A Technology-Agnostic Simulation Environment (TASE) for Iterative Custom IC Design across Processes

Satyanand Nalam, Mudit Bhargava*, Kyle Ringgenberg, Ken Mai*, and Benton H. Calhoun
*University of Virgnia, *Carnegie Mellon University*
*{svn2u,kringg,bcalhoun}@virginia.edu *{mbhargav,kenmai}@andrew.cmu.edu*

Abstract - **A designer's intent and knowledge about the critical issues and trade-offs underlying a custom circuit design are implicit in the simulations she sets up for design creation and verification. However, this knowledge is tightly conjoined with technology-specific features and decoupled from the final schematic in traditional design flows. As a result, this knowledge is easily lost when the technology specifics change. This paper presents a Technology Agnostic Simulation Environment (TASE), which is a tool that uses simulation templates to capture the designer's knowledge and separate it from the technology-specific components of a simulation. TASE also allows the designer to form groups of related simulations and port them as a unit to a new technology. This allows an actual design schematic to remain tied to the analyses that illuminate the underlying trade-offs and design issues, unlike the case where schematics are ported alone. Giving the designer immediate access to the trade-offs, which are likely to change in new technologies, accelerates the re-design that often must accompany porting of complicated custom circuits. We demonstrate the usefulness of TASE by investigating Read and Write noise margins for a 6T SRAM in predictive technologies down to 16 nm.**

I. INTRODUCTION

The rapid creation of new technologies according to Moore's Law has made cross-technology porting an important part of circuit design. An ideal method for porting custom circuits would reuse an existing design in a new technology while also re-optimizing or re-structuring the design to best take advantage of that process's features. For complicated blocks that defy complete automation, the best way to do this would expose the tradeoffs and knowledge that underlie the original design in full view of the designer in the new technology. Existing methods for porting offer imperfect solutions. Classic ASIC flows approach porting by largely decoupling the design (e.g. HDL) from a specific technology. The foundry typically provides the process dependent part of the design flow (standard cell library), allowing for relatively easy porting. Custom circuits, on the other hand, require more careful designer intervention for process porting. A large number of tools exist for direct porting of custom layout (e.g. [1]), but these do not carry any useful design information to help a custom designer. To access design information, designers can use built-in SPICE features (.alter, .param, .lib) that allow single netlists to support multiple processes, but these features become unwieldy for complex designs or when reusing netlists for different simulations. Many designers employ their own scripts for porting designs and simulations, but these tend to be ad hoc, proprietary, or generally unavailable to the community.

A large variety of optimization tools for analog circuits have been proposed, and some of these aid with process porting (e.g. [2]). The tool in [3] ports analog schematics and layout by identifying known sub-blocks (e.g. current mirror) and resizing those using SPICE simulation to meet designer supplied constraints. This approach represents a more intelligent method for porting that preserves some designer intent by maintaining sub-block characteristics, symmetry, etc. A similar tool captures a designer's knowledge and intent by embedding comments that specify simulation analyses and constraints in the schematic view for analog circuits [4]. This approach requires a fairly complex "language" of directives to specify the full range of possible simulations within the comments, but it offers the valuable benefit of carrying some underlying information that leads to the design (in the form of simulations) along with the design itself. This is in contrast to conventional porting, which obscures and discards the original knowledge and tradeoffs that led to the initial design.

In this paper, we present a Technology Agnostic Simulation Environment (TASE) that expands these ideas and generalizes them to apply to any custom circuits. For example, many digital designs like SRAM, dynamic logic, high speed arithmetic blocks, global interconnect, and off-chip I/O require custom design that would benefit from a better methodology for porting or for exploring new technologies. TASE provides an interface to existing SPICE simulation tools that abstracts away the process dependence of simulation setups with minimal impact on the existing tool flow. Further, TASE allows a designer to associate multiple simulation files, enabling an explicit coupling between a final design and important analysis of its underlying tradeoffs. TASE makes the following key **contributions**:

- Automates *groups* of related simulations across technologies
- Preserves and exposes designer intent and underlying tradeoffs, allowing rapid redesign and adjustment to unforeseen issues.
- Increases productivity of an iterative design process.
- Expands and "learns" with use as it captures increased levels of the designer knowledge and intent.

These features of TASE are even more important now, considering the changing landscape of CMOS technology, which creates three new problems that amplify the need for a technology independent simulation framework. First, process nodes below 65nm are an increasingly challenging design environment due to variability, multiple leakage mechanisms, increasing power, and other scaling-related issues. These new problems change the trade-offs behind a design, making old designs non-optimal or even non-functional. However, the insights and knowledge gained from designing the circuit in the previous technology are not readily bundled with the design itself. The second problem is the widening variety of manufacturing processes (CMOS bulk, Silicon on Insulator (SOI), High-K-Metal-Gate, etc.) and process features (low-power, high-performance, multi-V_T, etc.). To compare these options requires repetitive simulations using multiple model files. The choice of process is especially important to the many fabless IC companies. Finally, the complexity of the devices leads to frequent model changes, and the initial models can differ significantly from those of a mature process. Thus, designers must either over-margin their designs to guard against late, significant model changes or revise their designs at each model revision.

978-1-4244-5028-2/09/$25.00 ©2009 IEEE

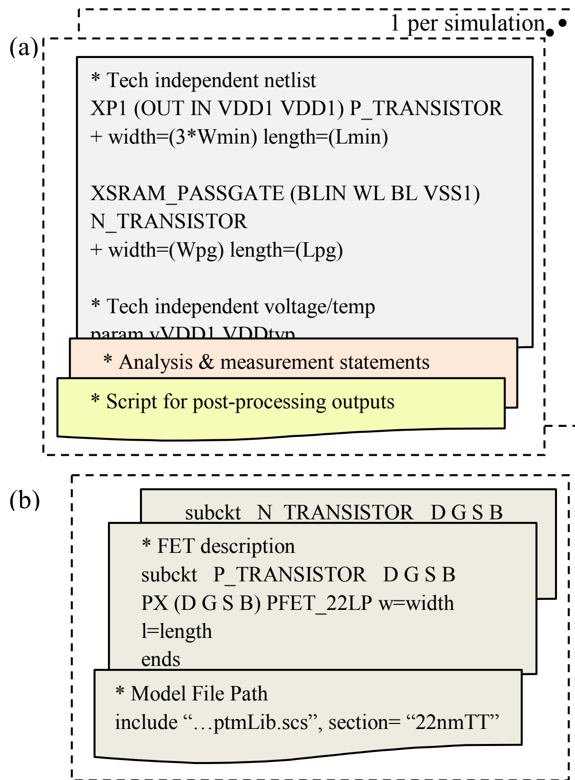

Fig 1: Example TASE (a) technology agnostic template and (b) technology-specific configuration.

Since TASE automates simulation across multiple processes or revisions of model files within a technology, it improves the speed and efficiency of design porting when underlying tradeoffs change. TASE also facilitates across-technology trends and forecasting and supports circuit / process co-optimization. The remainder of the paper details the TASE tool and provides example uses of the tool. Section 2 gives a detailed overview of the TASE tool. Section 3 shows how TASE can automatically characterize a new technology and forecast device properties for future technologies. In Section 4, we use TASE to aid in the sizing, characterization, and optimization of the six transistor (6T) SRAM cell across a number of future process technologies. Section V concludes the paper.

II. OVERVIEW OF TASE

TASE acts as a wrapper, currently implemented in MATLAB and PERL, to an existing SPICE simulation framework (e.g. Cadence). TASE is built around a low-level circuit simulator, and is best suited for quick characterization of relatively small circuit blocks composed of a few tens of transistors. In order for TASE to perform technology agnostic simulation, it must cleanly separate the technology-dependent aspects of a simulation from the technology-independent aspects. It is these technology-independent aspects that can propagate designer knowledge and intent. This is the key insight that enables TASE to reuse simulations for any technology or process corner.

The input to circuit simulators such as SPICE or Spectre has the following basic components, all of which contain technology specific information.

- Netlist – circuit topology description
- Stimuli and initial conditions
- Analysis statements (e.g. Transient, DC)
- Measurement statements (e.g. Delay, Power)
- Temperature and voltage of operation
- Device models

A. *TASE Structure*

We separate the components of a TASE simulation into a suite of technology independent templates and a single technology-specific configuration per process. A template exists for each simulation, and each **technology agnostic template** contains several components (Fig 1 (a)):

- Technology agnostic **netlist**
- Technology agnostic **stimuli and initial conditions**
- Technology agnostic **analysis and measure** statements
- Technology agnostic **post-processing** script

First, the netlist uses technology-independent references to sub-circuits that represent FETs ("P_TRANSISTOR" and "N_TRANSISTOR"). Device sizes, etc., are not specified using absolute numbers, but in terms of minimum feature size or as parameters to be defined by the user. Second, analysis and measurement statements set up the simulation and capture the relevant results. Many features of these statements are already process independent, but values that must change with process (e.g. transient run time) are parameterized. Finally, optional post-processing scripts (e.g. Matlab) extract, process, and plot data from the simulation results. The **technology-specific configuration** (Fig 1 (b)) contains all of the information that changes with each of the different processes:

- **Device model files**
- **Temperature and voltage of operation**
- **Simulation parameters** (e.g. Monte Carlo seed)
- **Template parameters** (e.g. device sizes)

Technology specific sub-circuits for NMOS and PMOS devices, with links to the device models, provide the generic P_TRANSISTOR and N_TRANSISTOR components used in the simulation templates.

The structure of TASE makes it quite easy to add new content. To add a new simulation, the designer must simply build a new technology independent template and place it in the templates directory, then add definitions for any new parameters to the top level execution file. Building a template is only slightly more time consuming than writing a direct simulation, so the overhead of this procedure is justified since the template can be re-run in any technology without modification. Adding a new technology requires only writing a new technology-specific configuration, which contains only the transistor sub-circuits and model path information. Now, these analyses and insights into the tradeoffs that lead to certain design decisions will port with the final design into new processes.

B. *TASE Operation*

To run TASE, the user sets up a top level **execution file** that can associate multiple templates into a group of related simulations. Through the execution file, the designer selects a technology-specific configuration (or multiple configurations for cross-process simulations) and a *group* of templates to run, which can be selected

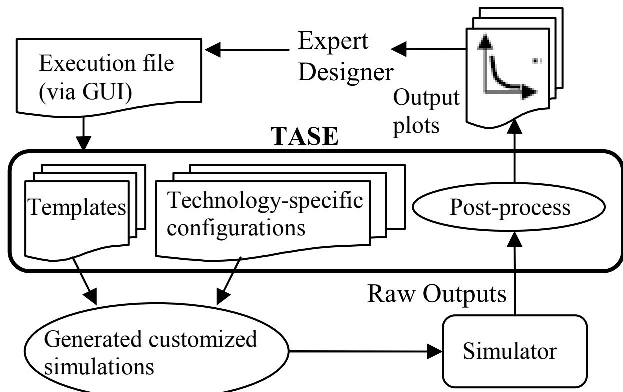

Fig 2: Flow Diagram for TASE.

individually or for a particular category (e.g. process characterization, SRAM, etc.). Fig 2 depicts the TASE tool flow. TASE combines the parameter values from the execution file and the specified model-file with the simulation templates, which converts the technology agnostic templates into simulations customized for the chosen technology. Next, TASE launches the circuit-level simulator(s) to run the selected simulations. After they conclude, TASE consolidates the post-processing scripts (Matlab) from the templates, launches the consolidated script to process the raw output data from the multiple simulations, and produces plots and results for the designer. Since templates can be reused in multiple groups, the post-processing for a given template can be specifically tailored to the context of the simulation in each group. This also supports using TASE as an iterative design tool, since simulation templates can be reused and refined in the context of a new execution file.

A custom graphical user interface (GUI), shown in Fig 3, allows a designer to rapidly generate and manage large sets of execution files. Upon launch, the GUI generates fields for each parameter, both required and optional, as well as a list of all available test simulations. Execution files can be saved and re-loaded to manage different designs grouped with their related tradeoff analysis simulations. An *Extrapolate* utility allows the designer to specify a range of values for up to two parameters. The tool then generates a set of execution files that covers each permutation on the specified parameters, allowing multi-dimensional simulation sweeps. The GUI also manages output plots, allowing their convenient display in Matlab. For example, slider bars allow the user to drag through the range of swept parameters and to immediately see the resulting permutation of the simulation results plotted in the GUI.

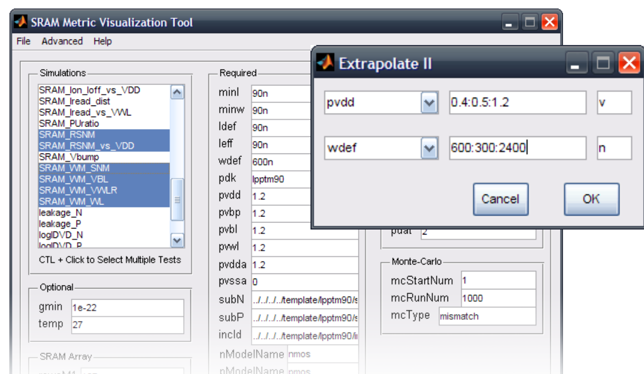

Fig 3: Screenshot of TASE GUI for managing execution files. Extrapolation tool option displayed.

III. Process Exploration and Forecasting

In this section, we demonstrate the technology agnostic nature of TASE using a simple example of device characterization and showing how TASE facilitates forecasting. Before starting a design in an unfamiliar technology, designers run a suite of simulations (e.g. fan out of four delay, beta ratio, and saturation current) to learn the basic characteristics of the technology. For example, a designer first encountering a 16nm technology may want to learn the saturation current (I_{ON}) and leakage current (I_{OFF}) characteristics with varying drain-source voltage (V_{DS}), gate-source voltage (V_{GS}), and device dimensions (W and L). To perform this analysis in TASE, the designer picks existing templates to evaluate these metrics by simply selecting them in the top level execution file and referencing the new model file.

Suppose the designer is concerned about gate leakage in the new technology, but discovers that no template exists to plot gate current. The designer can add a new template to TASE that evaluates gate leakage. This increases the utility of the tool, as the designer can now reuse the new template in the future for, say, re-evaluating the characteristics when process models are revised. TASE essentially grows and "learns" with use as more and more designer knowledge and intent integrates into the tool as new simulation templates.

Suppose the 16nm technology contains a standard V_T device and a low V_T device. The designer runs the selected templates with TASE using the technology-specific configuration for the standard V_T option. (We use 16nm PTM high-performance and low-power models for this demonstration.) He observes that the standard V_T NMOS I_{ON} is insufficient for his design, so he now must evaluate the same device characteristics for the new low V_T device. TASE allows him to rerun the same templates as before by simply selecting a different technology in the execution file. His knowledge about gate leakage which was captured earlier by TASE is also used to re-evaluate gate leakage for the low V_T device. He now observes that the I_{ON} meets the requirements of his design and decides to use the low V_T device for his design.

This simple example shows how TASE supports rapid simulation across different technologies, process corners, and model revisions without any changes to the low level simulation setup. This also facilitates porting to new technologies, since TASE can rapidly re-evaluate critical circuits and important metrics. Designers can easily archive large sets of simulations (e.g. multiple templates) for immediate use in a given technology by customizing top level execution files. With only a change to the technology reference in the execution file, TASE will re-run the entire suite of simulations specified by that file, in a new technology. We regularly use a standard execution file that produces a lengthy report plotting key device features for different operating regions in a new process.

This capability makes TASE useful for forecasting the feasibility of a design or the emergence of new problems across new technologies (e.g. with predictive technology models (PTMs) [5]). For instance, the ratio of I_{ON} to I_{OFF} is important to many custom circuit designers (e.g. for memory, dynamic logic, keepers, etc.). We use bulk CMOS and metal-gate/high-K PTMs to forecast I_{ON}/I_{OFF} down to 16nm in TASE. A *single* TASE execution that selects multiple processes and two templates produces the results in Fig 4 for the I_{ON}/I_{OFF} ratio and in Fig 5 for the NFET I_{OFF}. For bulk CMOS, we observe that the ratio degrades by almost two orders of magnitude. This is mainly due to exponentially increasing leakage.

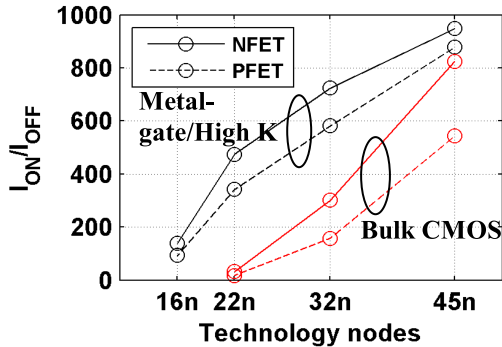

Fig 4: I_{ON}/I_{OFF} at 27°C for 45nm-16nm PTMs

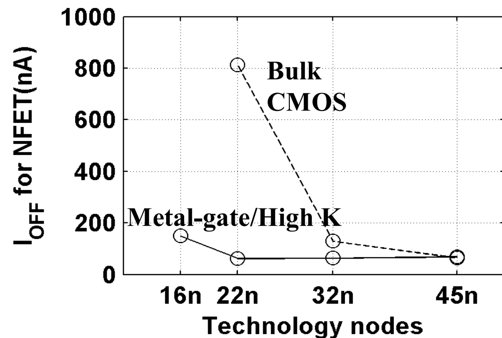

Fig 5: I_{OFF} at 27°C for 45nm-16nm PTMs

By using Metal-gate/high K techniques, the I_{ON} increases and the leakage goes down drastically when compared to bulk CMOS. This technology proves to be a good alternative to bulk CMOS for 22 nm. However, even this technology fails to deliver a substantial I_{ON}/I_{OFF} ratio at 16 nm. This is one of the many challenges facing designers in the future, particularly those working in the dynamic logic or SRAM domains. It would be quite easy for a designer to use TASE to quickly evaluate I_{ON}/I_{OFF} for other sub-45nm technology options. Finally, while device characterization is a simple example, the templates for specific device traits may have substantial bearing on custom circuit designs. For example, the dependence of leakage on device width and length can influence device sizing in an SRAM bitcell. TASE allows the I_{OFF} vs. W and L templates to be grouped with simulations of a bitcell using a distinct execution file, so that a designer can see how this trade-off changes in a new technology to re-evaluate its impact on the final design.

IV. USING TASE FOR 6T SRAM CELL DESIGN

Optimal design for the 6T cell requires understanding numerous tradeoffs that change with technology scaling, making it ideal for illustrating how TASE allows groups of simulations to provide trade-off information along with a final design. In this study, we use TASE with PTMs [5] to do a simplified feasibility analysis for 6T SRAM by investigating the scaling of read and write margins from 45nm down to 16nm. We study several read-assist and write-assist techniques and compare their impact on the cell's performance under worsening parametric fluctuations.

In this section, we show how TASE could assist a designer in sizing the devices in a 6T SRAM cell, checking the cell read and write margins, and making design decisions about necessary cell I/O circuit changes. TASE can dramatically reduce the time a designer must invest and can simplify the performance of these tasks. In

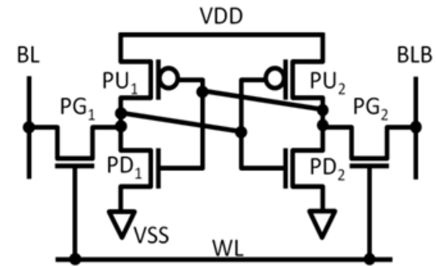

Fig 6: Conventional 6T SRAM cell

modern processes, growing device variations reduce the stability of conventional 6T SRAM bitcells () and limit the allowable scaling. The reduction of 6T cell stability has prompted designers to use alternative bitcells. For example, Intel's Nehalem processor uses an 8T-based cache [6]. There are many important cell stability metrics, but two of the most crucial are the Read Static Noise Margin (RSNM) and Write Noise Margin (WM) [7]. A low RSNM can result in the unintentional flipping of the stored value on a read, and a low WM can result in a cell being unwritable.

Inter- and intra-die variations coupled with the near minimum sized devices in the cell result in wide statistical distributions of RSNM and WM. Since modern chips can have large SRAM arrays, reaching up into the tens and hundreds of megabytes, the likelihood of a chip containing a bad cell due to RSNM or WM failure is significant.

To prepare TASE for evaluating the RSNM and WM, a designer must set up templates for each single simulation of interest. In this example, the initial two simulations of interest are Monte Carlo runs of WM and RSNM with local variation. The size (e.g. number of iterations) of each Monte Carlo run is a parameter that can be set in the execution file. It can thus vary for the same template as different execution files leverage that template for different purposes. Now we work through example uses of TASE that mimic how a designer might address the problem of designing 6T SRAM across scaled technologies.

A. Trend Forecasting

To understand how process scaling will affect RSNM and WM, we first run simulations using TASE to study the trends of SNM and WM across PTM files from 45nm down to 16nm. We derive the cell device sizes by extrapolating from an industrial 45nm sub-DRC cell. TASE makes this cross-technology analysis very simple. The designer needs to write a new process specific configuration for each PTM technology, which takes only a few minutes. Then the entire comparison can be launched with a single call to TASE that selects the WM and RSNM templates and the PTM processes. Fig 7 shows the output of this run. Again, the designer only sets up templates to examine RSNM and WM once, but TASE uses those to produce data across six technologies.

The data shows that both RSNM and WM decrease with each process shrink, so cells are becoming increasingly less reliable. At 16nm, the worst case values from a 1000-point Monte Carlo simulation for both RSNM and WM are negative, indicating unacceptable probability of read upsets (negative RSNM) and inability to write (negative WM). This clearly suggests that a conventional 6T cell will not be robust in future technologies and that we need to alter the cell sizing or implement new design innovations, like read and write assist circuits. This type of

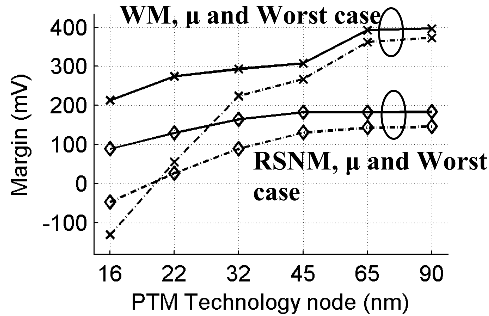

Fig 7: Read SNM & Write Margin across different technology nodes measured at TT 27°C and nominal voltages. The mean and worst cases are over 1000-point MC simulations.

forecasting reveals tradeoffs that might be important for a designer to relate to the final design. For example, the design at the 45nm node might use an assist method for RSNM only, since WM seems sufficiently high (Fig 7). However, at the 22nm node, WM becomes equally poor with RSNM, and it could even be worse farther out the distribution tails. A simple port of the design would obscure this tradeoff, but TASE allows the designer to group the WM and RSNM simulations with the top level simulations of the final array. This makes the key tradeoff easy to identify (e.g. by its grouping with the final design) and to re-evaluate (e.g. by re-running that execution file in TASE) in the new technology.

B. Device Sizing

The noise margins of a bitcell can be improved by modifying the device sizes. To understand the effectiveness of changes in device size in improving RSNM and WM, we use TASE to sweep the cell sizing space for the optimum device sizing. Again, this entails setting up one template that executes the sizing sweep. Using that template, a single execution of TASE produces the data across multiple processes (Fig 8). The plot shows how RSNM and WM vary with the width of PU, PD, and PG transistors. The widths are modified from 0.8 to 1.2 times the default sizing, which is extrapolated from a 45nm sub-DRC commercial cell. RSNM improves with the ratio of the sizing of PD to PG. WM increases when the ratio of strengths of PG and PU is higher. Fig 8 shows that changing device sizes is less effective as we move to finer technology nodes. Moreover, the conflicting design requirements for improving RSNM and WM means that designing the bitcell to improve one margin would worsen the other.

This example shows how TASE can support a semi-automated iterative design process. The designer can modify templates to adjust device sizes or circuit topology, alter design parameters in the execution file, or add new measurement or analysis statements in a template. By observing circuit performances over multiple runs, the designer can optimize the netlist to meet the desired specifications or constraints. This continuous interaction between the designer and TASE allows it to incorporate more and more of the designer's knowledge into the templates. For example, we mentioned in section III how single device simulations (e.g. of I_{OFF} vs. W and L) might influence bitcell design. TASE allows those simulations to be grouped with other bitcell simulations (e.g. Fig 8) so that the designer can sweep widths once and see the impact on multiple metrics of interest to the bitcell (e.g. NM, leakage, I_{read}, etc.).

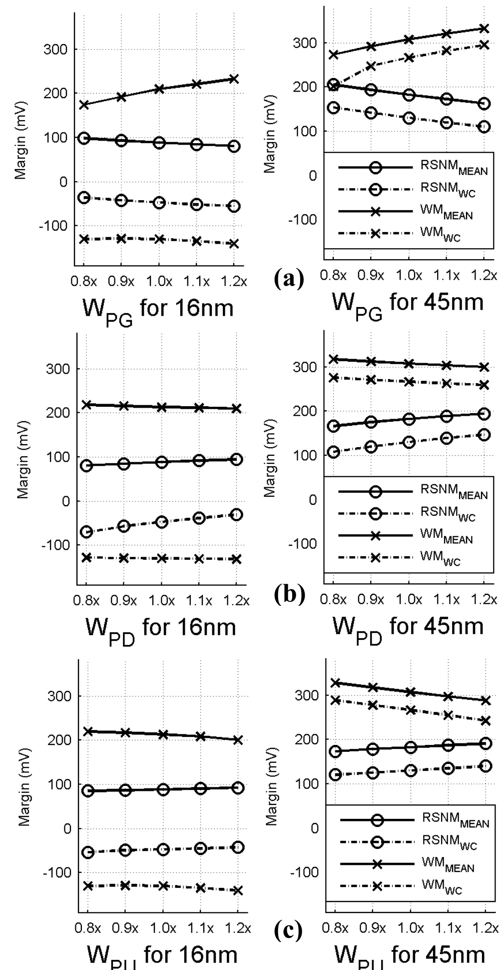

Fig 8: RSNM and WM variations with device sizes for 45nm and 16nm technology at 27°C TT at their nominal voltages. (a) RSNM, WM vs. width of passgate (b) RSNM, WM vs. width of pulldown (c) RSNM, WM vs. width of pullup

C. Corner Simulations

While we can now chose device sizes that maximize robustness at the TT global corner, we must ensure that the design has reliable noise margins at the worst-case process corners as well. TASE also easily provides for simulations at different global process corners either in a single technology or across multiple processes. If we model each global process corners as a process specific configuration, then we can generate this data without making any changes to the simulation templates.

Fig 9 shows the variations of RSNM and WM of the conventional bitcell across the global process corners in 45nm that result from a single TASE run. The RSNM is worst at the fast NMOS, slow PMOS (FS) corner and WM is worst at the SF corner. Both WM and RSNM degrade dangerously close to 0 at some corners. Obviously, a reliable design must meet the noise margin specifications at all process corners, and this information can readily be generated using TASE.

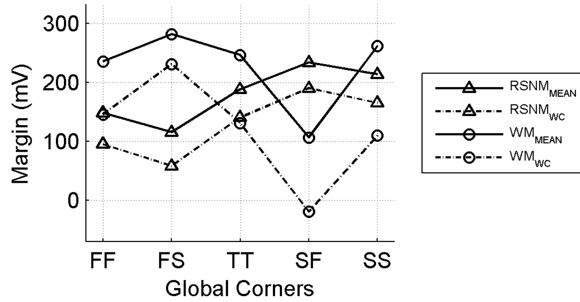

Fig 9: Read SNM & Write Margin across process corners in 45nm PTM @ 1.0V 27°C. The mean and worst case is over a run of 1000 MC simulations

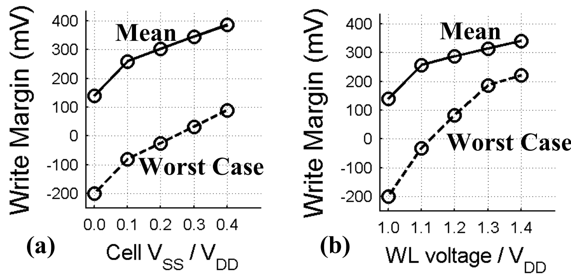

Fig 10: Mean and Worst Case (out of 1000 cells) WM in 16nm @ 0.7V 27°C TT corner with (a) Cell V_{SS} and with (b) WL raised by 10-40% of the VDD

D. Read/write Assist Evaluation

As the corner simulations have shown unacceptably small RSNM and WM for some corners, we may consider implementing some of the proposed circuit techniques for boosting the read and write margins. Several techniques have been proposed that alter voltages in the periphery to improve margins. These include altering the voltages of the cell array (V_{DD}, V_{SS}), wordline (WL), and bitline pre-charging devices to improve both RSNM and WM (e.g. [8]). However, we need to evaluate their efficacy in the target technology in order make the optimal choice for our design. This design problem provides an example of how TASE can aid in circuit design decisions and learn with increasing use.

The two write assist methods that we consider for this simple example are raising the V_{SS} supply to the cell and raising the voltage of WL by a few hundred mV during the write operation. We can add two new templates to TASE, one to test the WM using each method, and then simulate those templates across any technologies. Fig 10 compares the efficacy of the two techniques at a single node, as output from TASE. The plots show that V_{SS} boosting is more effective than WL boosting, but unfortunately neither approach is adequate by itself to provide sufficient worst-case WM.

We next evaluate improving RSNM by either raising the V_{DD} supply to the cell or by pre-charging the bitlines to sub-V_{DD} levels. Again, two new templates allow TASE to generate data across many technologies. Fig 11 shows the improvements in RSNM over conventional biasing using these techniques at 16nm. Again, this specific analysis shows that V_{DD} boosting can improve RSNM more, but that RSNM is still dangerously low. Cell biasing techniques can extend the use of the 6T cell for a few more technology nodes, but more aggressive solutions are necessary for SRAM before 16nm. As

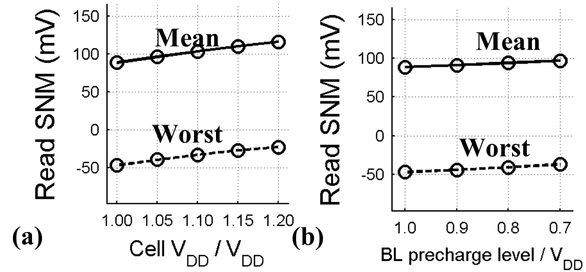

Fig 11: RSNM with dynamic cell biasing techniques in 16nm @ 0.7V 27°C and TT process (a) Supply to the array is swept from 1x to 1.2x of the nominal V_{DD}. WL driver is supplied with nominal V_{DD}. (b) BL is precharged to 1x to 0.7x times the nominal V_{DD}.

the set of potential read and write assist methods grows in TASE's template library, the designer can rapidly assess a wider range of potential designs in a new process. This allows the design to adjust more readily if the underlying tradeoffs change and call for an alternative to the initial design choice. The types of analyses in this example, which TASE can very quickly re-apply to new technologies, can help SRAM designers to forecast limitations to designs at future technologies, to compare potential new circuit solutions, and to optimize a design in a given technology.

V. CONCLUSIONS

In this paper, we presented TASE, an automated simulation environment that enables designers to preserve their knowledge of design issues and trade-offs in customized groups of technology-independent simulation templates. These templates need to be setup only once and can then be reused for any technology with minimal changes to user input. We demonstrated the utility of the tool through two examples. First, we showed how it helps designers acclimate to a new technology with minimal effort, how the designer can in turn help the tool expand its usefulness with time, and how it can help forecast device behavior. Second, we showed how the tool facilitates a rapid and thorough investigation of the SRAM reliability problem and a quick evaluation of possible solutions. This is possible because the designer can reuse existing templates for measuring RSNM and WM that are accompanied by other simulations that define the underlying tradeoffs.

VI. REFERENCES

[1] F. Fang and J. Zhu, "Automatic Process Migration of Datapath Hard IP Libraries," *ASP-DAC*, pp. 887-892, 2004.

[2] P. B. Wu, et al., "AMODA: A Flexible Framework for Automatic Migration of Analog Macro Designs Using Optimization Techniques," *Midwest Symp on Circuits and Systems*, pp. 988-991, 2000.

[3] S. Hammouda, et al., "Chameleon ART: A Non-Optimization Based Analog Design Migration Framework," *DAC*, pp. 885-888, 2006.

[4] Liu, D.; Sidiropoulos, S.; Horowitz, M., "A framework for designing reusable analog circuits," *Computer Aided Design, 2003. ICCAD-2003. International Conference on* , vol., no., pp. 375-380, 9-13 Nov. 2003.

[5] Zhao, W. & Cao, Y. New generation of predictive technology model for sub-45nm design exploration *ISQED '06,* 2006.

[6] R. Kumar and G. Hinton, "A Family of IA Processors," ISSCC, 2009

[7] Seevinck, E. *et al.*, "Static-noise margin analysis of MOS SRAM cells," *J. Solid-State Circuits*, vol. SC-22, no. 2, pp. 748–754, May 1987.

[8] Bhavnagarwala, A. *et al,"*A Sub-600-mV, Fluctuation Tolerant 65-nm CMOS SRAM Array With Dynamic Cell Biasing *J. Solid-State Circuits,* 2008*,* 43, 946-955.