SPICE Model Verification Test Suite

David H. Smith

TriQuint Semiconductor, Inc.
3625A Murray Blvd.
Beaverton, OR 97229
(503)644-3535 ext. 252
FAX: (503)644-3198
davids@tqs.com

revised: February 23, 1995

Abstract

This document describes the TriQuint Modeling test suite. Elemental "measurements" performed on individual FETs were used to characterize TriQuint's Pspice implementation of TOM-2. results provide a data-set against which other simulators can be verified.

The basic equations which define TOM-2 are described in a separate document: *TOM-2: An Improved Model for GaAs MESFETs*.

Contents

1 Introduction

The Analog Model Test Suite has been developed for the purpose of insuring compatibility of device models where these models are implemented in various circuit simulators. For the most part, these simulators are "Spice-like" due to the preeminence of Spice in this application.

These tests have been developed for the specific purpose of verifying implementations of TriQuint's GaAs MESFET model, TOM-2, to various simulators. The tests are based on TriQuint's experience in attempting to do something similar with a predecessor model, called simply TOM or TOM-1. We hope with this test suite to facilitate porting of TOM-2 and avoid the major difficulties encountered with the earlier attempts.

The test suite consists of a set of standard simulations along with the expected output results. Models for representative FETs in enhancement and depletion mode are provided.

The test circuits themselves are provided as generic Spice-compatible netlist templates. Implementation specific refinements such as expression evaluation, or flexible node naming conventions have been avoided.

The results are likewise presented in standard ACSII format so that they can be used for generic comparison between implementations in different simulators. Most simulators on the market can export data in some type of ASCII format, which can then be fairly easily converted to our generic formats using a short AWK or Perl program, or even a text editor. Alternatively, the ASCII format reference data can be easily coerced into a form suitable for import to a spread sheet or to a format acceptable to the specific-simulator.

A weakness of our approach is that using only such low-level simulations we are likely to miss weaknesses in simulating real circuits—convergence problems and so forth. This has been done to limit the scope and size of the test suite and also because with more complex circuits, it is more difficult to separate model-related problems with the models from problems in the core simulator. Therefore, the test suite provides information that will help to verify the accuracy of the TOM model implementation for a specific simulator, but provides little to insure the overall speed and robustness of the simulator itself.

2 The Tests

The tests themselves are quite simple and fall into two classes:

- A DC bias test that checks the gate and drain current over a matrix of bias conditions for V_{gs} and V_{ds}.
- An AC test to check the Small signal AC parameters over a similar, but smaller matrix.

2.1 Invertible FETs

All of the above tests are performed for a variety of temperatures and device widths to check proper scaling and temperature dependencies. Note that the TOM-2 model is supposed to be invertible: that is the model should give the same results if the FET being simulated is swapped source-for-drain. Actually, only the core or intrinsic FET is completely symmetrical, so that a side condition for being invertible is that $R_s = R_d$.

We have not explicitly provided the simulation results for inverted FETs since they are identical to the non-inverted results.

2.2 DC Bias Tests

The DC bias test circuit is illustrated in figure 1. This simple circuit incorporates three FETs of various widths. The voltage sources, and temperature are varied over a range of conditions using statements that are built-in to most SPICE-like simulators.

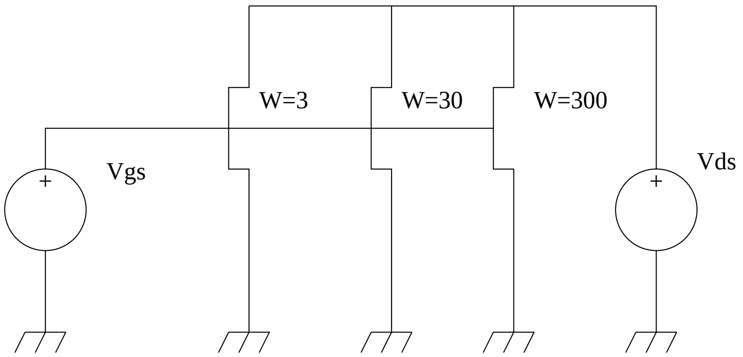

Figure 1: DC bias test circuit.

The complete test suite should also include this same test with the FETs inverted, that is with drain and source connections swapped.

The Pspice circuit file `iv.cir` was used for this test. This file is mostly compatible with SPICE2G6 syntax and therefore quite portable. It is provided in the Appendix A.

Note that it is important to check the gate current as well as the drain current to insure consistency.

2.3 AC Tests

The AC test checks small signal parameters and the capacitances. The circuit is shown in figure 2. This circuit was chosen for simplicity over the standard *S*-parameter measurement circuit[1]. However, S-parameters are easily computed if desired using the formulas given in the figure.

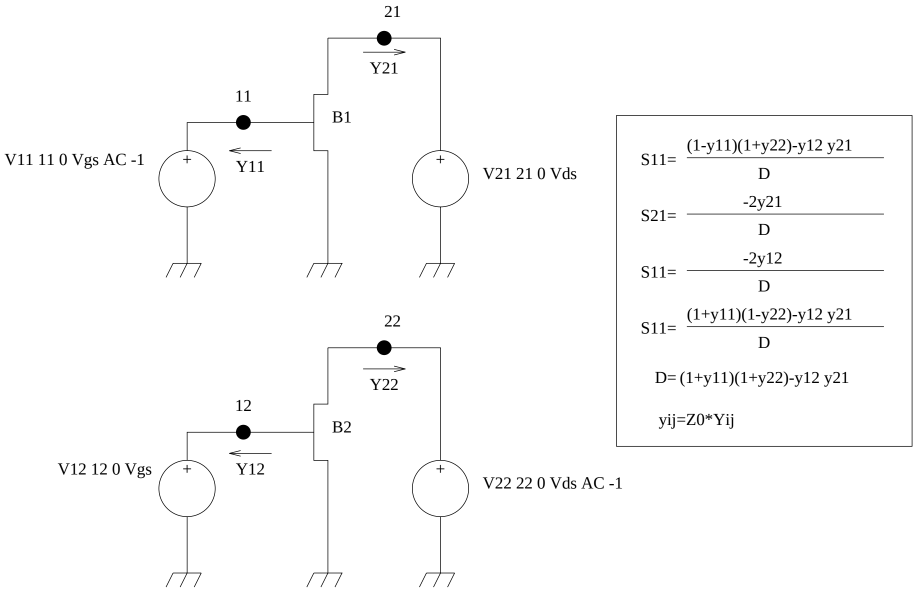

Figure 2: Small signal AC or Y-parameter test circuit. If desired, the admittances can be converted into S-parameters using the formulas in the insert. Depending on the simulator in use, it may be possible to do this using controlled voltage sources or a post processor program.

This test must range across frequency as well as bias, FET size, and temperature and so may generate a large amount of output data. The test matrix was therefore abbreviated in order span a reasonable range of combinations.

Simulators such as PSPICE which closely follow the SPICE2G input syntax do not provide a uniform mechanism for varying all of the parameters combinations we would like to provide. Therefore, we have provided a short Perl script to specify the various possible combinations. The listing is provided in appendix B. Field testers may wish to implement this behavior using macro or pre-processing features specific to the simulator under test.

This Perl script generates a single circuit file containing a sequence of simulation specifications. Most SPICE-like simulators can digest this file. Each individual simulation is based on a "template" and instructions for filling it and writing it to the circuit file. The Perl script is provided in appendix B. Even for the reader not versed in Perl, the specifications contained in this short script should be fairly clear.

2.4 About the Parameter TAU.

The parameter TAU was originally used for in the small signal FET model to represent a pure time delay in the drain current with respect to changes in the gate voltage. Such an element can be problematic to implement in a simulator for transient analysis. Though MicroSim has adapted a method to implement this feature in Pspice, we are not aware of a method in the open literature to do so.

Recognizing that it may be difficult to exactly reproduce the Pspice results, we have therefore supplied AC results for a specific value of TAU (3.8ps) and also for a similar circuit with TAU=0.

3 Transient Analysis

Simulators such as Spice use different algorithms for performing AC and transient analysis. It is possible for an incorrectly implemented model to produce inconsistent results for AC analysis and transient. For example, one may find the the small signal gain of an amplifier stage using either AC analysis or transient analysis with a low-amplitude sinusoidal voltage source. For a correct implementation, these two methods should produce identical results.

We have not provided specific results for comparing AC and transient analysis with low-amplitude voltage sources. However, it is assumed that the simulator will be checked for such consistency as with the inverted FET.

4 Comparison of Results and Output Data Formats

Assuming that one has done a new implementation of the TOM-2 model in a simulator—let's call it SimuleX for now, there immediately arises the question of how to compare the results for accuracy. Several choices are possible including:

1. Convert the Pspice results to SimuleX output format and use the SimuleX post-processor.

2. Convert the SimuleX output format to Pspice format and use the Pspice post-processor (probe/CSDF).

3. Convert both Pspice results and SimuleX results to a common format and compare using a tool belonging to neither Pspice or SimuleX. For example a spread sheet or general-purpose plotting package.

We believe the third alternative is easiest to implement and has the advantage of a providing a level field and common format in comparing simulators. With this approach, the model verification occurs outside of the SimuleX environment.

Unfortunately, there is probably no standard tool or data format which every implementor will have accessible. In this test suite, we have gone only part way to alleviate this: Along with the raw Pspice output files, we have provided the data in a simplified standard tabular format (`.tab` files). Unfortunately, most plotting software will not be able to plot this data directly, but the tabular format should be fairly simple to link or convert to general purpose plotting program.

At a later date, we may provide a Perl script to link this format to a freely distributable plotting program such as `gnuplot`.

5 What is Provided

The test suite is normally provided as a compressed tar file, `tom2test.tar.Z`. The following files are provided in the test suite:

File Name	Description
`README`	Explanation.
`tom2.ps`	Summary of TOM-2 device equations (postscript).
`testsuit.ps`	This document (postscript).
`tom2.mod`	Include file with device models.
`tom2a.mod`	same as `tom2.mod` except with `TAU`=0.
`iv.cir`	Input circuit file for DC bias point simulation.
`iv.out`	Pspice output from DC bias point simulation.
`ivget.pl`	Perl script to create I-V table from .out file.
`iv.tab`	Tabular ouput from DC bias point simulation.
`ypargen.pl`	Perl script to generate AC (Y-parameter) simulation file.
`ypargen.out`	Pspice output from AC (Y-parameter) simulation.
`ypargen0.out`	Pspice output from AC (Y-parameter) simulation (TAU=0).
`yparget.pl`	Perl script to create Y-parameter table from .out file.
`ypargen.tab`	Tabular output from AC (Y-parameter) simulation.
`ypargen0.tab`	Tabular output from AC (Y-parameter) simulation (TAU=0).

References

[1] Paul W. Tuinenga. *Spice: A guide to Circuit Simulation and Analysis Using Pspice*. Prentice Hall, 1992.

A TOM-2 reference models

```
.model Efet GasFet (level = -1
+       Alpha   =  5.69
+       Beta    =  0.000323
+       Gamma   =  0.058773
+       Delta   =  168
+       Vto     =  0.084
+       Q       =  2
+       Rs      =  555
+       Rd      =  555
+       Rg      =  1.5
+       Is      =  4.55e-15
+       N       =  1.17
+       Cgs     =  12.6e-16
+       Cds     =  2.9e-16
+       Cgd     =  2.82e-16
+       Vbi     =  0.8
+       Vmax    =  0.7
+       Tau     =  4.6e-12
+       Eg      =  0.9
+       Xti     =  2.34
+       Vtotc   =  -0.00098
+       Betatce =  -0.27
+       Trs1    =  0.0021
+       Trd1    =  0.0021
+       NG      =  1.25
+       ND      =  0.2
+       Alphatce=  -0.18
+       Gammatc =  0
+       CgsTCE  =  -0.00005
+       CgdTCE  =  0.0002
+       VbiTC   =  -0.0006
+ )

.model Dfet GasFet (level = -1
+       Alpha   =  3.6
+       Beta    =  0.00014
+       Gamma   =  0.083
+       Delta   =  270
+       Vto     =  -0.625
+       Q       =  2
+       Rs      =  518
+       Rd      =  518
+       Rg      =  1.5
+       Is      =  3.0106e-15
+       N       =  1.17
+       Cgs     =  1.45e-15
+       Cds     =  3.12e-16
+       Cgd     =  2.21e-16
+       Vbi     =  0.8
+       Vmax    =  0.7
+       Tau     =  3.8e-12
+       Eg      =  0.9
+       Xti     =  2.34
+       Vtotc   =  -0.000822
+       Betatce =  -0.168
+       Trs1    =  0.00224
+       Trd1    =  0.00224
+       NG      =  0.995
+       ND      =  0.1
+       Alphatce=  -0.0143
+       Gammatc =   -0.000033
+       CgsTCE  =  0.0003
+       CgdTCE  =  0.0002
+       VbiTC   =  -0.001
+ )
```

B DC circuit file: `iv.cir`

```
* SPICE2 compatible IV test circuit.
.inc tom2.mod
b3 1 2 0 Dfet 3
b30 1 2 0 Dfet 30
b300 1 2 0 Dfet 300

vds 1 0
vgs 2 0

.temp -55 25 75 125
.dc vds 0 4 .1 vgs -1.0 1.0 .1
.options acct opts limpts=1000 width=132
.print DC v(2) ig(b3) id(b3) ig(b30) id(b30) ig(b300) id(b300)
.probe/csdf
.end
```

C AC circuit file generation Perl script: `ypargen.pl`

```
# Perl script to generate Y-parameters circuit matrix.

$temp=" 25";
$width=300;
$vgs=0;
foreach $vds (0,.2,.4,.6,.8,1.0,1.5,2.0,3.0,4.0,5.0) {
        &print_one_circuit;
}

$vds=1.5;
foreach $vgs(-1,-.8,-.6,-.4,-.2,0,.2,.4,.6,.8,1.0) {
        &print_one_circuit;
}

$vgs=0;
foreach $width (3,30,300,3000) {
        &print_one_circuit;
}

$width=300;
$vds=1.5;
$temp=" -55 25 75 125 175";
foreach $vgs (-.25,0,.5) {
        &print_one_circuit;
}

sub print_one_circuit {
print <<"EOF";
* W=$width Vds=$vds Vgs=$vgs

.inc tom2.mod
b1 21 11 0 Dfet $width
v11 11 0 $vgs AC -1
v21 21 0 $vds

b2 22 12 0 Dfet $width
v12 12 0 $vgs
v22 22 0 $vds AC -1

.temp $temp
.print AC ir(v11) ii(v11) ir(v21) ii(v21) ir(v12) ii(v12) ir(v22) ii(v22)
.op
.ac lin 51 1 50e9
.options acct opts limpts=1000 width=132
.end
EOF
}
```