Simulation Standard

SILVACO International

A Journal for Circuit Simulation and SPICE Modeling Engineers

Parallel SmartSpice: Fast and Accurate Circuit Simulation Finally Available

Introduction

Circuit simulation is a necessary everyday tool to circuit designers who need to constantly verify and debug their circuits during the design process. As engineers face larger, more complex designs and tighter project schedules, fast SPICE simulation with no loss in accuracy has become a necessity. Simulation indeed accounts for a large portion of the time spent in the design and optimization of a new circuit.

Several approaches are being used in SPICE to speed up circuit simulation including:

- Table lookup methods: instead of evaluating the model equations for all the devices as a regular SPICE simulator would do, this approach evaluates currents and charges as a function of the node voltages and stores them in a lookup table. This table is used afterwards to get the corresponding values for a specific device with specified node voltages.
- Circuit partitioning: these methods take advantage of the latency in some parts of the circuit to apply different time steps to different sub-circuits during a timing analysis. This leads to an event-driven simulation where only a particular event (voltage/ current threshold) would trigger the re-evaluation of a sub-circuit outputs.

These methods can significantly speed up the simulation of a circuit, but they both suffer from a loss of accuracy. Depending on the particular application and circuit specification, the loss of accuracy may or may not be acceptable. At a final stage of test and verification of a circuit, it usually is not acceptable.

In order to keep SPICE level accuracy, the approach chosen to speed up SmartSpice is parallelization. Several processors will cooperate to simulate one circuit in a fraction of the time necessary for one processor to do the same job. The only prerequisite is the availability of a multiprocessor computer. There are two affordable ways to make multiprocessing capability available:

- through a compute server: small departments can set up a compute server hosting 4 to 16 processors with a fair amount of memory to serve as a central computing power. Access would be granted through X terminals or PCs or low-end workstations.
- through desktop computers: most desktop workstations nowadays(and even PCs) already host two or more processors readily available for any parallelized application.

The compute server approach has some advantages over the desktop workstation approach. First, a central compute server can accommodate a large number of people if they don't all need its computing power at the same time. Whereas a desktop workstation is usually meant for individual use regardless of the actual machine load over a long period of time. Second, the large amount of memory in a compute server can be of a great help in simulating very large circuits, even if the application remains sequential and uses only one single processor. The same applies for any kind of memory-bound applications that one needs to run. Third, multiprocessor computers from some hardware vendors seem incapable of scaling a parallel application to the number of available processors because of background processes. For instance, a parallel application using 2 processors runs faster on a 4 processor computer than on a 2 processor computer.

Continued on page 2....

INSIDE

Volume 8, Number 5, May 1997

In the next section, we will outline the structure of a SPICE simulator with emphasis on the time consuming tasks. Then we will describe the methods applied to parallelize SmartSpice and present some experimental results.

Workload chart

Figure 1 shows a simplified flowchart that describes the order in which the functions contained within each device model are called. The "setup" function is called at the very beginning of the simulation of a circuit. It initializes the model parameters to their default values and allocates storage for the assembled matrix, taking into account the model's internal nodes.

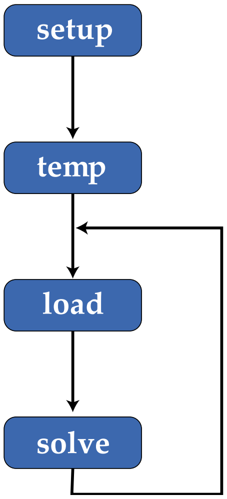


Figure 1. Simplified flowchart of a transient simulation.

The "temp" function is called at the beginning of each analysis to evaluate temperature dependent variables and some time independent variables too. These will be used during the time loop that follows.

As for the "load" routine, it performs the actual model evaluation for every single device in the circuit. After evaluating the model equations, the result is assembled in a sparse matrix and a right hand side. Each row of the matrix corresponds to a node in the circuit or a model's internal node.

Once the sparse matrix and the right hand side are assembled, a linear system is solved in the "solve" routine using an LU decomposition and a backward+forward triangular solves.

During a transient simulation, the "load" and "solve" routines account for almost the entire execution time of the simulation. Depending on the size of the circuit and the particular device model used, the proportion of time spent in the load varies between 20 and 80 percent of the total execution time. Usually, the larger the circuit is, the longer the time spent in the "solve".

Parallelization

To parallelize SmartSpice, we focus on the two main time consuming tasks: the matrix assembly ("load") and the linear system solution ("solve"'). This will leave a fraction of sequential code in SmartSpice, like the preprocessing, post-processing, all I/Os etc. It is of utmost importance to keep this fraction of sequential code as small as possible in order to get acceptable overall performance. Figure 2 shows the maximum theoretic speedup achievable by any code depending on the fraction that remains sequential and assuming an optimum efficiency in the parallelized section of the code.

Because of the large amount of data involved and the inherent sequentiality in the waveform calculation, SmartSpice is an application that shows data parallelism and not functional parallelism. Therefore, an SPMD (Single Program Multiple Data streams) paradigm is better suited than a Fork and Join paradigm for parallelization.

Due to the tightly coupled nature of the computations and the rather small time spent in one iteration during transient analysis in most circuit sizes, it is highly recommended to keep the communication between collaborating processors as low as possible. Therefore, we chose a tightly coupled architecture, namely a shared memory architecture to implement the SPMD paradigm. On the one hand, this will allow the processors to access a common data space for communication instead of passing messages through a communication channel. On the other hand,communicating through a shared memory requires explicit synchronization, unlike passing messages where synchronization is implicit and the execution more asynchronous. Nevertheless, the gain is in the access to communicated data: the first one is through a system bus, the second one is through a much slower communication network.

On generic Unix-like based operating systems, there are

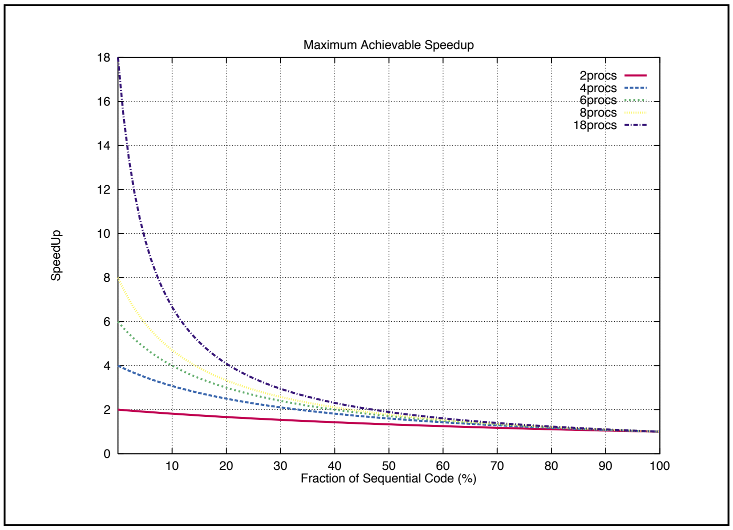


Figure 2. Theoretical achievable speedup with a fraction of sequential code.

two ways of implementing a parallel paradigm: multiprocessing and multi-threading. Actually, a thread is only a light weight version of a process. Several processes on the same computer share the same physical address space but have separate virtual address spaces, protected by access security mechanisms. Threads on the contrary share both physical and virtual address spaces. As a plus,threads inherit the parent's variables whereas tasks don't, making it necessary to communicate or recompute these variables for each task. Thus, a multi-threaded implementation is much more efficient than a multi-tasked implementation. Recall that PVM and MPI, two portable parallel development environments, use tasks and not threads in their shared memory implementation.

To summarize, parallel SmartSpice relies on an SPMD paradigm on shared memory parallel computers and uses the POSIX 1003.1c norm threads,which makes it portable across platforms from different hardware vendors.

Parallelization of the assembly

The parallelization of the assembly phase is rather straightforward. The circuit is partitioned into as many chunks of equal size as there are processors in order to achieve a good load balance. Each chunk is then assigned to a processor for model evaluation. Code transformation was necessary in the model evaluation to account for multiple write access conflicts that may result when two or more processors are updating devices sharing a common node. Thanks to a combination of optimization techniques, including write caching and inner loop blocking, the use of synchronization locks has been highly decreased for optimum performance. Although the flow of control has been altered to some extent, the computations involved in parallel SmartSpice are exactly the same as the original SmartSpice leading to the same accuracy.

Parallelization of the solver

The parallelization of the solve routine is much more complex. Parallelizing an LU decomposition with pivoting is still an active research field. To apply state of the art knowledge in this area, we have implemented a parallel LU decomposition based on level scheduling of the elimination tree with events-based synchronization. We first factorize symbolically the matrix after it has been renumbered for sparsity and better parallelism. Afterwards,we build the elimination tree that describes computational dependency information between the columns of the matrix. Once this is done, the nodes of the tree are assigned to levels such that no dependency appears between nodes on any one level. This guarantees that all nodes (corresponding to tasks) in the same level are entirely independent and can be eliminated in parallel. Then, the nodes are numbered according to a bottom-up walk of the elimination tree and arranged in a task queue. Processors pick up columns to process from that task queue and make sure the dependency constraints are satisfied before performing any actual computation. Instead of using a barrier synchronization at each level of the elimination tree as similar algorithms in the literature may suggest, we improved the performance of the algorithms by switching to an event-based synchronization. This allows for a more asynchronous execution and exhibits a better parallelism. This is essential because of the extremely sparse nature of spice-like matrices.

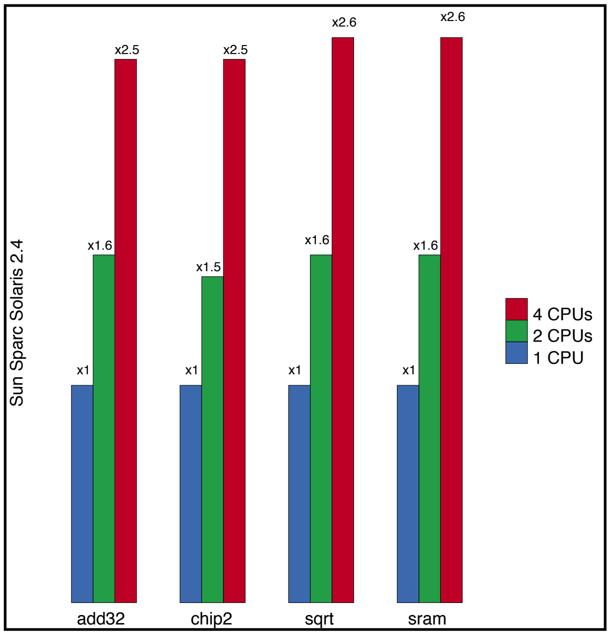


Figure 3. Parallel SmartSpice speedup on 2 UltraSparc CPU's using the MCNC benchmark suite

Experimental Results

As a first step, we present here performance results on several circuits using Silvaco's implementation of the BSIM3v3 transistor model. The circuits are part of the MCNC benchmark suite. Their sizes range from 288 to 18816 MOS devices. The first experiments have been carried out using 2 and 4 processors of a SUN sparc Server running Solaris 2.4 As Figure 3 shows, the average simulation speed exceeds 1.5 times that of a sequential execution. Therefore, a substantial saving of over 33% of simulation time is induced by the use of only one extra processor over 60% of the simulation time is saved when using 4 processors. Further results relating our latest experiments on several hardware platforms hosting up to 8 processors will be publicly available shortly.

Conclusion

In an effort to reduce circuit development cycles, Silvaco introduces the first Parallel version of SmartSpice. This product is available on Sun servers running Solaris 2.4 and 2.5, SGI servers running IRIX 6.2 and 6.4 and HP S/X/V-Class servers running HPUX 11.0. This paper reports an average speedup of 1.58 on 2 processors and 2.55 on 4 processors for a broad range of circuits out of the MCNC benchmark suite. The precision of the original SmartSpice is maintained while the simulation speed is significantly increased.

SmartSpice Integration into the Cadence Design Framework

Since 1992, it has been possible to use SmartSpice within the Cadence Design Framework through the purchase from Cadence of an option known as the "HSPICE Plug". Unfortunately, the level of integration afforded by this option has been limited. With the advent of version 4.4 of Cadence Design Framework II, Cadence has made available their Open Analog Simulation Integration Socket (OASIS) interface, which provides access to internal subroutines within the Analog Artist Electrical Design System. Working closely with the OASIS group, Silvaco has taken advantage of this new interface, along with the Cadence SPICE Socket, to create a new and substantially tighter integration of SmartSpice with Analog Artist.

Under this new integration scheme, SmartSpice now explicitly appears in the Analog Artist simulator menu and Analog Artist now generates fully compatible SmartSpice netlists. Simulation within the Analog Artist environment is transparently performed by SmartSpice in batch mode,with full support for saving, plotting and marching variables. SmartSpice itself now generates results in Parameter Storage Format (PSF), which may be viewed in the Cadence Waveform Window. Cross-probing and back-annotation are fully supported via the Composer Design Entry tool.

This integration is now available in SmartSpice 1.5.2.R. However, users will need to purchase an OASIS Simulation Interface license from Cadence. The installation process is carried out with the command "SmartSpice -install", and is fully automated and self-verifying. A complete model library, SmartSpiceLib, is also distributed with SmartSpice. The library can be referenced via a "cds.lib" file, generated automatically with the command "smartspice -configure."

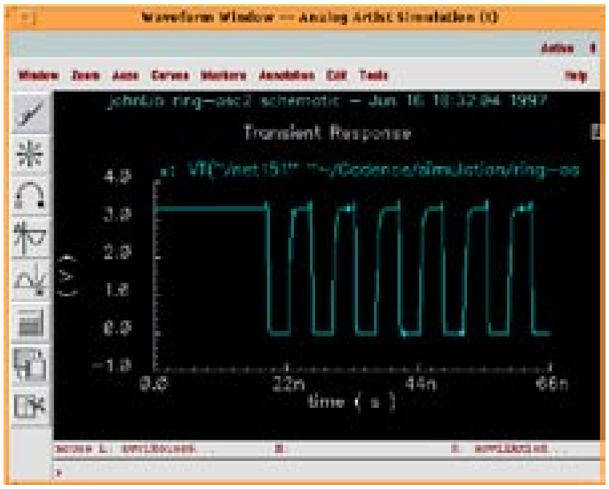

Waveform Window — Analog Artist Simulation (1)
Active
Window Zoom Axes Curves Markers Annotation Edit Tools
Help
johnLib ring—osc2 schematic — Jun 16 18:32:04 1997
Transient Response
VT("/net151" "~/Cadence/simulation/ring—os
4.0
3.0
2.0
1.0
0.0
-1.0
(V)
0.0
22n
44n
66n
time (s)

Figure 2. SmartSpice PSF output in the Waveform window.

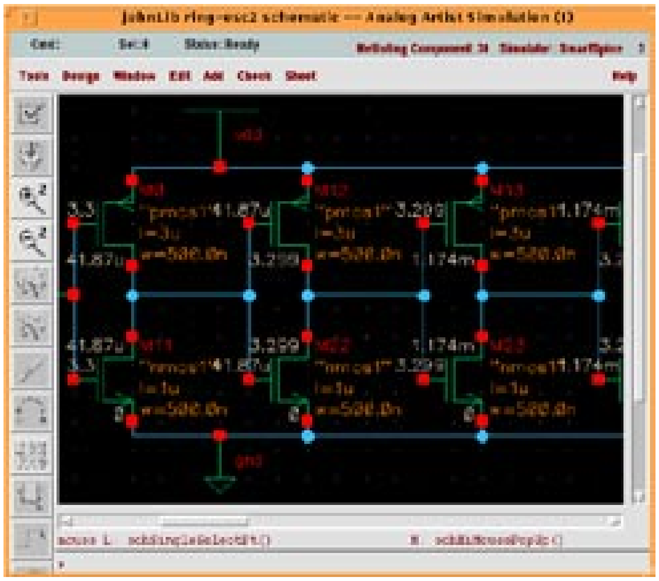


Figure 1. Schematic with SmartSpice components.

As an example of the SmartSpice/Analog Artist integration, Figure 1 shows a ring oscillator constructed in the Composer Design Entry tool from components of the SmartSpiceLib model library. This particular view is back-annotated with a set of node voltages at a single time point. In Figure 2 a plot of the output voltage as plotted in the Waveform Window is displayed and in Figure 3 a plot of the corresponding signal as plotted in SmartSpice is shown for comparison.

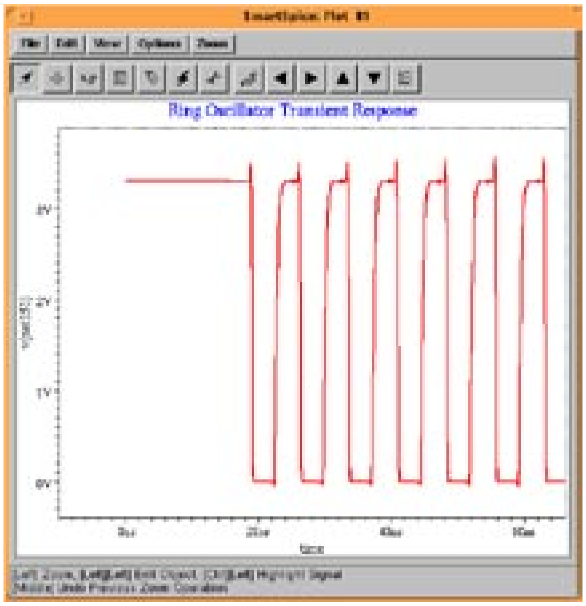

SmartSpice: Plot
File Edit View Options Zoom
Ring Oscillator Transient Response
3V
2V
1V
0V
0s
20n
40n
60n
time

Figure 3. Corresponding signal plotted directly in SmartSpice.

SmartSpice Integration into the VIEWlogic Framework

Introduction

The SmartSpice - VIEWlogic integration allows the replacement of HSPICE™ by SmartSpice for analog simulation in the VIEWlogic framework. It uses VIEWlogic's ViewScript language to add the option of using SmartSpice in the ViewDraw Schematic editor and the SpiceLink netlister. The integration can be made on an installation wide basis by changing the master copy of VIEWlogic or on a per user basis by installing the integration elsewhere and changing the environment of users.

Main features

The correct installation of the SmartSpice - VIEWlogic integration should result in the following changes to the VIEWlogic software.

1) The Power view cockpit has an extra toolbox called "Silvaco" - This groups together the set of VIEWlogic tools that you are needed for Analog simulation, and adds a button to allow SmartSpice to be run directly from the VIEWlogic framework.

2) ViewDraw has an extra menu, called "SmartSpice" - This provides an easy to use interface to SmartSpice to netlist the Schematic, add simulation parameters and start the simulation, all without leaving ViewDraw.

3) The Analog Netlist utility (SpiceLink) has an extra button marked "SmartSpice" - For those that are used to using HSpice™ in the VIEWlogic framework, this provides a button that will allow users to continue to use the same work process and just swap SmartSpice where HSpice™ was previously used.

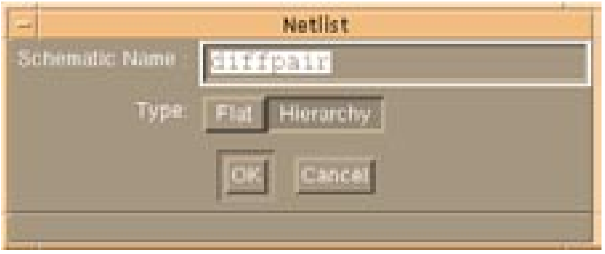
Example Usage

After edits to the schematic are written, the netlister must be run before any simulation can be done. To do this select "Netlist" from the SmartSpice menu in ViewDraw.

A dialog will appear, it should have the correct name of the project in the text field, and also gives the option of "Hierarchical" or "Flat" netlisting.

Make sure the project name is correct, if not change it. Choose Flat or Hierarchical and press OK.

After the netlist has been completed, simulation parameters can be added. Users can add parameters for AC, DC, Distortion and Transient analysis from this menu. When an analysis type is selected from the menu, an appropriate dialog will appear and allow the user to enter the values for the parameters. More than one type of analysis can be done for each simulation.



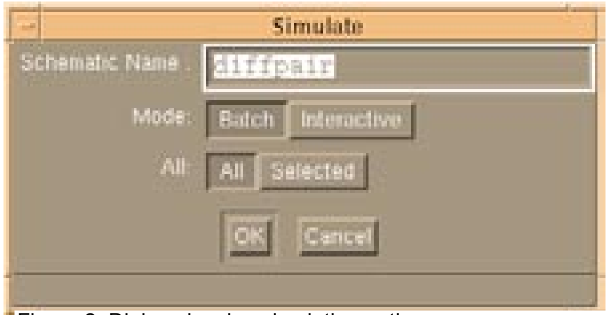
Netlist

Schematic Name : diffpair

Type: Flat

Figure 1. Dialog showing netlist options.

Next select "simulate" from the SmartSpice menu. The following dialog will be displayed:



Simulate

Schematic Name : diffpair

Mode: Batch

All: All

Figure 2. Dialog showing simulation options.

Again check that the project name is correct, if not correct it. The option of running SmartSpice interactively or in batch mode is then selected. Unless there are problems with the simulation it is best to run in batch mode as SmartSpice will automatically create all the output files with the correct extensions.

Users can select nets of interest and to print information about those nets by selecting on the schematic and choosing the "Selected" option on the simulate dialog. Choosing "save all" will result in all of the vectors that SmartSpice calculates to be saved in the output files.

The simulation will run and the output from the simulation is presented in a popup dialogbox. When the simulation is finished the log can be inspected.

The next stage is to startup ViewTrace. When one of the files generated by the simulation is opened, one file is generated by each type of analysis, the output traces of the vectors that were chosen to save will be seen.

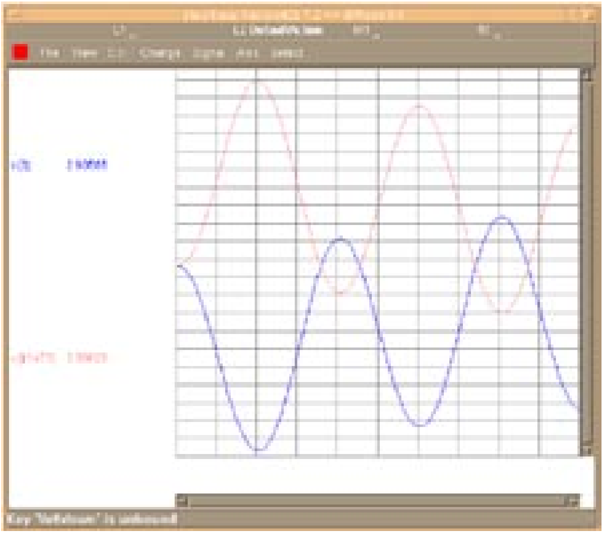


Figure 3. Simulation output in Viewtrace.

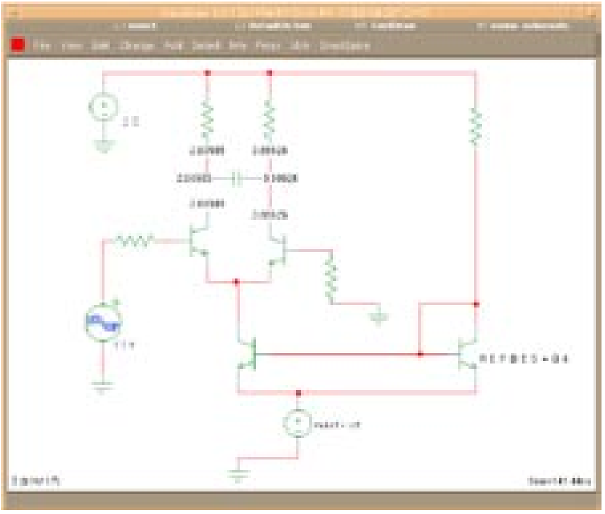


Figure 4. Viewdraw schematic with simulation values back-annotated.

The values for the simulation can be back annotated into the schematic by selecting the "back annotation" option. To do this in VIEWlogic select "Annotate" from the Change menu and press the check box. Now select a trace to follow, then use the middle mouse button to select a time. The values from the trace will then appear on the schematic.

Users can cross probe from the schematic to ViewTrace. To do this highlight the net that you are interested in, within ViewTrace choose FromSchem... in the signal menu and the select show. The trace for the selected net will be displayed.

Detect Process Variations Using Analysis of Variance and Wafer Map

Introduction

Due to the inherent process variation, the yield of ICs is always lower than 100 percent. E-test parameters are used to monitor possible process variation or deviation. Using the Wafer Map feature from SPAYN can help to intuitively look at the distribution of parameter values on a single wafer for process engineer, integration engineer or device characterization engineer. Moreover, a useful analysis method called Analysis of Variance has also been introduced to detect the variation of parameter values from different test structures.

Wafer Map Display of Parameter Values

Suppose a SPICE level3 data set for CMOS has been extracted from five locations of a wafer. These could be test structures. After loaded into SPAYN, an attribute search can be performed and a parameter search is followed to filter out possible outliers. The Wafer Map window can now be invoked. VTO_N for NMOS and WD_N for NMOS are examples. Choose an appropriate way to include the die location information. If the parameter is extracted using Silvaco's UTMOST, the die location information can be loaded by choosing the "UTMOST" button on the "Load" window. Go to the parameter combo box and choose the parameters of interest to display, i.e., VT0_N and WD_N. The Wafer Map helps to identify certain patterns of value distribution when parameter values have been extracted from numerous locations. In this case, other than the Wafer Map display, it would be beneficial to have some statistical calculations to verify the parameter values are the same for different wafers at some specific location.

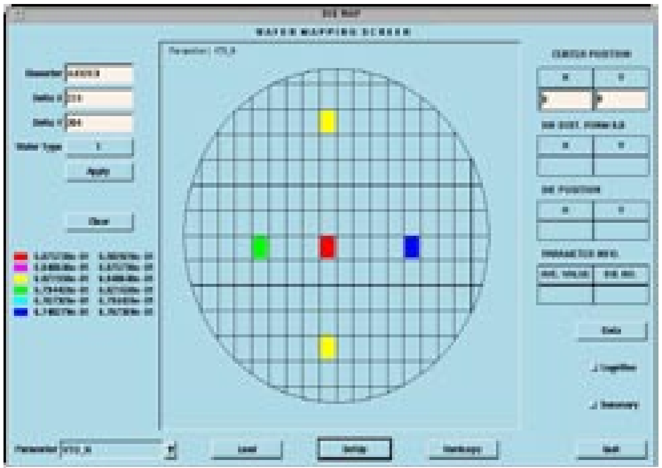


Figure 1. Wafer Map display of parameter values.

Analysis of Variance Table for Detecting Variation in E-test Parameter Values

In this section a very useful statistical analysis method will be introduced that has many potential applications in understanding inherent semiconductor process variations. For example, to understand how certain process step would affect an E-test parameter, several setups for that particular process step could be evaluated and then apply the analysis of variance to the result to see if the difference in this particular process step has an influence on the parameter measured at the E-test step.

This method is used here to test if the parameter value VTO_N and WD_N from five test structures are the same statistically. In the VTO_N example forty wafers were sampled. The following is the One Way ANOVA table for this analysis.

Source of Variation	SS	df	MS	F statistics
Between Treatments	0.000991461	4	0.000247865	0.373906
Error(within treatments)	0.0238647	36	0.000662909	
Total	0.0248562	0		

Set the significance level to 0.05 and since F0_statistics = 0.373906 < F_percentile = 2.60597, it is seen that the VTO_N values at these five locations are not significantly different.

Repeating this for wafer one to wafer eleven provides the ANOVA table for WD_N.

Source of Variation	SS	df	MS	F statistics
Between Treatments	2.66626e-14	4	6.66565e-15	4.32388
Error(within treatments)	2.00407e-14	13	1.54159e-15	
Total	4.67033e-14	17		

Again, set the significance level at 0.05 and since F0_statistics = 4.32388 > F_percentile = 2.96471, it is then seen that the parameter WD_N differs for these five test structures. This indicates the variation in the manufacturing process is significant and its underlying causes should be evaluated.

Device Level Simulation Challenges for DSM Mixed Signal Design

Pallab Chatterjee - President, P&D Engineering Consultants, Inc.

As a mixed signal circuit designer, the availability and applicability of tools for use in Deep Submicron circuit design, leaves a lot to be desired. There are several challenges facing the designer in arena. A few of these are: Performance aspects of SPICE type simulators, ability to get accurate and application region representative “analog” models from “digital” wafer foundry suppliers, tool set commonality across multiple platforms and multiple clients and data set reuse on previously developed intellectual property (IP).

A solution to some of these issues can be found in the SmartSpice program, as opposed to HSPICE^{tm}, PSPICE-^{tm}, and Spectre^{tm}. I will recount a recent design experience that indicated how SmartSpice solved a number of problems that would not be resolved with the other tools.

The design targets for the block in question is a fairly straight forward Unity Gain stable 400MHz+ CMOS only amp, with a common mode range of 1.5v to 3.5V, power supply of 5.0v, and sub 10ns settling time. The process is a standard single poly 3 metal 0.5μm N-well flow, with diffusion caps and poly resistors. The devices have a nominal Vt of ~0.8v. The design environment was initially HSPICE^{tm} based for device model libraries (level 28 & 49) and use a design library based on PSPICE^{tm} design libraries.

The design flow began by converting the baseline PSPICE^{tm} library data to HSPICE^{tm} library format. This conversion included the reassimilation of the legacy data libraries in HSPICE^{tm} to verify that the design conversion was completed correctly. (Note: This effort was not accounted for in the original schedule). After the base libraries were converted, the new design process could start.

Initially, the design was started by looking for a DC operating bias condition that also met the high level transfer function (.TF statement) criteria for the application. The sub-micron Level28 models had significant difficulty finding valid solutions on what were scaled version of known good designs. Additionally, the temperature and process spread effects seemed to not behave properly as “traditional” engineering variations. This was the first indication of a modeling and tool problem.

We then progressed the design process to the AC analysis stage, and found inconsistencies in the reported gains and impedances of devices between the AC and DC models - even if the SAVEBIAS and NODESET options were enabled to give the same starting conditions. The DC figures made sense based on previous designs, but the AC results were not correct - they indicated 5%+ deviations in gain figures. These results were greater than 10% deviation for open loop conditions. The phase plots indicated that the design was marginally stable at 40-45^{o} of phase margin in typical models and as low as 25^{o} at the process corners.

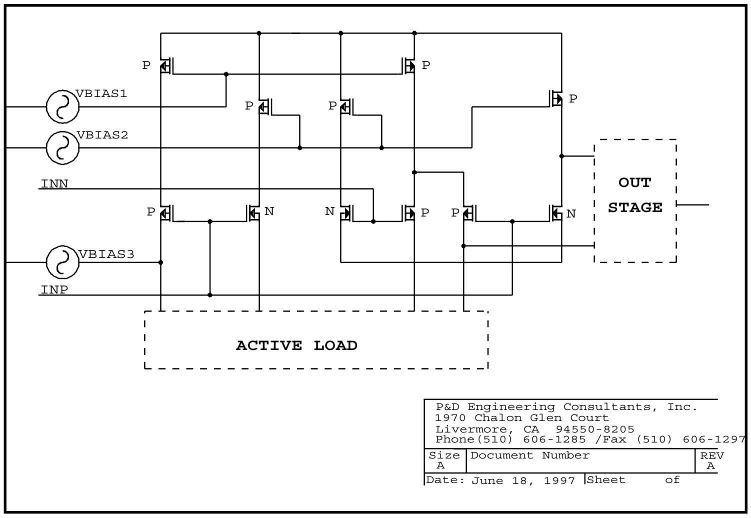


Figure 1. Simplified schematic of amplifier core.

The transient analysis (closed loop only) indicated a stable operating condition with a standard critically damped response (sub 10ns settling time) for a nominal 27MHz 1Volt step function input. This result, although expected based on paper design analysis, was unexpected based on the results of the AC analysis. If the AC results were correct, the design should have exhibited significant damped ringing or gone into uncontrolled oscillations due to the limited phase margin and the high bandwidth input signal.

Neither of these anticipated outputs was found. The circuits had moderate difficulty finding an initial solution for either the fast or slow process corners.

A simplified schematic of the amplifier core design is shown in Figure 1. The results for the closed loop amplifier configuration (vdb, vp and vt) is also shown.

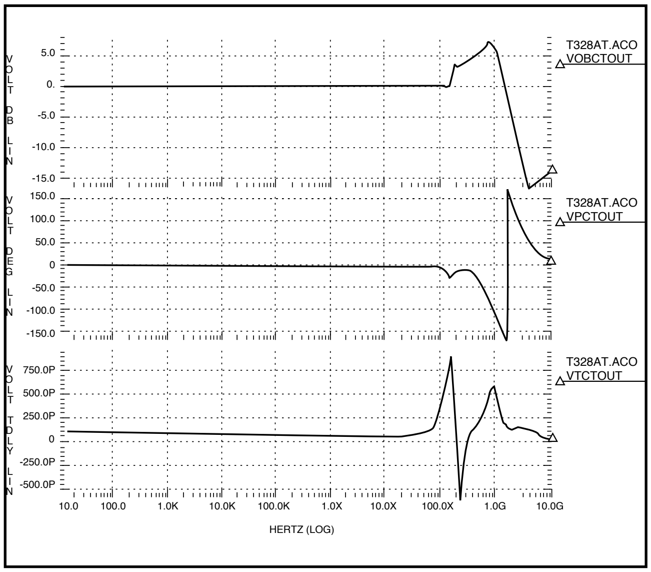


Figure 2. HSPICE^{TM} AC analysis results.

These curves clearly show that the device has a marginal unity stability based on the peaking of the gain curve (~8db) which starts at ~100MHz and goes until ~1.2GHz. This is not consistent with the transient analysis which shows well behaved responses until ~400MHz.

The HSPICE^{tm} results were fairly inconclusive and did not indicate a high degree of confidence on a design that was just a process scaling of a known good cell. After informing the wafer fab that we believed there was a modeling problem, they informed us that BSIM3V3 models were available, for Spectre^{TM} only, which would give us better results. However, as a small design group, we did not want to support yet another simulator package that has its own file/library format and special model requirements.

After we received and reviewed the new models, it indicated that the BSIM3V3 models were created and curve fit using UTMOST. Since the models were optimized by UTMOST, it implicit indicates that the models were optimized and will run on SmartSpice.

The use of SmartSpice fits well into our current methodology that is heavily based on design reuse. For due diligence on the new models, the existing PSPICE^{TM} libraries were run on SmartSpice to verify the design targets and simulator convergence. The PSPICE tm libraries and circuit files ran with no problems and *no modifications* required. The only modification that was required was the change of the Spectre^{tm} model LEVEL from 11 to 49.

After the results for the original PSPICE^{tm} models were obtained, we attached the current circuit design which had HSPICE^{tm} control files. The HSPICE^{tm} circuit files ran on the SmartSpice software with out any modification (other than updating paths). The results of the SmartSpice runs were interesting. They identified a gain consistency (to the 5th decimal place) on the DC and AC analysis. Further, they indicated a UGBW of greater than 2GHz, which was anticipated in the paper/hand calculation design and is supported by the transient response.

The transient response showed settling is sub 8ns in a very stable amplifier. There was no difficulty in finding the initial transient solution at typical, fast, or slow models. The devices also behaved properly with temperature skews. Figure 3 shows the unity gain configuration AC analysis which exhibits a sub 3db peaking and a larger phase margin.

The results from SmartSpice are accurate, fast, consistent (between DC, AC, and Transient analysis) and easy to review with the flexibility of the graphic post-processor. The ease of use on the product is excellent based on the fact that there is no code translation or learning curve required before utilizing existing design data. For DSM and general mixed signal design it is the preferred device level simulation tool as it allows for maximal use of historical/reference designs as well as schematic capture/netlist to simulation flows for new designs.

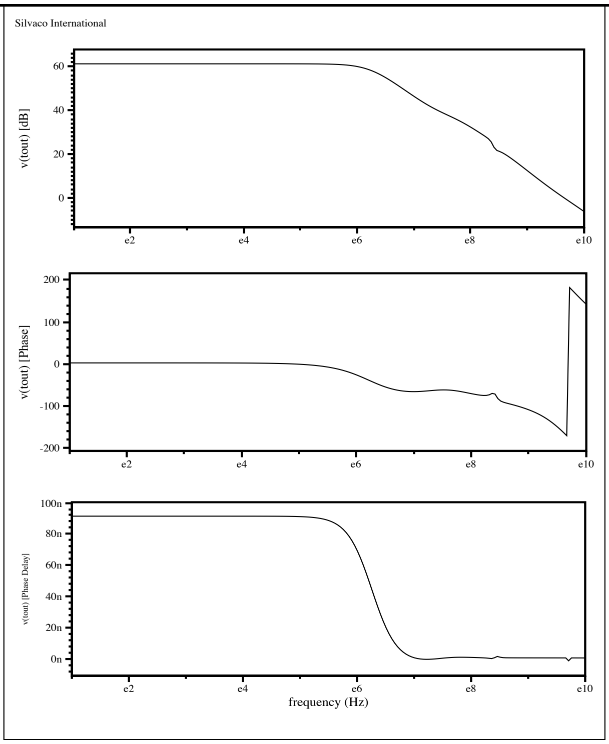


Figure 3. SmartSpice AC analysis results.

For questions regarding this article contact P. Chatterjee (cpdc@earthlink.net)

Farewell to the ‘Simulation Standard’ ...... and Welcome to the “TCAD Driven CAD” Family of Technical Journals

Silvaco is bidding farewell to the publication of our ‘Simulation Standard’ monthly technical journal. In five years of publication from June 1992 through June 1997, interest in this publication has been outstanding. Subscription in this period has grown significantly from 6000 to over 15,000.

The ‘Simulation Standard’ has come to be recognized worldwide as a premier source of the latest information on advances in semiconductor technology modeling and simulation. With this publication, Silvaco has succeeded in its mission of providing continued updates on the latest technology advances to the semiconductor community worldwide. With the vast majority of all subscriptions now being delivered through direct mail, readers in all corners of the world now concurrently receive the latest technical information available. Silvaco would like to again express our appreciation to the many guest authors from industry and academia whose outstanding contributions have helped build this publication to the industry leading stature that it now enjoys.

As Silvaco continues to evolve as a company, the time has now come to expand the focus of our monthly technical publication. A new family of journals will be launched in July 1997 under the banner of ‘TCAD Driven CAD’. The new publication will consist of three distinct issues, each published once quarterly. The new publication schedule is as follows:

“Journal for Process and Device Engineers”
February, May, August & November

“Journal for Circuit Simulation and
Spice Modeling Engineers””
January, April, July, October

“Journal For IC CAD/CAE Engineers””
March, June, September & December

This series of journals will provide both the leading edge technology coverage our readers have come to expect, and expanded coverage on the rapidly emerging integration of TCAD and traditional CAD in all phases of IC design. Readers can continue to look to Silvaco for unique insight into the technology issues that drive the development and manufacturing of next generation IC designs.

Calendar of Events

May

1
2
3
4
5 CICC-Exhibition Booth #12
6 CICC-Exhibition Booth #12
7 CICC-Exhibition Booth #12
8 CICC-Exhibition Booth #12
9
10
11
12
13
14
15 Workshop - Munich
16
17
18
19
20 Workshop - Guildford
21
22
23
24
25
26 ISPSD - Weimar, Germany
27 ISPSD - Weimar, Germany
28 Workshop - Grenoble
29
30 Workshop - Japan
31

June

1
2
3
4 Blaze Workshop - Italy
5
6
7
8
9 DAC 97 - Anaheim, CA
10 DAC 97 - Anaheim, CA
11 DAC 97 - Anaheim, CA
Workshop - Scotland
12 Workshop - Munich
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27 Workshop - Japan
28
29
30

Bulletin Board

New Instrument Drivers in UTMOST

In response to customer requests, several new instrument drivers have been developed for UTMOST III. These include:

- Noise Meters: Re-written driver for HP3562A Spectrum Analyzer to provide better measurement speed and synchronize work with Silvaco's S3245A Noise Amplifier
- Switching Matrix: A driver has been added for the new HP3235 switching matrix
- Probers: Two new semiautomatic prober drivers have been added including the Ultracision 880e manufactured in the USA, and the Micronics SP900B from Japan
- Thermal Chuck: A new driver for the Micronics WEC-10 (Japan) allows hot/cold control from -55deg C to +200 deg C

SmartSpice Workshops in Japan

Two workshops on SmartSpice are being held in Japan in July. The workshop titled *"Introducing SmartSpice - The Leader in Analog Circuit Simulation"* will be presented in Tokyo on July 9, and in Osaka on July 10. The presentation by Dr. Alex Zavorine will focus on the latest developments in SmartSpice that are rapidly establishing it as the leading analog simulator worldwide: parallelization, convergence, speed, accuracy and advanced device models.

SmartSpice Integration Improved

The level of SmartSpice integration within major CAD vendor frameworks has been made substantially tighter. This allows SmartSpice to be run directly from within the host environment. Improvements in the SmartSpice output routines allow the simulation results to be directly loaded into each framework's waveform display tools. This greatly enhances the cross-probing and back-annotation features of the integration. The previous integration technology will continue to be supported for older versions of the individual frameworks. A more detailed description of the Cadence and VIEWlogic integrations in given in this issue.

*For more information on any of our workshops, please check our web site at **http://www.silvaco.com***

The Simulation Standard, circulation 15,000, Vol. 8, No. 5, May 1997 is copyrighted by Silvaco International. If you, or someone you know wants a subscription to this free publication, please call (408) 567-1000 (USA), (44) (1483) 401-800 (UK), (81)(45) 820-3000(Japan), or your nearest Silvaco distributor.

Hints, Tips and Solutions

Mustafa Taner, Applications and Support Engineer

Q. How can I measure and extract Area (CJ) and Sidewall (CJSW)capacitances using UTMOST?

A. The CJ/CJSW routine(Routine#72) in UTMOST can be used to extract area and sidewall capacitances. CJ and CJSW parameter extraction requires measuring two different diodes. One of the diodes should have relatively large area compare to its periphery (Area structure; Typical size: W=200 um L=200um) and the second structure should have long periphery which is comparable to its area (Periphery or Finger structure; Typical size: W= minimum design rule width L= 200 um and 50 fingers).

The extraction of CJ and CJSW requires that both diodes should be measured and the two unknowns (CJ and CJSW) should be solved from the two equations:

```
C1 = CJ * A1 + CJSW * P1
C2 = CJ * A2 + CJSW * P2
```

where; C1: Measured capacitance value for the area diode.
C2: Measured capacitance value for the periphery diode.

The CJ/CJSW routine utilizes the above equations and automatically extracts and updates parameters CJ, PB, MJ, CJSW, PBSW and MJSW in the parameter screen.

The Kelvin measurement method should be used for capacitance measurements.The LCR meters have High Voltage, High Current, Low Voltage and Low Current terminals. The cables which are coming from the high Voltage and Current terminals should be connected together using a "T" connector and the T connector should be connected to the "High" probe. The Low Current and Low Voltage terminals should be connected to the "Low " probe.

The parasitic capacitances associated with the metal lines and pads should be compensated prior to the actual measurement. The Capacitance Calibration in the "Hardware Configuration" screen should be executed for proper parasitic capacitance compensation. To calibrate, the High-Terminal probe should contact the pad which is connected to the bottom junction and the Low-Terminal probe should stay just above the pad (no contact!) which is connected to the top junction. After the hardware connection is completed press the "Calibrate" button in the "Capacitance Calibration" screen to start the calibration.

After the calibration is completed follow these steps to complete the CJ/CJSW measurement:

1. Calculate the Area and Periphery of both structures.
2. In the main UTMOST screen Press the "Hardware" followed by the "Probing" buttons to open the "Hardware Probing" screen and select the "Devices". This will open the "Device Pads" screen

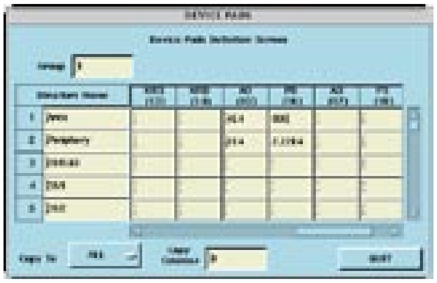


Figure1. Device Pads Screen.

3. Enter the Area and Periphery values for the Device#1 (area) and Device#2 (periphery). (Figure 1.).
4. Open the "Strategy" screen delete the "Width" and "Length" values for Device#1 and Device#2.
5. Open the Measurement Setup screen for the CJ/CJSW routine and enter the measurement variables "start_bias" and "stop_bias". (Typical values:start=-0.1, stop=3 or 5). Toggle the "Auto Polarity" button to "Enable".
6. Contact both pads of Area diode using the high and low probes.
7. Open the "Extraction" screen, select the "CJ/CJSW" routine and press the "Measure" button. UTMOST will prompt a message: "Connect AREA diode Press ENTER to continue"
8. After the measurement is completed UTMOST will prompt the second message:"Connect SIDEWALL diode Press ENTER to continue"
9. Move the probes to the periphery structure and press ENTER to complete the measurement.
10. After the periphery structure is measured UTMOST will display both junction capacitance curves on the "Graphics" screen.

In order to extract the initial values for CJ, PB, MJ, CJSW, PBSW and MJSW select the "Fit" option from the Graphics screen. The extracted values will be copied into the parameters screen and the simulation with these initial parameters will be overlayed with the measured data.

The Junction capacitance parameters can be selected for global optimization and by using the internal simulator all six parameters can be optimized together.

Call for Questions

If you have hints, tips, solutions or questions to contribute, please contact our Applications and Support Department
Phone: (408) 567-1000 Fax: (408) 496-6080
e-mail: support@silvaco.com

Hints, Tips and Solutions Archive

Check our our Web Page to see more details of this example plus an archive of previous Hints, Tips, and Solutions
http:://www.silvaco.com

Revenues, Revenues, Revenues.......

Many of our customers may have recently been presented with so-called "credible" information that puts Silvaco's 1996 revenues at $2.3 million. It has been brought to our attention that one of our competitors, *with apparently nothing better to offer customers*, has been widely quoting this erroneous Dataquest industry revenue brief.

Silvaco would like to point out a few pertinent facts:

- Silvaco currently maintains 10 wholly owned sales and support offices in locations throughout North America, Asia and Europe
- Staffing at all Silvaco International companies now totals approximately 140 full time employees
- As previously reported, 1997 first quarter revenues from Silvaco's Japan subsidiary *alone* totaled $2.06 million
- Silvaco's Japan office does not now or never has represented the major source of revenues for Silvaco International

Silvaco currently remains a privately held company and bears no responsibility to report revenues to industry watchers such as Dataquest. Looking at the publicly available information on the structure of Silvaco International today, it is clear that the numbers presented in no way mesh with the reality of managing this rapidly growing software company. To quote these numbers as a way of gaining competitive advantage and disguising the shortcomings of their product offerings is pitiful at best, and at worst the last desperate grasp at survival by a fading competitor.

SILVACO
International

4701 Patrick Henry Drice, Building 2
Santa Clara, CA 95054

Telephone: (408) 567-1000
Fax: (408) 496-6080
URL: http://www.silvaco.com