Near-Exponential VBA Macro

- Charting for engineering and science models sometimes requires many orders of magnitude flexibility and speed in adjusting the scale of the axes of coordinates. Manual scaling using Excel's menu is cumbersome and a macro was already developed for doing this. That macro can independently change chart axis scales between 1 to 10000 in oscilloscope style: 1, 2, 5, 10, 20, 50, 100 …)
- The problem with that macro is that it contains a fairly large array (acting as a look-up table) which is defined within the body of the macro. That array contained 13 elements but sometimes we might need much more than that.
- This tutorial will solve that problem by introducing a similar approach in which we combine a 3-element array with two VBA functions (***Mod*** and ***Int***) to achieve a practically unlimited range with a very short macro.

by George Lungu

Let's look at the macro to the right. While associated with a button with a range of [0,12] this macro will achieve a 4 order of magnitude variation of the value in cell B3

1

```
Dim arrScale As Variant

Private Sub Scale_1_Change()
    arrScale = Array(1, 2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, 10000)
    [B3] = arrScale(Scale_1.Value)
End Sub
```

2

```
Dim a, b As Long
Dim arrScale As Variant

Private Sub Scale_2_Change()
    arrScale = Array(1, 2, 5)
    a = Scale_2.Value Mod 3
    b = Int(Scale_2.Value / 3)
    [B7] = arrScale(a) * 10 ^ b
End Sub
```

- This macro is a little longer but it will be able to generate values in a practically unlimited range.
- The modulo function "Mod n" applied to a number extracts the remainder of the division of that number by n.
- The function Int(x) will extract the integer part of x.
- You can check that the following is true: a =m*Int(a/m) + a Mod m where "a" and "m" are an integers

3

```
Private Sub Scale_3_Change()
    arrScale = Array(1, 2, 5)
    a = Scale_3.Value Mod 3
    b = (Scale_3.Value - Scale_3.Value Mod 3) / 3
    [B11] = arrScale(a) * 10 ^ b
End Sub
```

This macro is similar to #2. Use it if you forget the function "Int" but you remember "Mod"

4

```
Private Sub Scale_4_Change()
    arrScale = Array(1, 2, 5)
    a = Scale_4.Value - 3 * Int(Scale_4.Value / 3)
    b = Int(Scale_4.Value / 3)
    [B15] = arrScale(a) * 10 ^ b
End Sub
```

This macro is similar to #2. Use it if you forget the function "Mod" but you remember "Int"

```
Private Sub Scale_5_Change()
    arrScale = Array(1, 2, 5)
    [B19] = arrScale(Scale_5.Value Mod 3) * 10 ^ Int(Scale_5.Value / 3)
End Sub
```

5

Use this if you like to save some lines of code

Use this if you don't like to use VBA arrays

```
Private Sub Scale_6_Change()
    If (Scale_6.Value Mod 3) = 1 Then a = 2 Else a = 1
    If (Scale_6.Value Mod 3) = 2 Then a = 5
    [B23] = a * 10 ^ Int(Scale_6.Value / 3)
End Sub
```

6

All the macros are implemented in Sheet 1. The range of the first macro is 6 orders of magnitude and the rest have a range of 9 orders of magnitude (decided only by the spin button range: [0,27])

As an application of the previous ideas, the macros for a scatter chart axis scale change are shown below

```
Private Sub Scale_X_Change()
    arrScale = Array(1, 2, 5)
    a = arrScale(Scale_X.Value Mod 3) * 10 ^ Int(Scale_X.Value / 3)
    ActiveSheet.ChartObjects("Chart_1").Chart.Axes(xlCategory).MaximumScale = a
End Sub
----------------------------------------------------------------------------------------------------
Private Sub Scale_Y_Change()
    arrScale = Array(1, 2, 5)
    b = arrScale(Scale_Y.Value Mod 3) * 10 ^ Int(Scale_Y.Value / 3)
    ActiveSheet.ChartObjects("Chart_1").Chart.Axes(xlValue).MaximumScale = b
End Sub
```