# Application Note:

# Using Sherlock in Excel or other COM environments

Sherlock, in addition to being a stand-alone machine vision application, is a COM server. COM is an acronym for Component Object Model, which is a software architecture specification designed by Microsoft that allows components supplied by independent software vendors to interoperate in a dependable, regulated fashion in a Windows-based environment. The most common use of Sherlock's COM is within a Visual Basic Graphical User Interface.

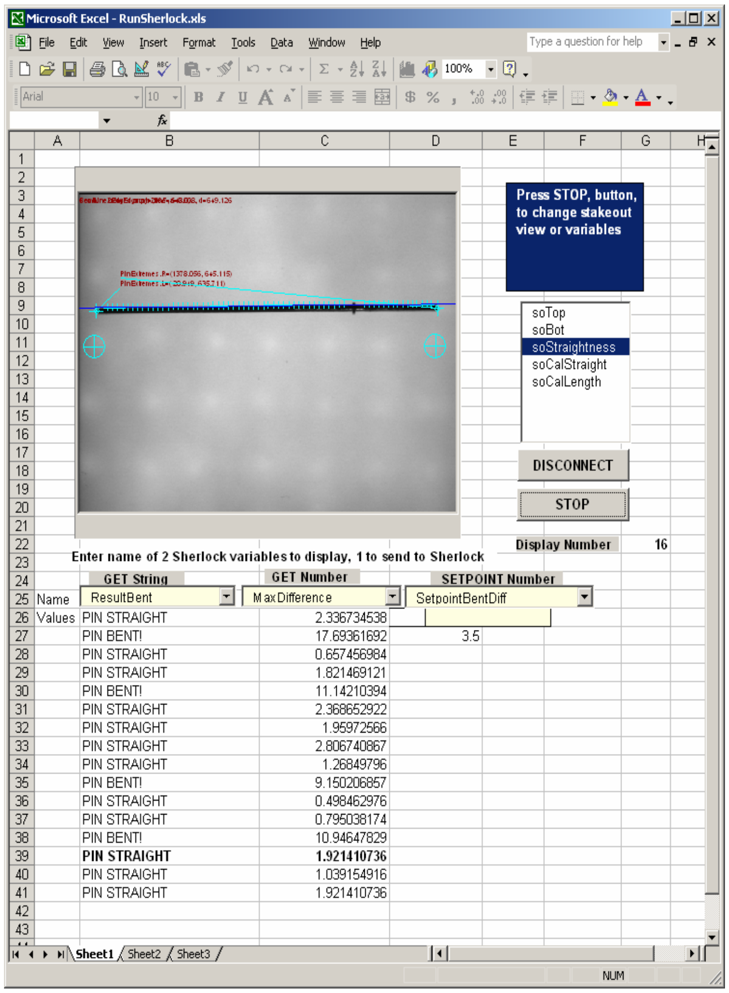
There may be situations occasions when you may need to use a front end other than Visual Basic for Sherlock. For example you may want to incorporate your Sherlock vision system into the commercial HMI package such as RSView or LabVIEW controlling your application. Or, you may want to provide a simplified front end to your operators in the form of an Excel spreadsheet.

This Application Note describes the use Sherlock's COM server ability within an Excel workbook. The procedure will be very similar to any other COM environment, and also to the procedure described in the **Sherlock User Manual** for including Sherlock in a Visual Basic project. The main difference is that the Excel workbook environment already exists, whereas you create the Visual Basic environment from scratch in the form of a Visual Basic Project. Also, you are very likely to already own Microsoft Excel, but may not have access to Visual Studio.

This Application Note includes the workbook RunSherlock.xls. It will run as described in the next section, connecting to Sherlock and loading the last active investigation. The tutorial section following that gives instructions for creating your own workbook, with code to paste into your VBA project.

## RunSherlock.xls Workbook Operation:

The workbook responds to the following 6 events:


Microsoft Excel - RunSherlock.xls
File Edit View Insert Format Tools Data Window Help
Type a question for help
100%
Arial
10
PinExtremes.R=(1378.056, 645.115)
PinExtremes.L=(20.949, 636.711)
Press STOP, button, to change stakeout view or variables
soTop
soBot
soStraightness
soCalStraight
soCalLength
DISCONNECT
STOP
Sheet1
Sheet2
Sheet3
NUM

| | A | B | C | D |
|---|---|---|---|---|
| 22 | | | | |
| 23 | | Enter name of 2 Sherlock variables to display, 1 to send to Sherlock | | |
| 24 | | GET String | GET Number | SETPOINT Number |
| 25 | Name | ResultBent | MaxDifference | SetpointBentDiff |
| 26 | Values | PIN STRAIGHT | 2.336734538 | |
| 27 | | PIN BENT! | 17.69361692 | 3.5 |
| 28 | | PIN STRAIGHT | 0.657456984 | |
| 29 | | PIN STRAIGHT | 1.821469121 | |
| 30 | | PIN BENT! | 11.14210394 | |
| 31 | | PIN STRAIGHT | 2.368652922 | |
| 32 | | PIN STRAIGHT | 1.95972566 | |
| 33 | | PIN STRAIGHT | 2.806740867 | |
| 34 | | PIN STRAIGHT | 1.26849796 | |
| 35 | | PIN BENT! | 9.150206857 | |
| 36 | | PIN STRAIGHT | 0.498462976 | |
| 37 | | PIN STRAIGHT | 0.795038174 | |
| 38 | | PIN BENT! | 10.94647829 | |
| 39 | | **PIN STRAIGHT** | **1.921410736** | |
| 40 | | PIN STRAIGHT | 1.039154916 | |
| 41 | | PIN STRAIGHT | 1.921410736 | |

Display Number 16

1. User Clicks **CONNECT** button on Sheet1
   a. A ***Sherlock*** object, objSherlock is created on Sheet1. This object is not visible but can be manipulated with VBA code in the code module for Sheet1.
   b. ***Sherlock*** object loads the last investigation that was run.
   c. A list of all Stakeouts in the investigation is obtained from ***Sherlock***. A ListBox on Sheet1 is populated with the stakeout names.
   d. A list of all Sherlock variables in the loaded investigation and their types is obtained from ***Sherlock***. All STRING types are added to one combo box on Sheet1, and all NUMBER types are added to two other combo boxes on Sheet1. Other variable types, such as BOOLEAN, are ignored.
   e. The **CONNECT** button is re-captioned to be the **DISCONNECT** button.
2. User clicks the **START** button on Sheet1.
   a. The Sherlock ***Display*** window on Sheet1 is connected to the stakeout whose name is selected in step 1c.
   b. ***Sherlock*** mode is set to Continuous_Inspection. Every time an investigation is completed, a ***Sherlock.Run_Completed*** event will be raised, which means that the host environment (Excel in this case) calls a subroutine named ***Sherlock_Run_Completed*** in the code module for Sheet1.
   c. The **START** button is re-captioned to be the **STOP** button
3. For each Sherlock Run_Completed event:
   a. The Sherlock ***Display*** window on Sheet1 is instructed to refresh its view.
   b. If the user has selected a STRING variable from combo box **<u>Variable1</u>**, the value of that variable is obtained from *Sherlock* and posted in a worksheet cell under the combo box.

    c. If the user has selected a NUMBER variable from combo box **<u>Variable2</u>**, the value of that variable is obtained from *Sherlock* and posted in a worksheet cell under the combo box
    d. **<u>Variable3</u>** is intended to display a setpoint – a variable that sets limits for determining Pass or Fail. If one has been selected by the user, and it's value has been changed in its Textbox, the new value is sent to ***Sherlock***.
    e. If the user has selected a setpoint variable from combo box **<u>Variable3</u>**, the new value of that variable is obtained from ***Sherlock*** and posted in a worksheet cell under the combo box.
4. User clicks the **STOP** button on Sheet1
    a. The Sherlock ***Display*** window on Sheet1 is disconnected from its stakeout.
    b. ***Sherlock*** mode is set to Halt_Inspections.
    c. User can now select a different stakeout, or select different variables to display or set. He can also change the number of values that are displayed.
    d. User can either click **Start**, can DISCONNECT Sherlock, or can close the workbook.
    e. The **STOP** button is re-captioned to be the **START** button
5. User clicks the **DISCONNECT** button on Sheet1
    a. ***Sherlock*** mode is set to Halt_Inspections. (ignored if already halted).
    b. The Sherlock ***Display*** window on Sheet1 is disconnected from its stakeout. (ignored if already disconnected)
    c. ***Sherlock*** object is deactivated (set to **Nothing)**
    d. The **DISCONNECT** button is re-captioned to be the **CONNECT** button
6. User closes the workbook
    a. ***Sherlock*** mode is set to Halt_Inspections. (ignored if already halted)
    b. The Sherlock ***Display*** window on Sheet1 is disconnected from its stakeout. (ignored if already disconnected)
    c. *If **not already done, Sherlock*** object is deactivated (set to **Nothing**)

The VBA code may be examined in the Visual Basic Editor. To do so, select *<u>Macro</u>* from the *<u>Tools</u>* menu, then *<u>Visual Basic Editor</u>*, or press alt-F11.

# Tutorial: Creating the workbook

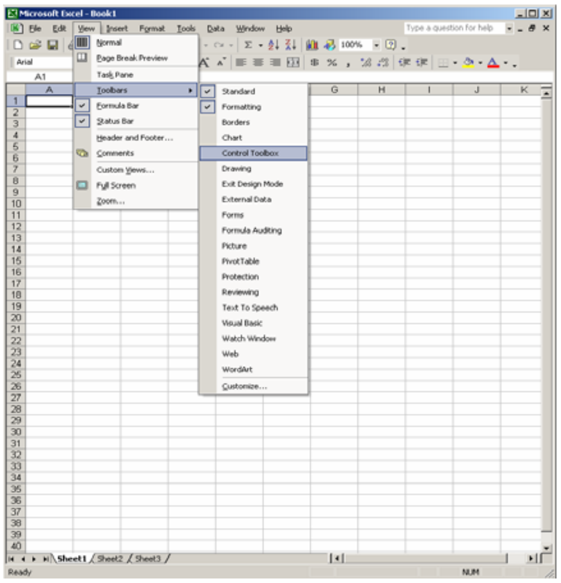
This tutorial will show you how to create an Excel workbook which uses the Sherlock COM methods, events, and properties, and which includes a Sherlock ActiveX Display window for live viewing of a stakeout.

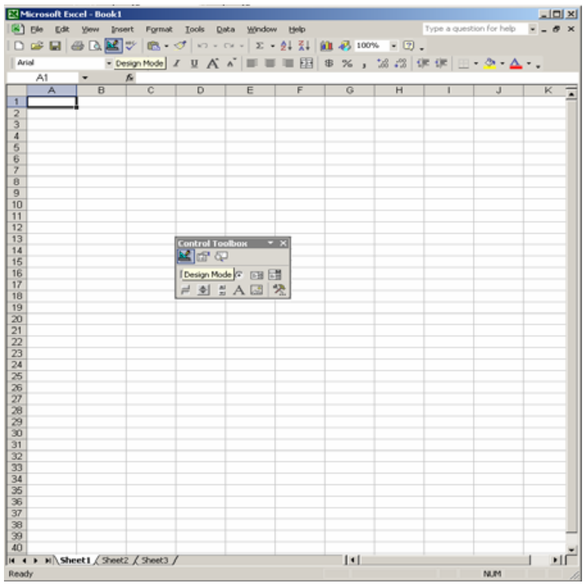
## Open a new workbook

- Close all open workbooks, and open a new Excel workbook.

## Enter Design Mode

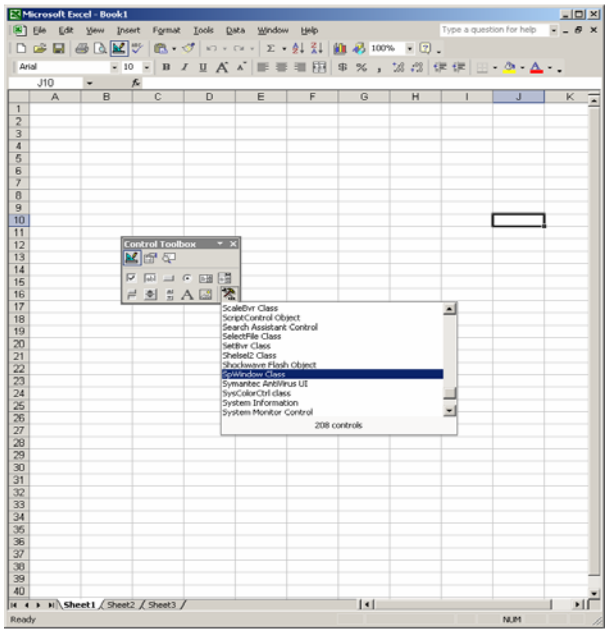
- Select the View menu, then Toolbars, then Control Toolbox. (see screenshot)
- The Control Toolbox may be shaped differently than this illustration. You can reshape it by dragging its edges.
- Mouse hover over the icons in the toolbox to see pop-up descriptions of the controls. Click the icon in the upper left of the Control Toolbox, ***Enter Design Mode***. When in design mode, this icon becomes ***Exit Design Mode***.



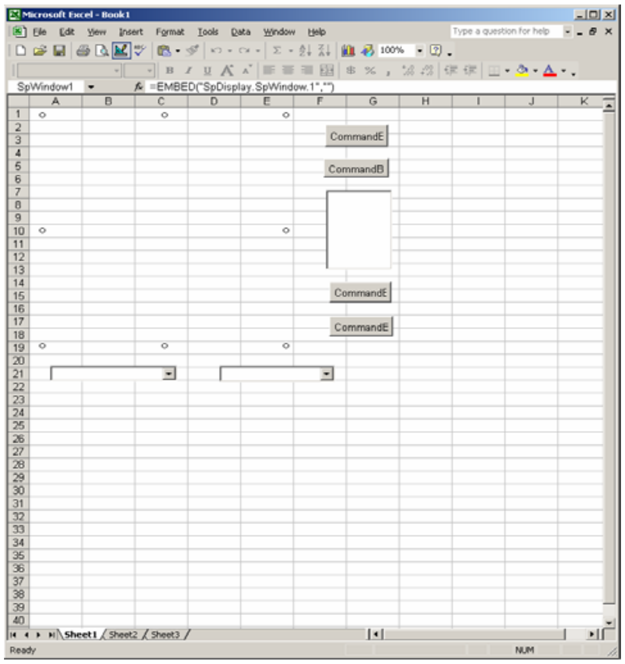


## Add Stakeout Display, Command Buttons, and ListBoxes

- While in Design Mode, click the icon at lower right, ***More Controls***



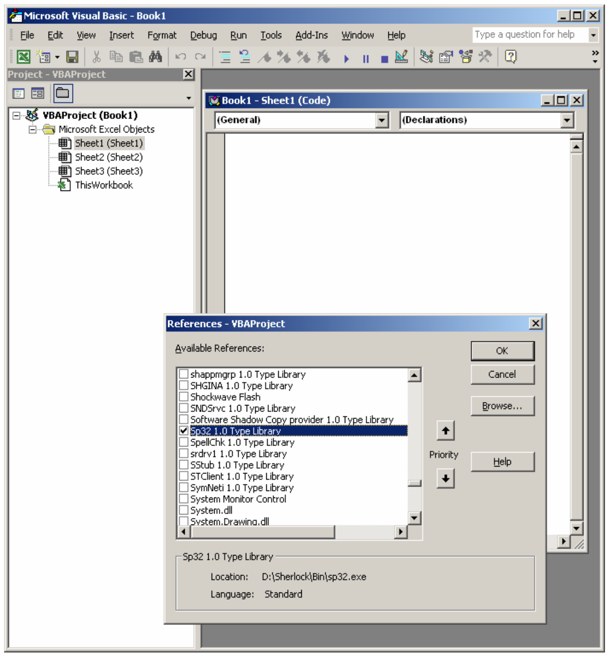
- Select ***SpWindow Class*** from the pop-up list. The items are listed alphabetically.
- Draw a rectangle from cell A1 to cell E19 for your ***SherlockDisplay***. It can be resized later. The control will be transparent while in ***Design Mode***, but is visible in ***Run Mode***.
- Select controls from the Control Toolbox by clicking the control's icon once, then drawing the shape on the worksheet. Add 4 command buttons, one List Box, and two Combo Boxes, as shown below.


SpWindow1 =EMBED("SpDisplay.SpWindow.1","")

- Right-Click the top command button and select Properties. Change the Name to cmdConnect and change the Caption to CONNECT. Change the remaining buttons to cmdDisconnect and DISCONNECT, cmdStart and START, cmdStop and STOP.
- In the same manner as the command buttons, change the ListBox name to lboxStakeouts, one ComboBox name to cboxVariable1, and the other ComboBox name to cboxVariable2.
- Exit ***Design Mode*** by clicking on ***Exit Design Mode***. Close the Control Toolbox

## Add project reference to Sherlock library

- Open the Visual Basic Editor. To do so, select <u>*Macro*</u> from the <u>*Tools*</u> menu, then <u>*Visual Basic Editor*</u>, or press alt-F11
- From the Visual Basic editor, click the Tools menu item and select References… to bring up the VBAProject References Dialog box. Note that SpDisplay 1.0 Type Library is already included as part of the Sherlock Display added to Sheet1 earlier.
- Unselected entries are arranged alphabetically following selected entries. Scroll down to Sp32 1.0 Type Library and select it by checking its checkbox. Click OK.



- [ ] shappmgrp 1.0 Type Library
- [ ] SHGINA 1.0 Type Library
- [ ] Shockwave Flash
- [ ] SNDSrvc 1.0 Type Library
- [ ] Software Shadow Copy provider 1.0 Type Library
- [x] Sp32 1.0 Type Library
- [ ] SpellChk 1.0 Type Library
- [ ] srdrv1 1.0 Type Library
- [ ] SStub 1.0 Type Library
- [ ] STClient 1.0 Type Library
- [ ] SymNeti 1.0 Type Library
- [ ] System Monitor Control
- [ ] System.dll
- [ ] System.Drawing.dll

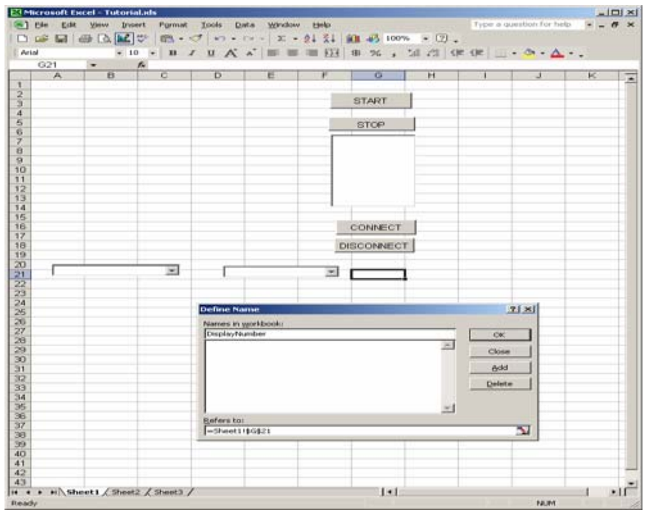
Sp32 1.0 Type Library
Location: D:\Sherlock\Bin\sp32.exe
Language: Standard

## Add event handlers to the buttons

- While still in the Visual Basic Editor, paste the Code For Sheet1, in the <u>*VBA Code*</u> section at the end of this document80
- , into the Sheet1(Code) window. If it is not already open, double click on Sheet1 in the Project Explorer pane on the left side of the screen.
- Paste the Code For ThisWorkbook, in the <u>*VBA Code*</u> section, into the ThisWorkbook (Code) window. If it is not already open, double click on ThisWorkbook in the Project Explorer pane on the left side of the screen

## Add a named range to Sheet1

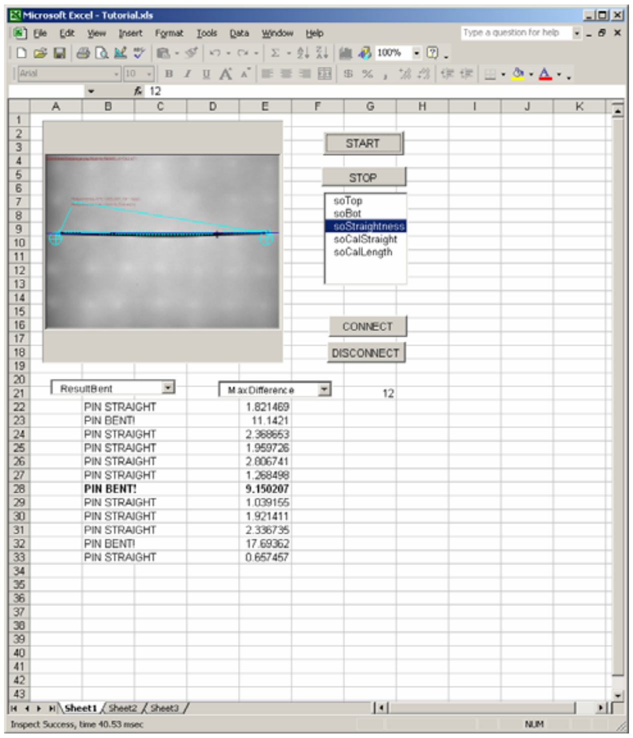
- Go back to the worksheet view, Sheet1.
- Click on cell G21
- Select the **Insert** menu item, then **Name**, then **Define**.
- Enter ***DisplayNumber*** in the dialog's name line. Click OK. Enter the number of variables to display before wrapping back to the top. Enter 12 for now. You can change it at any time.



Names in workbook:
DisplayNumber
Refers to:
=Sheet1!$G$21

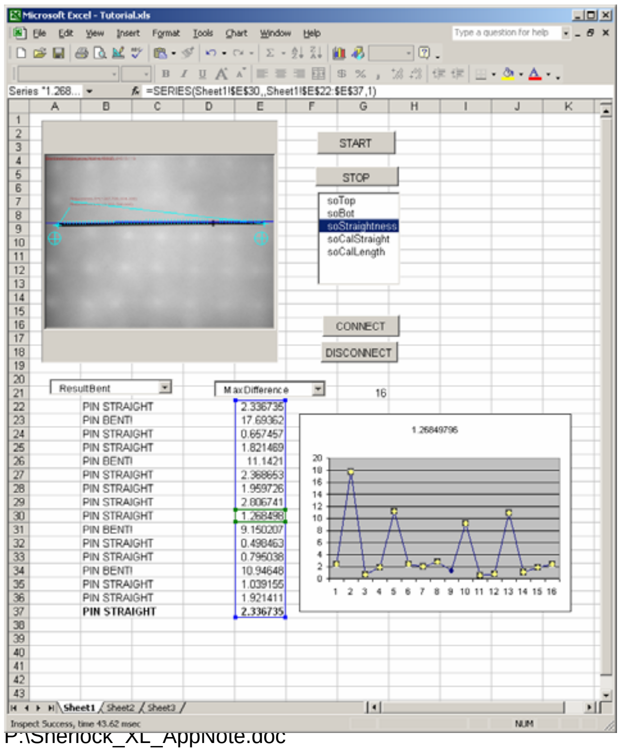
## Save the workbook and test it.

- Save the workbook. You might call it **Tutorial.xls**. Or you might name it after your favorite machine vision supplier (ipd is spelled with all lowercase letters).
- Click **CONNECT**. Watch for progress messages in the status bar. Click **START**. Select a variable from a combo box and watch it display.
- If VBA detects an error, it will raise a message box. Click on the **Debug** button to display the line of VBA code, or sometimes the VBA code module, that caused the error.



| Row | B | E | G |
|---|---|---|---|
| 21 | ResultBent | MaxDifference | 12 |
| 22 | PIN STRAIGHT | 1.821469 | |
| 23 | PIN BENT! | 11.1421 | |
| 24 | PIN STRAIGHT | 2.368653 | |
| 25 | PIN STRAIGHT | 1.959726 | |
| 26 | PIN STRAIGHT | 2.806741 | |
| 27 | PIN STRAIGHT | 1.268498 | |
| 28 | **PIN BENT!** | **9.150207** | |
| 29 | PIN STRAIGHT | 1.039155 | |
| 30 | PIN STRAIGHT | 1.921411 | |
| 31 | PIN STRAIGHT | 2.336735 | |
| 32 | PIN BENT! | 17.69362 | |
| 33 | PIN STRAIGHT | 0.657457 | |

Inspect Success, time 40.53 msec



Series "1.268... =SERIES(Sheet1!$E$30,,Sheet1!$E$22:$E$37,1)

| Row | B | E | G |
|---|---|---|---|
| 21 | ResultBent | MaxDifference | 16 |
| 22 | PIN STRAIGHT | 2.336735 | |
| 23 | PIN BENT! | 17.69362 | |
| 24 | PIN STRAIGHT | 0.657457 | |
| 25 | PIN STRAIGHT | 1.821469 | |
| 26 | PIN BENT! | 11.1421 | |
| 27 | PIN STRAIGHT | 2.368653 | |
| 28 | PIN STRAIGHT | 1.959726 | |
| 29 | PIN STRAIGHT | 2.806741 | |
| 30 | PIN STRAIGHT | 1.268498 | |
| 31 | PIN BENT! | 9.150207 | |
| 32 | PIN STRAIGHT | 0.498463 | |
| 33 | PIN STRAIGHT | 0.796038 | |
| 34 | PIN BENT! | 10.94648 | |
| 35 | PIN STRAIGHT | 1.039155 | |
| 36 | PIN STRAIGHT | 1.921411 | |
| 37 | **PIN STRAIGHT** | **2.336735** | |

Inspect Success, time 43.62 msec

## Bonus Feature

An example of a way to take advantage of the host environment is by use of the Excel Chart wizard.

- Click **STOP**. Reformatting the spreadsheet while Sherlock is updating it will result in an error.
- Highlight the cells containing Sherlock values.
- Start the Chart Wizard – **Insert** menu, then select **Chart**.
- Click **Line** Chart, then click **Finish**.
- Resize by grabbing the sizing handles at the corners
- Click the **START** button. The chart will be updated in real time.

## VBA Code

### ***Sheet1 Code*** *Copy and paste into the* <u>*Sheet1*</u> *code module of your VBA project as instructed in the tutorial section*

```
Option Explicit

Public WithEvents objSherlock As Sherlock
Public nErr As RET_TYPE
Public strSource As String
Public FullInvestigationName As String



Private Sub cmdConnect_Click()
   Dim strInvName As String
   Dim iCtr As Integer
   Dim strNames() As String
   Dim VarNames() As String, VarType() As Long


      '
      ' Create and activate Sherlock object, load last active investigation
      '
      Application.StatusBar = "Loading Sherlock last investigation..."
      Set objSherlock = CreateObject("Sp32.Sherlock")
      nErr = objSherlock.InvestigationLoadLast                'Get last active investigation
      nErr = objSherlock.InvestigationNameGet(FullInvestigationName) 'Get the investigation
name

      '
      ' Display fully qualified investigation name in StatusBar
      '
      Application.StatusBar = "Sherlock investigation at " & FullInvestigationName

      '
      ' Get names of all stakeouts so user can select
      ' one for display
      '
      nErr = objSherlock.StakeoutsGet(strNames)
      For iCtr = 0 To UBound(strNames)
         lboxStakeouts.AddItem strNames(iCtr)
      Next iCtr
      lboxStakeouts.ListIndex = 0

      '
```

```
        ' Get names of all STRING and NUMBER variables so user can
        ' select a couple for display.  All others are ignored.
        '
        ' Variables can be of type VAR_STRING, VAR_NUMBER, VAR_BOOL,
        ' VAR_POINT, VAR_LINE, VAR_NUMBER_ARRAY, VAR_BOOL_ARRAY,
        ' VAR_POINT_ARRAY, VAR_LINE_ARRAY, or VAR_STRING_ARRAY.
        '
        nErr = objSherlock.VarGetInfo(VarNames, VarType)
        For iCtr = 0 To UBound(VarNames)
            If VarType(iCtr) = VAR_STRING Then
                cboxVariable1.AddItem VarNames(iCtr)
            ElseIf VarType(iCtr) = VAR_NUMBER Then
                    cboxVariable2.AddItem VarNames(iCtr)
            End If
        Next iCtr

        ' Enable the START button
        cmdStart.Enabled = True

End Sub 'cmdConnect_Click()


   Private Sub cmdDisconnect_Click()
        'Close out Sherlock
        If Not (objSherlock Is Nothing) Then
            Application.StatusBar = "Closing Sherlock..."
            nErr = objSherlock.InvestigationModeSet(I_HALT)
            SpWindow1.DisconnectStakeout
            Set objSherlock = Nothing
        End If
        '
        ' Clear names of all stakeouts and variables
        '
        lboxStakeouts.Clear
        lboxStakeouts.ListIndex = -1
        cboxVariable1.Clear
        cboxVariable1.ListIndex = -1
        cboxVariable2.Clear
        cboxVariable2.ListIndex = -1

        ' Disable the START button and prompt for action
        cmdStart.Enabled = False

    '
    ' Supress messages asking user to save workbook
    '
    Application.DisplayAlerts = False
```

```
End Sub 'cmdDisconnect_Click()



Private Sub cmdStart_Click()
   Dim strCurName As String


      '
      ' Connect selected stakeout and start investigations
      '
      strCurName = lboxStakeouts.Text   'Get name from ListBox
      SpWindow1.ConnectStakeout strCurName
      nErr = objSherlock.InvestigationModeSet(I_CONT)
End Sub 'cmdStart_Click()

Private Sub cmdStop_Click()
      '
      ' Halt investigations and Disconnect stakeout
      '
      If Not (objSherlock Is Nothing) Then
         nErr = objSherlock.InvestigationModeSet(I_HALT)
         SpWindow1.DisconnectStakeout
      End If
      TextBox1.Text = "Select a stakeout from the list below, then press START button"
   End If

End Sub 'cmdStartup_Click()


'
' Investigation Complete
'   Update the stakeout display
'   Display/Set any variables specified by user
'
Private Sub objSherlock_RunCompleted(ByVal bSuccess As Long, _
                                    ByVal dTime As Double)

   Dim strNames() As String
   Dim Var1Name As String, var1Value As String
   Dim Var2Name As String, var2Value As Double
   Dim Var3Name As String, var3Value As Double
   Dim NbrToShow As Integer
   Static iCounter As Long
   Dim iCtr As Integer, iCtr2 As Integer
   Dim DispRow As Integer

   SpWindow1.UpdateDisplay      'Update Stakeout display
```

```
    DispRow = Range("DisplayNumber").Row + 1

    If Range("DisplayNumber") <> "" Then
        NbrToShow = Range("DisplayNumber")
    Else
        NbrToShow = 12
        Range("DisplayNumber") = 12
    End If

    'Track the rolling display of current reading
    iCtr = iCounter Mod NbrToShow
    'Track prior reading
    iCtr2 = (iCounter - 1) Mod NbrToShow
    'Update the static counter for next reading
    iCounter = iCounter + 1

    ' Refresh display of string var 1 if specified by user
    If Trim(cboxVariable1.Text) <> "" Then
        Var1Name = Trim(cboxVariable1.Text)
        nErr = objSherlock.VarStringGet(Var1Name, var1Value)
        'Sheet1.Range("BDispRow") = var1Value
        Cells(DispRow + iCtr, 2) = var1Value
        Cells(DispRow + iCtr, 2).Font.Bold = True   'Highlight the current reading
        Cells(DispRow + iCtr2, 2).Font.Bold = False 'Un-Highlight the prior reading
    End If

    ' Refresh display of number var 2 if specified by user
    If Trim(cboxVariable2.Text) <> "" Then
        Var2Name = Trim(cboxVariable2.Text)
        nErr = objSherlock.VarNumberGet(Var2Name, var2Value)
        Cells(DispRow + iCtr, 5) = var2Value
        Cells(DispRow + iCtr, 5).Font.Bold = True  'Highlight the current reading
        Cells(DispRow + iCtr2, 5).Font.Bold = False 'Un-Highlight the prior reading
    End If

    ' Show inspect time and result
    Application.StatusBar = IIf(bSuccess = 1, "Inspect Success", "Inspect Failure") & _
                        ", time " & FormatNumber(dTime, 2, vbTrue) & " msec"

End Sub 'End objSherlock_RunCompleted()
```

## ***ThisWorkbook Code*** *Copy and paste into the* *ThisWorkbook* *code module of your VBA project as* instructed in the tutorial section.

```
Option Explicit

'
' User has asked Excel to close
'   If not already done, halt,
'   disconnect, and free Sherlock first,
'   for a clean Sherlock shutdown
'
Private Sub Workbook_BeforeClose(Cancel As Boolean)

    Dim RetCode As RET_TYPE

    If Not (Sheet1.objSherlock Is Nothing) Then
        Application.StatusBar = "Closing Sherlock before closing workbook..."
        RetCode = Sheet1.objSherlock.InvestigationModeSet(I_HALT)
        Sheet1.SpWindow1.DisconnectStakeout
        Set Sheet1.objSherlock = Nothing
    End If

End Sub 'Workbook_BeforeClose()
```

## ***Contact:***

### **IPD Engineer / Report Author:**

Bob McMinn
IPD Applications Engineer
Phone: 978-670-2075
rmcminn@goipd.com