Session 2222

Using Multiple Software Packages to Solve Complex Problems

Micheal Parten
Department of Electrical and Computer Engineering
Texas Tech University

I Introduction

Engineering students normal learn to use a number of different software and simulation packages. However, in most cases, they use and view these as separate entities that have little relationship to each other. Thus, when one software package doesn't do quite what is really needed they frequently abandon the software package, the problem they are trying to solve or both. Using multiple software packages to solve problems is an important lesson to learn.

This paper presents a specific example of how to incorporate multiple software programs that can demonstrate to students the power of this approach. In this case, an electric circuit simulation package is used in conjunction with statistical analysis. Specifically, the maximum frequency of operation of complex integrated circuits can frequently be estimated by examining the frequency of a ring oscillator. The many complex processing steps necessary to build the integrated circuit have variations that change the parameters of the circuit changing the maximum frequency of oscillation. Predicting the effect specific process variations have on the maximum frequency of oscillation is a complex and difficult task. However, the approach to the problem, which is applicable to many other engineering problems, is manageable. The approach requires using circuit simulation software along with statistical analysis software and the necessary interface between the two packages. The information below comes from this assignment to a group of students. All of the figures come directly from student presentations.

II Circuit Simulation

A CMOS ring oscillator is shown in Figure 1. The transient response of the oscillator can be simulated using a number of different circuit simulation packages, in this case, PSPICE. The MOS devices are represented by complex, nonlinear models. In this example, a level 3 model was used. The model coefficients for the specific NMOS and PMOS transistors used in this example are shown in Figure 2.

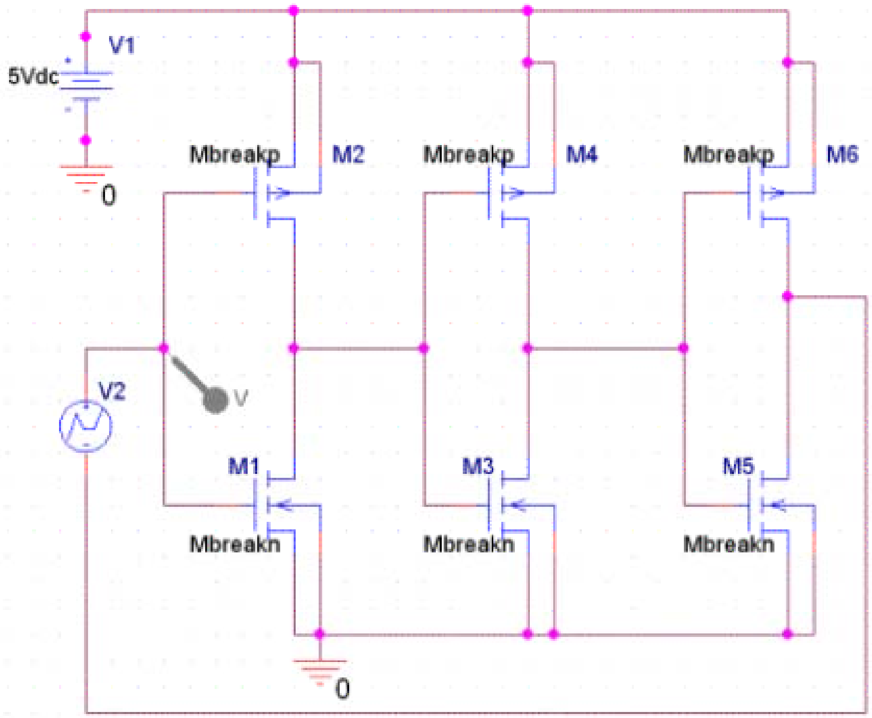


Figure 1. CMOS Ring Oscillator

```
.model Mbreakn NMOS(Level=3 Gamma=0 Delta=0 Eta=0 Theta=0 Kappa=0 Vmax=0 Xj=0
+             Tox=100n Uo=600 Phi=.6 Rs=1.624m Kp=20.53u W=.3 L=2u Vto=2.831
+           Rd=1.031m Rds=444.4K Cbd=3.229n Pb=.8 Mj=.5 Fc=.5 Cgso=9.027n
+             Cgdo=1.679n Rg=13.89 Is=194E-18 N=1 Tt=288n)
.model Mbreakp PMOS(Level=3 Gamma=0 Delta=0 Eta=0 Theta=0 Kappa=0 Vmax=0 Xj=0
+             Tox=100n Uo=300 Phi=.6 Rs=70.6m Kp=10.15u W=1.9 L=2u Vto=-3.67
+             Rd=60.66m Rds=444.4K Cbd=2.141n Pb=.8 Mj=.5 Fc=.5 Cgso=877.2p
+             Cgdo=369.3p Rg=.811 Is=52.23E-18 N=2 Tt=140n)
```

Figure 2. MOS Model Parameters

The circuit was simulated for transient analysis with a Lot variation of Tox of 20% for each transistor. A Monte Carlo simulation of 20 values with a uniform distribution was run. The resulting transient result is shown in Figure 3 along with the FFT frequency decomposition in Figure 4 and a sample log file in Figure 5.

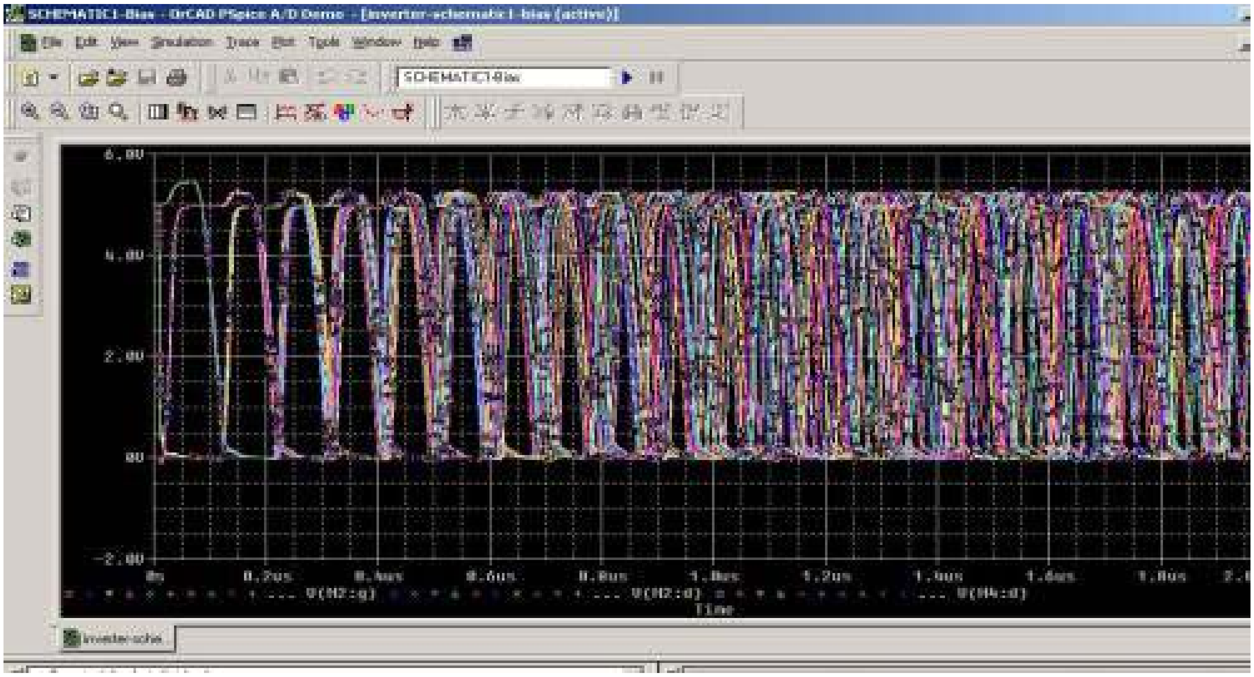


Figure 3. Ring Oscillator Transient Response to Changes in Tox.

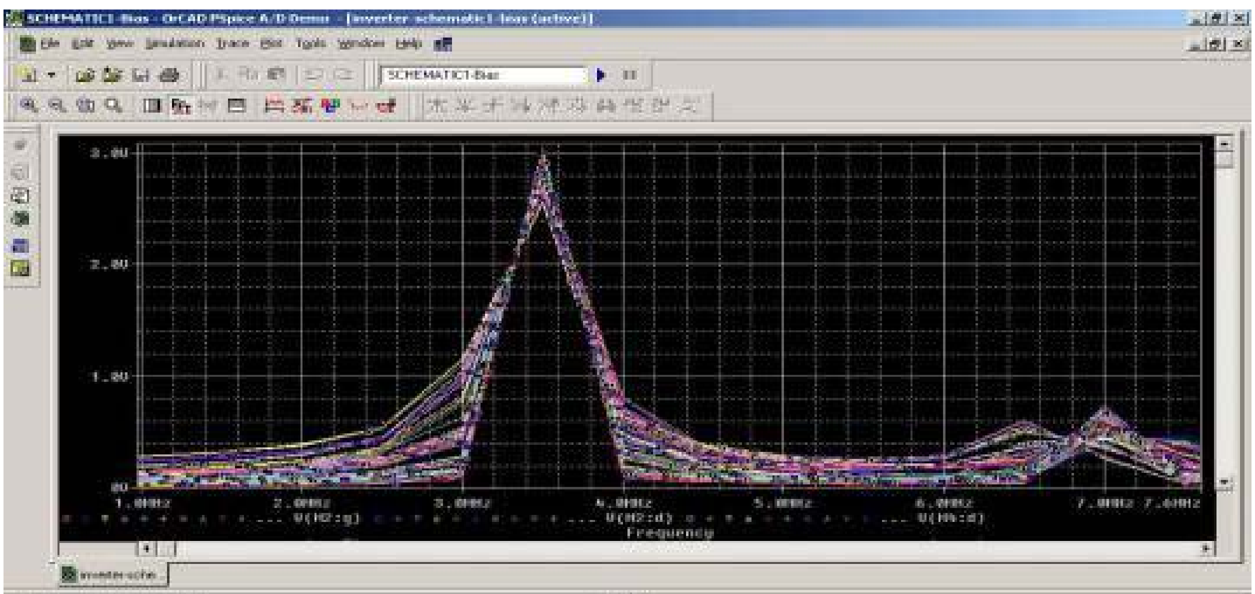
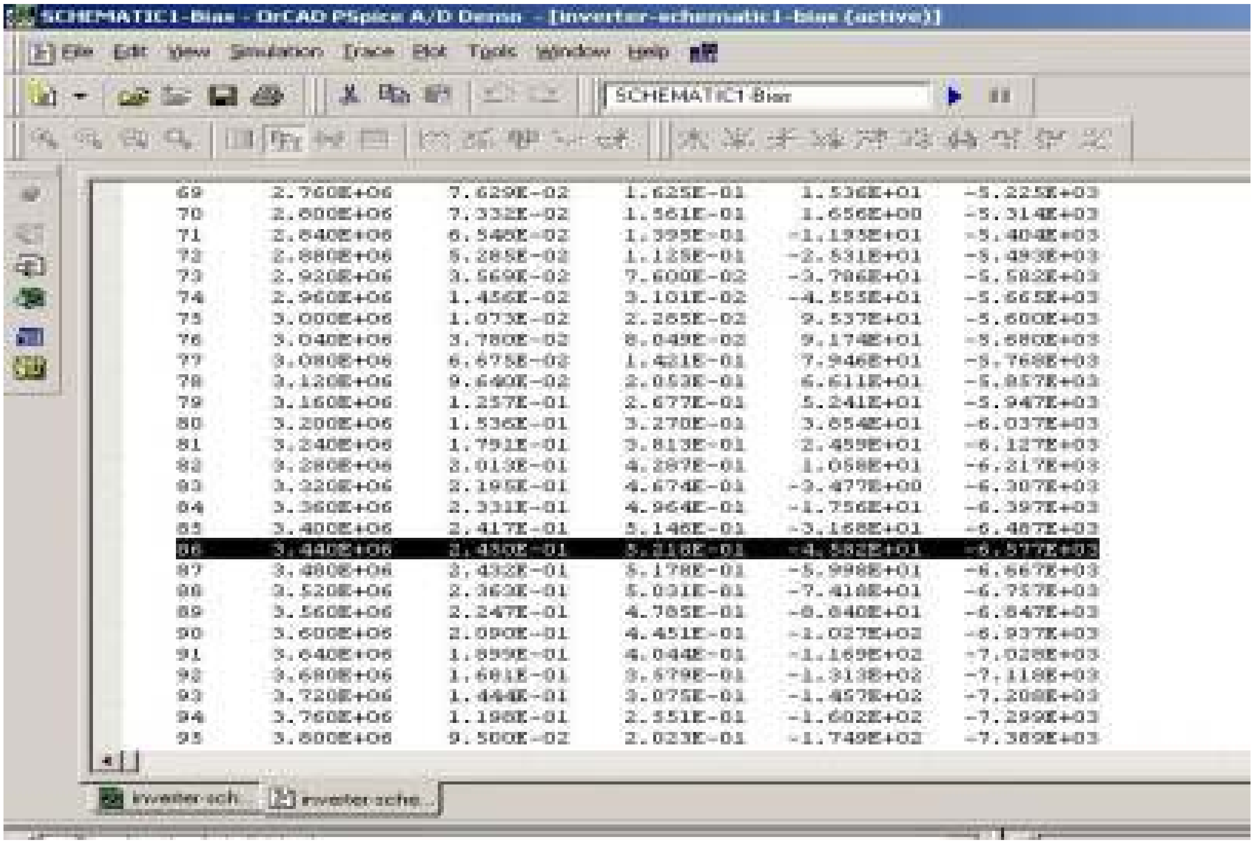


Figure 4. Ring Oscillator Frequency to Changes in Tox.



SCHEMATIC1-Bias - OrCAD PSpice A/D Demo - [inverter-schematic1-bias (active)]

69	2.760E+06	7.629E-02	1.625E-01	1.536E+01	-5.225E+03
70	2.800E+06	7.332E-02	1.561E-01	1.656E+00	-5.314E+03
71	2.840E+06	6.546E-02	1.395E-01	-1.195E+01	-5.404E+03
72	2.880E+06	5.285E-02	1.125E-01	-2.531E+01	-5.493E+03
73	2.920E+06	3.569E-02	7.600E-02	-3.786E+01	-5.582E+03
74	2.960E+06	1.456E-02	3.101E-02	-4.555E+01	-5.665E+03
75	3.000E+06	1.073E-02	2.285E-02	9.537E+01	-5.600E+03
76	3.040E+06	3.780E-02	8.049E-02	9.174E+01	-5.680E+03
77	3.080E+06	6.675E-02	1.421E-01	7.946E+01	-5.768E+03
78	3.120E+06	9.640E-02	2.053E-01	6.611E+01	-5.857E+03
79	3.160E+06	1.257E-01	2.677E-01	5.241E+01	-5.947E+03
80	3.200E+06	1.536E-01	3.270E-01	3.854E+01	-6.037E+03
81	3.240E+06	1.791E-01	3.813E-01	2.459E+01	-6.127E+03
82	3.280E+06	2.013E-01	4.287E-01	1.058E+01	-6.217E+03
83	3.320E+06	2.195E-01	4.674E-01	-3.477E+00	-6.307E+03
84	3.360E+06	2.331E-01	4.964E-01	-1.756E+01	-6.397E+03
85	3.400E+06	2.417E-01	5.146E-01	-3.168E+01	-6.487E+03
86	3.440E+06	2.450E-01	5.218E-01	-4.582E+01	-6.577E+03
87	3.480E+06	2.432E-01	5.178E-01	-5.998E+01	-6.667E+03
88	3.520E+06	2.363E-01	5.031E-01	-7.418E+01	-6.757E+03
89	3.560E+06	2.247E-01	4.785E-01	-8.840E+01	-6.847E+03
90	3.600E+06	2.090E-01	4.451E-01	-1.027E+02	-6.937E+03
91	3.640E+06	1.899E-01	4.044E-01	-1.169E+02	-7.028E+03
92	3.680E+06	1.681E-01	3.579E-01	-1.313E+02	-7.118E+03
93	3.720E+06	1.444E-01	3.075E-01	-1.457E+02	-7.208E+03
94	3.760E+06	1.198E-01	2.551E-01	-1.602E+02	-7.299E+03
95	3.800E+06	9.500E-02	2.023E-01	-1.749E+02	-7.389E+03

Figure 5. Sample PSPICE Log File

III Data Extraction

The oscillator data was extracted from PSpice log files by writing a Perl script. Perl is a high level programming (scripting) language. It is a derivative of C as well as other languages. Perl was designed for text and file manipulation. The reasons for using Perl are that the PSpice log is large and verbose (~3500 lines), but the log is highly patterned. Perl is well suited for data extraction in this case. To extract data with Perl, the input is read into an array by

```
Open (In_File, “<‘file name’”);
@lines = <In_File>;
```

The input can then be examined, looking for specific headers, by

```
foreach $line (@lines)
{ }
if $line =~ /’search material’/
{ }
```

Lines can be split into an array based on a data divider by

@colors=split(/ /,”red blue green”);

One, some, or all of the elements can be stored or printed.

It is essential to know what data is important to extract. For the ring oscillator example, frequency changes are critical. The Monte Carlo analysis changes the designated model parameters for each run. The value of the varied parameters and the resulting frequency are required. The Perl program

- determines the Monte Carlo Pass number and the pass variables,
- reads in all frequencies and amplitudes from the Fourier analysis,
- determines the fundamental frequency, and
- reports all of the required information.

Figure 6 shows portions of the Perl program and a screen shot of the output file.

```
#handle printing next lines  $prnnext
if ($prnnext > 0)
{
        $prnnext --;
        if ($prnnext == 0  )
        {
                @parameters = split(/      /,$line);
                @parameters[2] =~ s/ //; #get rid of extra space
                #print "@parameters[2]: @parameters[3]\n";
                $space = " ";
                $newLine = "\n";
                $variable[$test_index]= $variable[$test_index].$newLine.$model_change;
                $variable[$test_index]= $variable[$test_index].$space.@parameters[2].$space.@parameters[3];
        }
}
```

```
sub ScitoDec()
#changes an array of numbers in scientific notation to an array of numbers in
#decimal notation
{
        local(@array) = @_;
        foreach $testnum (@array)
        {
                $testnum =~ s/^-0/^-/;
                @splitnum = split(/E/,$testnum);
                $number = (10**@splitnum[1]*@splitnum[0]);
                $testnum = $number;
        }
        @result = @array;
}
```

```
$output = @parameters[2]."_report.txt";
open ( CENTRFREQ_REPORT, ">$output" ) || die " $output $! \n ";
foreach $variable (@variable)
{
        $variable =~ s/^          //;
}
#print "@variable \n";

format CENTRFREQ_REPORT =
+-------------------------------------------------+
|                  Oscillator Freq                |
+-------------------------------------------------+
|      Model Variable          ||  Frequency(Hz) |
+=================================================+
|^>>>>>>>>>>>>>>>>>>>>>>>>>    ||                 |
 $variable[0]
|~^>>>>>>>>>>>>>>>>>>>>>>>>    ||    @>>>>>>>>>> |
 $variable[0],$CenterFreq[0]
|-------------------------------------------------|
|^>>>>>>>>>>>>>>>>>>>>>>>>>    ||                 |
 $variable[1]
|~^>>>>>>>>>>>>>>>>>>>>>>>>    ||    @>>>>>>>>>> |
 $variable[1],$CenterFreq[1]
|-------------------------------------------------|
|^>>>>>>>>>>>>>>>>>>>>>>>>>    ||                 |
 $variable[2]
|~^>>>>>>>>>>>>>>>>>>>>>>>>    ||    @>>>>>>>>>> |
 $variable[2],$CenterFreq[2]
|-------------------------------------------------|
|^>>>>>>>>>>>>>>>>>>>>>>>>>    ||                 |
```

```
+-------------------------------------------------+
|                  Oscillator Freq                |
+-------------------------------------------------+
|      Model Variable          ||  Frequency(Hz) |
+=================================================+
|       CMOSN TOX 1.0000E-07   ||                 |
|     Mbreakp TOX 1.0000E-07   ||      3.440E+06 |
|-------------------------------------------------|
|       CMOSN TOX 6.4651E-08   ||                 |
|     Mbreakp TOX 6.4651E-08   ||      3.760E+06 |
|-------------------------------------------------|
|       CMOSN TOX 2.7773E-08   ||                 |
|     Mbreakp TOX 2.7773E-08   ||      3.960E+06 |
|-------------------------------------------------|
```

Figure 6. Perl Program and Output

IV Statistical Analysis

The objective of the project is to determine if the oxide thickness of all the MOSFETs in a ring oscillator has an effect on the central frequency of the ring oscillator. PSPICE collected data was extracted using a Perl program and was analyzed using Matlab. A graph of the data is shown in Figure 7.

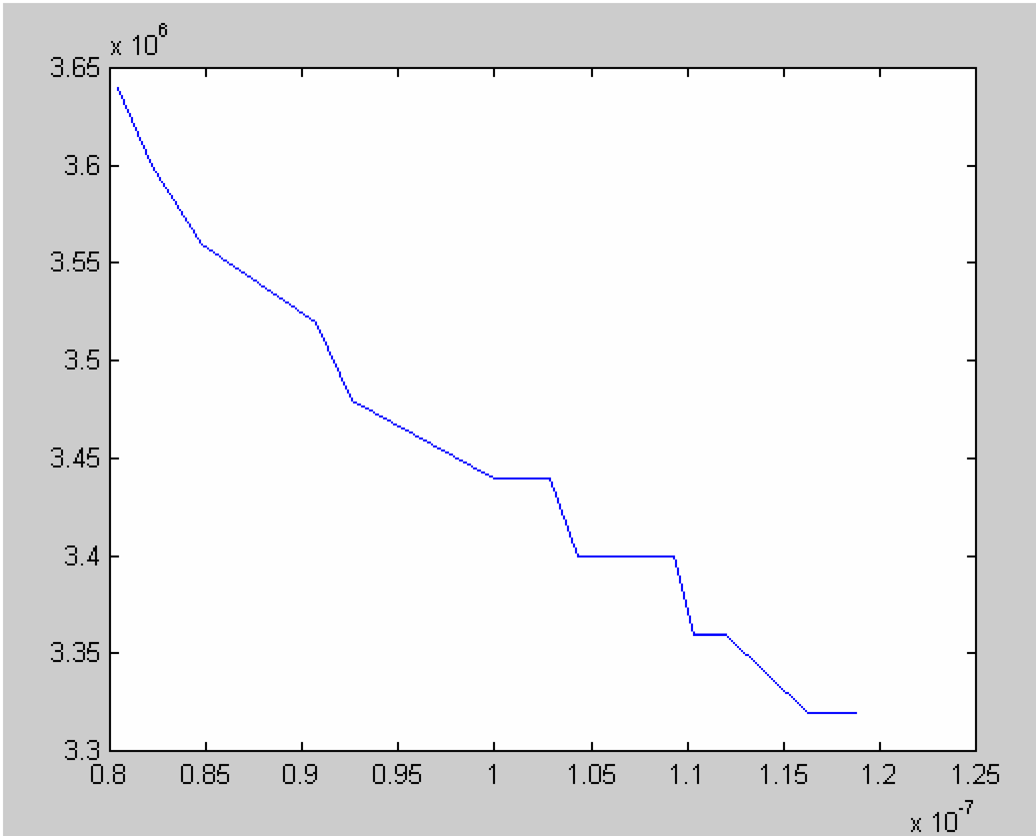


Figure 7. Frequency versus Tox

The data was modeled as a straight line using the statistical analysis functions in Matlab. The basic relationships are shown in Figure 8.

$$a = \bar{y} - b \cdot \bar{x}$$

$$b = \frac{S_{xy}}{S_{xx}}$$

$$S_{xy} = \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})$$

$$S_{xx} = \sum_{i=1}^{n} (x_i - \bar{x})$$

$$\hat{Y} = a + b \cdot x$$

Figure 8. Least Squares Fit Equations

The T-test was used to determine the degree of confidence of a relationship between the frequency and the parameter change. The basic relationships are shown in Figure 9.

$$S_e^2 = \frac{S_{yy} - (S_{xy})^2 / S_{xx}}{n-2}$$

$$t = \frac{(b - \beta)}{S_e} \sqrt{S_{xx}}$$

Figure 9. T-test Relationships

The null hypothesis was $\beta = 0$ with a 95% confidence interval. Results of t-test were

S_e=1.59e+004
b = -7.68e+012
t = -25.9
n = 20
df = 18
$t_{\alpha/2}$=2.101

Therefore, the null hypothesis was rejected since -25.9 < -2.101, concluding that there is a relationship between oxide thickness and frequency.

V Conclusions

From their comments, the students that went through this exercise seem to have come away with a better appreciation for the capabilities of the different software packages. More importantly, they seem to now realize that much more can be done by using the strengths of multiple software packages.

References

1. Sigi St.Helene, Kelly DeShields and Ben Brackett, "Ring Oscillator," Internal Report ECE TTU, Lubbock, Texas, October, 2002
2. John Beshears and Noah Wilkes, " Data Extraction With Perl," Internal Report ECE TTU, Lubbock, Texas, November, 2002
3. Kirt Williams and David McDonald, " Statistical Analysis of Ring Oscillator," Internal Report ECE TTU, Lubbock, Texas, November, 2002

Micheal E. Parten is a Professor of Electrical Engineering at Texas Tech University. Dr. Parten has conducted research and published in the areas of instrumentation, control, modeling and simulation of a variety of systems, including hybrid electric vehicles. Dr. Parten has served for over sixteen years as the Director of the Undergraduate Laboratories in Electrical Engineering.