection should be established.

Parameter	Type	Description
uUsbDevice	Integer	number of the USB device from which the serial number is to be returned (0 , 1, ... , TestoGetNumberOfUsbDevives -1)
Return value	Integer	Serial number of the Testo analyzer

Testo330OpenWithUsbSerialNumber

This function is used to open the connection between the PC and the analyzer.

It will attempt to establish a USB connection to the measuring analyzer with the given serial number. If the meter is not found or it is not ready to measure, so the connection is terminated with an error.

In case of error the function Testo330GetLastError provides additional information why no connection was possible.

The connection must be closed in any case with the Testo330Close function.

Parameter	Type	Description
bSetSlaveMode	Integer	This parameter determines whether the measuring analyzer is set to slave mode. In slave mode, the meter cannot be operated using the keypad on the device. This avoids conflicts with competing actions. A value of 0 means that the slave mode is not set, the value 1 means that the slave mode is set.
uSerialNumber	Integer	Serial number of the analyzer to which the USB connection should be established. This serial number should be a value returned by the previously called function

		Testo330GetUsbSerialNumber
Return value	Integer	Returns a handle to the opened analyzer connection. This handle is used in further calls as the first parameter. If the value is zero, then the connection could not be performed. The reason of the error can be obtained using the function

Testo330SetLanguageUS

This function is used to set the language of the returned strings (channel ID, unit, fuel name) to US English. This function is only supported on the testo 320 / 330 (Version 2010) and 350. This function should be called immediately after opening the connection. This feature is useful if you want to have fixed channel names and units independent of the regional settings of the PC and the instrument.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Testo330Close

This function has to be called in any case as the last function. The function closes the connection to the analyzer. Other functions are then no longer possible.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Testo330CloseOnly

This function Testo330Close clears the slave mode on the analyzer and closes the connection. In case of a connection error (e..g Bluetooth is out of range) please use the function Testo330CloseOnly for closing the connection. This function does not send any data to the analyzer and does not clear the slave mode. This may be not possible if the connection is already broken.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Testo330GetSerialNumber

This function returns the serial number of the connected analyzer.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Serial number of the connected analyzer

Reading real time measure values

Excel330OnlineInit

This function is used as initialization of the online measurement. This function has to be called before any other function Excel330Online… .

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Excel330OnlineNumChannels

This function returns the number of measurement channels that are being returned by an online measurement of the connected analyzer.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Number of measure channels

Excel330OnlineChannelID

This function returns the identifier for an online measurement channel. The identifier consists of the physical unit and the name.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
uChannel	Integer	Number of the measuring channel (0, 1, ..., Excel330OnlineNumChannels-1)
szID	Unicode string (max 32 characters)	Identifier of the channel (unit and name)
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Excel330OnlineRead

This function reads the current measure values from the analyzer. The measured values can then be read with the function Excel330OnlineValue.
To avoid overloading of the analyzer this function should not be called more than once per second.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Excel330OnlineValue

This function returns the current measured value of a measurement channel (physical unit and the name).

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
uChannel	Integer	Number of the measuring channel (0, 1, ..., Excel330OnlineNumChannels-1)
szID	Unicode string (max 32 characters)	Current value as text
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Testo330StartZeroing

This function starts a zeroing phase of the analyzer. In a zeroing phase, fresh air is introduced into the sensors. There is a zero calibration, that is, the measuring analyzer is at the end of this phase on the assumption that all the gas concentration is zero and the oxygen level is 21%.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Testo330IsZeroing

The duration of the zeroing phase is determined automatically by the meter. This function can be used to query whether the zeroing phase is still active. Only after finishing the zeroing phase the online measurement can be started with valid measure values.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Returns whether the zeroing phase is still running. If the value is zero, the zeroing phase is completed, the value is nonzero the zeroing phase is still running

Testo330StartPump

This function starts the measuring pump. The pumps transport the gas to be measured by the sensors. After a short ramp-up phase until the gas reached the sensors the measure values are a valid.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Returns whether the zeroing phase is still running. If the value is zero, the zeroing phase is completed, the value is nonzero the zeroing phase is still running

Testo330StopPump

This function stops the measuring pump. After that the online readings are no longer updated, that means that another calling of the Excel330OnlineRead function will not alter the readings.
At the end of a measurement, the pump should be stopped as soon as possible, because the pump has a limited life time.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Returns whether the zeroing phase is still running. If the value is zero, the zeroing phase is completed, the value is nonzero the zeroing phase is still running

Testo330GetLastError

This function returns additional hints for the last occurred error.

Parameter	Type	Description
Return value	ASCII string	Returns an additional text. Is currently only supported by the function Testo330Open

Testo330GetMeasStatus

This function returns the current state of the analyzer, so whether it is ready for measurement or possibly is just in a different state.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	State of the analyzer, see the following table

Measuring state	Description
0	Not initialized
1	Wait state
2	Measurement
3	Zeroing
4	Rinse
5	Zeroing & Startup
6	Dead time
7	Stabilization
8	Measurement program Zeroing
9	Measurement program Ramp-Up
10	Measurement program Measuring
11	Measurement program Rinse
12	Measurement program End

The testo 330 (version 2004) provides as states only the values 2 and 3 (measurement and zeroing).

Reading measure values from stored protocols

The Testo analyzers can store measure protocols in its memory. You can read these measure protocols with the toolbox.

Typically you get first an overview of the protocols stored in the analyzers. This list is shown to the user. The user will select one of these protocols. From this protocol the complete data (measure values) are read from the analyzer to the PC.

Testo330GetAllProtocols

This function is used to read the complete overview of the stored protocols from the analyzer into the PC. This function has to be called as the first function for reading stored measure protocols.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Testo330GetNumProtocols

This function is used to get the number of measure protocols in the analyzer. The function Testo330GetAllProtocols has to be called before this function.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Testo330GetProtocolInfo

This function is used to read the overview information of one protocol. The function Testo330GetAllProtocols has to be called before this function.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
iProtocol	Integer	Number of protocol from which the overview information should be returned (0,1,…,Teso330GetNumProtocols()-1)
szFolder	Char*	Buffer of at least 32 Bytes which holds on return the folder name

		of the protocol
szLocation	Char*	Buffer of at least 32 Bytes which holds on return the location name of the protocol
uYear	Integer	On return: year of starting time (2011, 2012,…)
uMonth	Integer	On return: month of starting time (1,…,12)
uDay	Integer	On return: day of starting time (1,…,31)
uHour	Integer	On return: hour of starting time (0,…,23)
uMinute	Integer	On return: minute of starting time (0,…,59)
uSecond	Integer	On return: second of starting time (0,…,59)
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Testo330ReadProtocol

This function is used to read the complete data of one protocol from the analyzer to the PC. This function is typically called if the user selected one of the protocols.

If you select a protocol with a lot of measure data this function will take some time.

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
iProtocol	Integer	Number of protocol from which the complete data should be read (0,1,…,Teso330GetNumProtocols()-1
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Testo330GetNumColumns

This function is used to get the number of different measure values in the current protocol. The function Testo330ReadProtocol has to be called before this function
… .

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Number of columns (different measure values)

Testo330GetNumRows

This function is used to get the number of measurements in the current protocol. The function Testo330ReadProtocol has to be called before this function

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
Return value	Integer	Number of rows (measurements at different time stamps)

Testo330GetColumnName

This function is used to get the unit and identifier of one measure value (column). The function Testo330ReadProtocol has to be called before this function

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
iColumn	Integer	Number of column (0,1,…,Testo330GetNumColumns()-1)
szName	Char*	Buffer of at least 32 Bytes which holds on return the name of the column as ASCII string. Unit and identifier are ended with a $ for easy separation in Excel.
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Testo330GetValue

This function is used to get one measure value. The function Testo330ReadProtocol has to be called before this function

Parameter	Type	Description
uHandle	Integer	Handle to the connection opened by Testo330Open or Testo330OpenWithUsbSerialNumber
iColumn	Integer	Number of column (0,1,…,Testo330GetNumColumns()-1)
iRow	Integer	Number of row (0,1,…,Testo330GetNumRows()-1)
szValue	Char*	Buffer of at least 32 Bytes which hold on return the measure value as string
Return value	Integer	Error code. If the value is zero the function code succeeded without error otherwise an error occurred, see section error codes.

Visual Basic (Excel) example

```
Option Explicit

Dim w As Worksheet
Dim hTesto330 As Long
```

```
' Functions in testo330base.dll

Private Declare Function Testo330Open Lib "testo330base.dll" (ByVal bIrDA As
Integer, ByVal bSlaveMode As Integer, ByVal bDummy As Integer) As Long
Private Declare Function Testo330Close Lib "testo330base.dll" (ByVal hTesto330 As
Long) As Long

Private Declare Function Testo330GetAllProtocols Lib "testo330base.dll" (ByVal
hTesto330 As Long) As Long
Private Declare Function Testo330GetNumProtocols Lib "testo330base.dll" (ByVal
hTesto330 As Long) As Long
Private Declare Function Testo330GetProtocolInfo Lib "testo330base.dll" (ByVal
hTesto330 As Long, ByVal uProt As Long, _
    ByVal szFolder As String, ByVal szName As String, _
    ByRef uYear As Long, ByRef uMonth As Long, ByRef uDay As Long, _
    ByRef uHour As Long, ByRef uMin As Long, ByRef uSec As Long) As Long
Private Declare Function Testo330ReadProtocol Lib "testo330base.dll" (ByVal
hTesto330 As Long, ByVal uProt As Long) As Long
Private Declare Function Testo330GetNumColumns Lib "testo330base.dll" (ByVal
hTesto330 As Long) As Long
Private Declare Function Testo330GetNumRows Lib "testo330base.dll" (ByVal
hTesto330 As Long) As Long
Private Declare Function Testo330GetColumnName Lib "testo330base.dll" (ByVal
hTesto330 As Long, ByVal iCol As Long, ByVal szName As String) As Long
Private Declare Function Testo330GetValue Lib "testo330base.dll" (ByVal hTesto330
As Long, ByVal iCol As Long, ByVal iRow As Long, ByVal szValue As String) As Long

' Start reading saved measure protocols

Public Sub ReadProtocolsTesto330()

    Dim err As Long
    Dim NumProt As Long
    Dim i As Long
    Dim j As Long
    Dim cRows As Long
    Dim cCols As Long
    Dim iIndex As Long
    Dim szValue As String
    Dim szFolder As String
    Dim szName As String
    Dim szItem As String
    Dim uYear As Long
    Dim uMonth As Long
    Dim uDay As Long
    Dim uHour As Long
    Dim uMin As Long
    Dim uSec As Long
    Dim protDay As Date
    Dim protTime As Date
    Dim w As Worksheet

    ' open connection to testo 330/320/335 instrument
        hTesto330 = Testo330Open(0,1,0)
    If (hTesto330 = 0) Then
        ' error
```

```
        MsgBox "No testo 330/320/335 measure instrument found."
        Exit Sub
    End If

    err = Testo330GetAllProtocols(hTesto330)
    NumProt = Testo330GetNumProtocols(hTesto330)

    protocolForm.listProtocols.Clear
    For i = 0 To NumProt - 1
        szName = Space$(16)
        szFolder = Space$(16)
        err = Testo330GetProtocolInfo(hTesto330, i, szFolder, szName, uYear,
uMonth, uDay, uHour, uMin, uSec)
        protDay = DateSerial(uYear, uMonth, uDay)
        protTime = TimeSerial(uHour, uMin, uSec)

        ' Folder and Location names are ended with $ to find real end of string
        For j = 1 To Len(szFolder)
            If Mid(szFolder, j, 1) = "$" Then
                szFolder = Mid(szFolder, 1, j - 1)
                GoTo LabelA
            End If
        Next j
LabelA:

        For j = 1 To Len(szName)
            If Mid(szName, j, 1) = "$" Then
                szName = Mid(szName, 1, j - 1)
                GoTo LabelB
            End If
        Next j

LabelB:
        If Len(szFolder) > 0 Then
            ' testo 335 with folders
            szItem = FormatDateTime(protDay, vbShortDate) & " " &
FormatDateTime(protTime, vbLongTime) & " " & Trim$(szFolder) & "\" & Trim$(szName)
        Else
            ' testo 330/320 without folders
            szItem = FormatDateTime(protDay, vbShortDate) & " " &
FormatDateTime(protTime, vbLongTime) & " " & Trim$(szName)
        End If
        protocolForm.listProtocols.AddItem szItem
    Next i

    protocolForm.SetIndex -1
    protocolForm.Show
    iIndex = protocolForm.GetIndex()
    If iIndex >= 0 Then
        ' Read the selected protocol from the instrument
        Set w = ActiveSheet
        w.Cells(1, 1).Select
        w.UsedRange.Clear

        w.Range("A1:AZ1").Interior.Color = RGB(255, 243, 189)

        err = Testo330ReadProtocol(hTesto330, iIndex)
```

```
        cCols = Testo330GetNumColumns(hTesto330)
        cRows = Testo330GetNumRows(hTesto330)

        ' get identifiers of columns
        For i = 0 To cCols - 1
            szValue = Space$(32)
            err = Testo330GetColumnName(hTesto330, i, szValue)
            w.Cells(1, i + 1) = szValue
        Next i

        ' get all measure values
        For i = 0 To cRows - 1
            For j = 0 To cCols - 1
                szValue = Space$(32)
                err = Testo330GetValue(hTesto330, j, i, szValue)
                w.Cells(i + 2, j + 1) = szValue
            Next j
        Next i

    End If

    Testo330Close hTesto330

End Sub
```

Error Codes

Most functions return as return value an error code. The values have the following meaning:

Error code	Description
0	No error has occurred
1	Incorrect parameter value
4	Connection to the measure analyzer cannot be established
5	Incorrect response received from the measure analyzer
6	Instrument is zeroing, no real time measurement possible at the moment
8	Feature is not supported on this instrument