

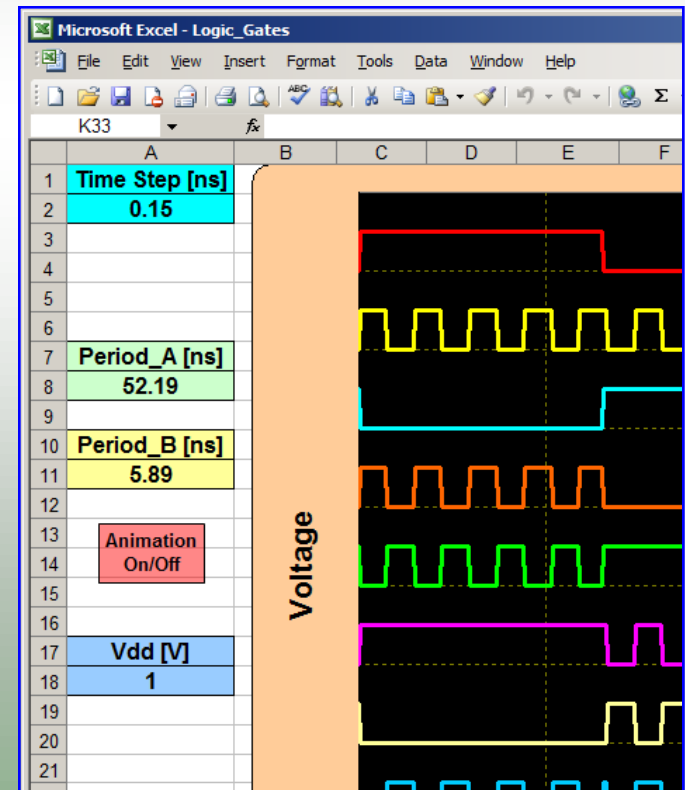
Introduction to Digital Electronics – Ideal Logic Gates – Part#3

by George Lungu

- This tutorial is the third part of the series introducing the most common logic gates and their implementation in Excel, first as ideal static models and later using more realistic animated models which include propagation delay, loading, strength, power supply voltage.
- In the first part we talked about common logic gates (INV, AND, NAND, OR, NOR, XOR, XNOR), their logic equations, their truth tables and then we went forward in the second part to implementing a first iteration of a model in Excel 2003 together with a simple animation.
- In this section the model is upgraded by employing user defined functions.

Create a new worksheet:

- In the original Excel file (named Logic_Gates.xls) make a copy of the last worksheet (Basic_3) and rename it "Basic_4".
 - In the new worksheet insert a new parameter called "vdd" which stands for the power supply voltage: cell A17: "Vdd [V]" (this is a just a label) => cell A18: "1"
 - Give cell A18 the name "vdd" by highlighting the cell and typing in the name box (upper left corner) the name "vdd" (without quotation marks).
 - If the name is not accepted it means you might have used it in a different place already and you can remove it first by clicking in any cell => Insert (top menu) => Name => Define => select the name from the pop up list => Delete
- <www.excelunusual.com>



What is a user defined function?

- MS Excel has a large number of built in functions but sometime we might need to create a special function that does not exist in the library of existing functions.
- The user defined functions will be much easier to use in the spreadsheet by simplifying the formulas but, when used widely within the worksheet, at the penalty of speed.

Create a user defined function for the inverter:

- Bring up the visual basic editor:
Tools => Macro => Visual Basic Editor
(or simply type Alt F11)

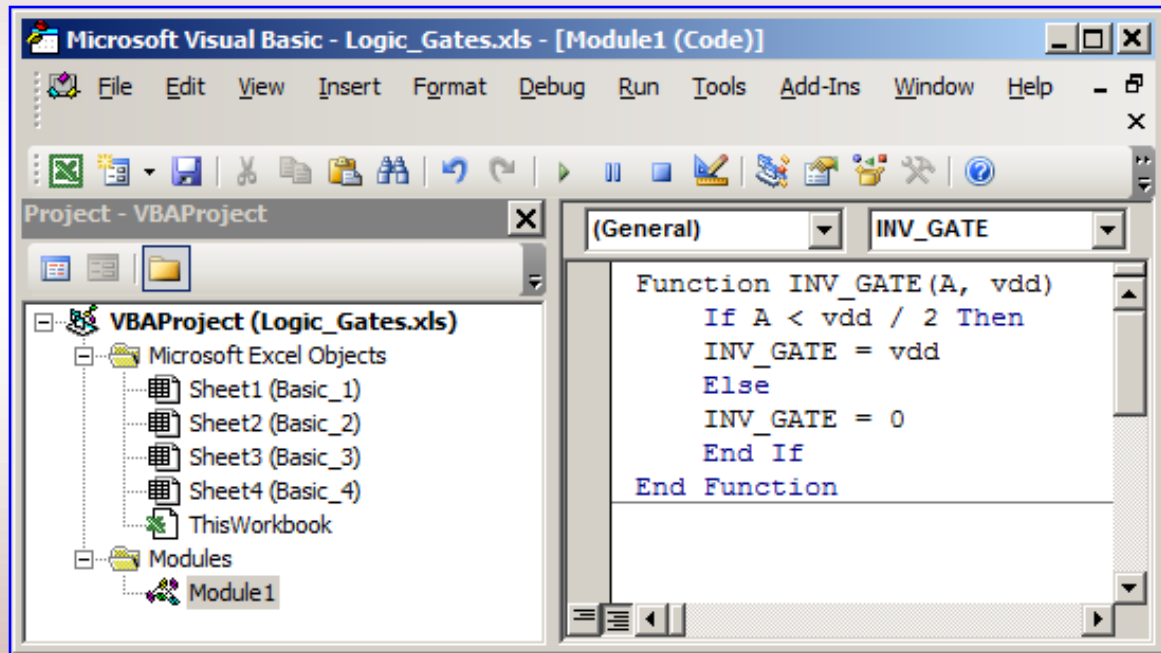
- The code for a function has to always be stored in a "Module" for the function to work.

- At the left (in the VBAProject area) click right => Insert => Module

- Move the cursor to the right (the edit area) and type the code seen in the next snapshot ==>>>

- A, and vdd are called the arguments of the function (the input and the voltage supply in this case)

- In certain cases the type (integer, double, Boolean, etc) must be specified for both the function return (INV_GATE) and the arguments (A and vdd) as we will see in the next page.



A more complete syntax:

- This is the same function as the one in the previous page except that the type of each argument and the type of return are explicit.

```
Function INV_GATE(A As Double, vdd As Double) As Double
  If A < vdd / 2 Then
    INV_GATE = vdd
  Else
    INV_GATE = 0
  End If
End Function
```

Create the functions AND, NAND, OR and NOR

```
Function AND_GATE(A, B, vdd) As Double
  If A > vdd / 2 And B > vdd / 2 Then
    AND_GATE = vdd
  Else
    AND_GATE = 0
  End If
End Function
```

```
Function NAND_GATE(A, B, vdd) As Double
  If A > vdd / 2 And B > vdd / 2 Then
    NAND_GATE = 0
  Else
    NAND_GATE = vdd
  End If
End Function
```

```
Function OR_GATE(A, B, vdd) As Double
  If A < vdd / 2 And B < vdd / 2 Then
    OR_GATE = 0
  Else
    OR_GATE = vdd
  End If
End Function
```

```
Function NOR_GATE(A, B, vdd) As Double
  If A < vdd / 2 And B < vdd / 2 Then
    NOR_GATE = vdd
  Else
    NOR_GATE = 0
  End If
End Function
```

Create XOR function:

```
Function XOR_GATE(A, B, vdd) As Double
    If (A < vdd / 2 And B > vdd / 2) Or (A > vdd / 2 And B < vdd / 2) Then
        XOR_GATE = vdd
    Else
        XOR_GATE = 0
    End If
End Function
```

Create XNOR function:

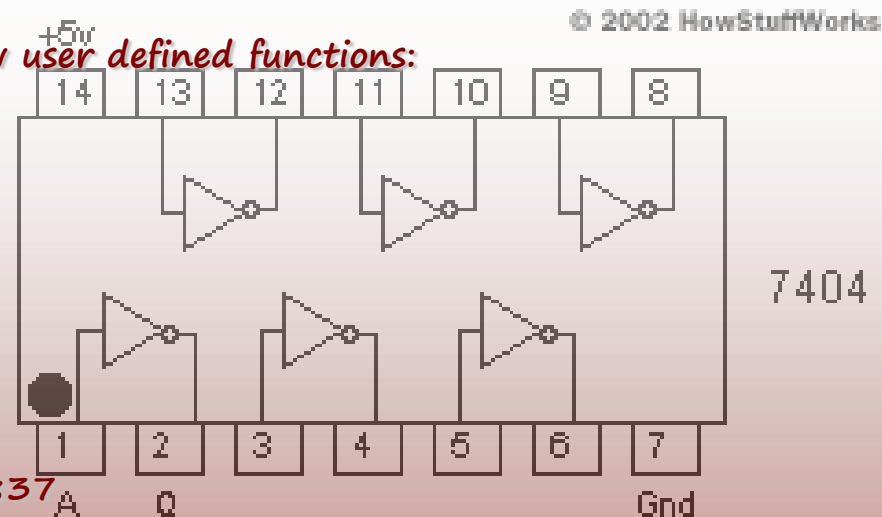
```
Function XNOR_GATE(A, B, vdd) As Double
    If (A < vdd / 2 And B > vdd / 2) Or (A > vdd / 2 And B < vdd / 2) Then
        XNOR_GATE = 0
    Else
        XNOR_GATE = vdd
    End If
End Function
```

Using the custom functions in the worksheet:

- Modify the following cell formulas to include the new user defined functions:

- Cell D37: “=INV_gate(B37,vdd)”
- Cell E37: “=AND_gate(B37,C37,vdd)”
- Cell F37: “=NAND_gate(B37,C37,vdd)”
- Cell G37: “=OR_gate(B37,C37,vdd)”
- Cell H37: “=NOR_gate(B37,C37,vdd)”
- Cell I37: “=XOR_gate(B37,C37,vdd)”
- Cell J37: “=XNOR_gate(B37,C37,vdd)”

after which, copy range D37:J37 down to row 837

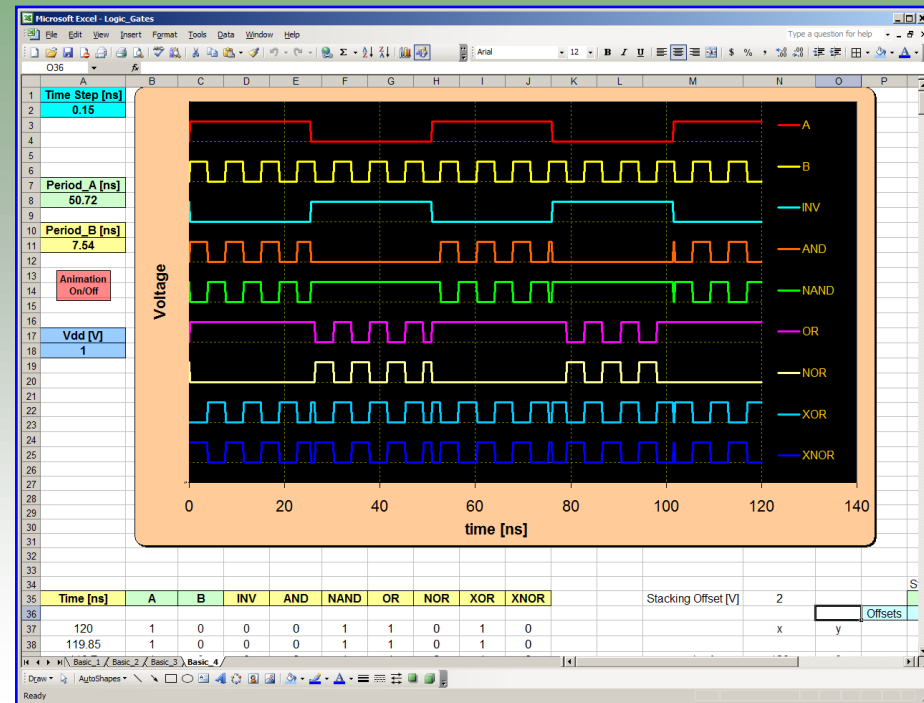


The final model:

– Using the new setup involving custom functions, we can run the macro by clicking the “Animation_On/Off” button. We see that the model is running but it is much slower than the previous one (using built-in cell formulas)

A special macro for measuring the speed of the model:

```
Sub speed_test()  
N = Not N  
[A25] = 0  
Measurement_Start = Timer()  
Do Until N = False Or (Timer() – Measurement_Start) > 60  
p = p + 0.1  
DoEvents  
[A25] = [A25] + 1  
[A8] = 10 * (4 + 2 * Sin(p / 10))  
DoEvents  
[A11] = 10 * (2.2 + 2 * Sin(p / 7))  
DoEvents  
Loop  
End Sub
```



– We used a special macro which runs exactly for 60 seconds then stops the model and during all this time it records the number of iterations in cell A25.

– The time function “Timer()” records the number of seconds since midnight

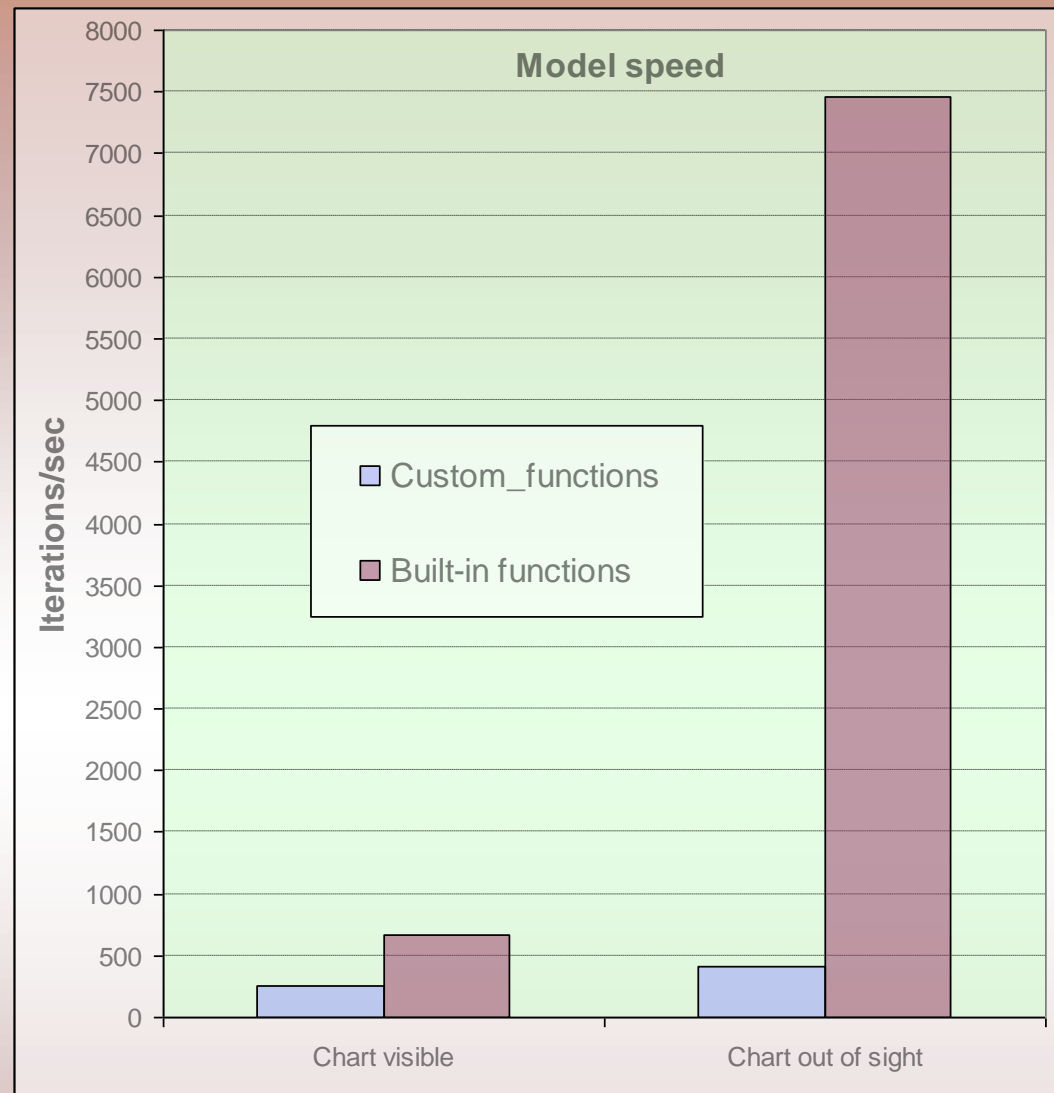
– “Measurement_Start” is a variable defined as type double and it is initialized with the number of seconds passed since midnight at the moment of starting the macro

Speed test results:

- There were a two different tests run, one in which both models (one with custom functions and one with built-in functions) were run with the waveform chart visible. As expected, the speed was pretty low for both cases but lower for the custom function model. It seems that live char-ting consumes a significant amount of computer resources.

- For the second test the chart was moved out of sight so that a pure computation speed could be measured. Here the speed difference was more than an order of magnitude in favor of the model with built-in functions.

- As a result I highly recommend refraining from using (or sparingly using) user defined functions in any animated model that require speed !!!



	Custom_functions	Built-in functions
Chart visible	260	674
Chart out of sight	412	7455