Detecting and Managing Device Reliability in Block-level and Chip-level Simulations

LSI Corporation
Aaron Symko
aaron.symko@lsi.com

Session #9.14

Presented at

cadence designer network

Silicon Valley 2007

Abstract

Excessive voltage can be a significant reliability concern for active devices in an integrated circuit; thus, individual devices that pose a reliability risk must be identified during the circuit design process. This session will present a methodology for detecting over-voltage conditions that can lead to common reliability failure mechanisms in submicron integrated circuits. The specific failure mechanisms we aim to detect are time dependent dielectric breakdown (TDDB), hot carrier aging (HCA), and negative bias temperature instability (NBTI). Our device reliability methodology includes automated detection of over-voltage conditions; summary reporting for devices that pose a reliability risk; and a tool for displaying, managing, and signing off on over-voltage warnings from block-level and chip-level verification simulations.. The methodology has been proven using simulation results from block-level simulations with Spectre and chip-level simulations with AMS-Designer. The methodology presented enables LSI to significantly reduce verification cycle time, gain market share by improving customer confidence and minimize costs by reducing field failures of our parts.

1. Introduction

In second half of 2006, the Preamp group at LSI moved to a smaller technology feature size for which device lifetime and reliability became an increasing concern. Our designers needed a methodology to detect over-voltage and reliability issues during the design phase. For prior technologies, just detecting device breakdowns was sufficient. However, for the new technology node, we additionally needed device reliability information; specifically, we needed to know the estimated lifetime of each device. Thus, we developed a flow with two distinct aspects: over-voltage detection and reliability (lifetime) reporting.

Over-voltage conditions are easily verified with a fixed voltage test methodology, in which fixed limits are set for various FET device junctions [1]. A device that operates outside the fixed voltage ranges, at any time point, is considered to be in an over-voltage condition. Furthermore, an over-voltage condition may be intentional or unintentional. Intentional over-voltages are designed with careful review based on product specifications [1]. However, unintentional over-voltages can stem from incorrect connections or unexpected voltage spiking. It is these types of conditions which must be thoroughly investigated to determine if device reliability has been compromised. Because the over-voltage approval process is time consuming and resource intensive, it is highly desirable to have the ability to save the approved warnings in a database and auto-apply them to later re-simulations. Note that a fixed over-voltage methodology is only intended to be a sanity check and does not guarantee that device reliability criteria is met [1].

The three main degradation mechanisms for FET devices are time dependent dielectric breakdown (TDDB), hot carrier aging (HCA), and negative bias temperature instability (NBTI) [1]. Each aforementioned mechanism will degrade specific device operating characteristics and shorten usable operating life. The lifetime of a given device, with respect to the three main failure mechanisms, is the length of time for which the device will operate within specification of its operating characteristics. Moreover, the total integrated circuit (IC) failure rate, with respect to TDDB, can be calculated by accumulating reliability and gate area information for every FET device contained on the IC.

TDDB is the long term wear-out of the insulating properties of a CMOS gate leading to the formation of a conducting path through the oxide to the substrate. Furthermore, TDDB is strongly correlated to the number of defects in the gate oxide during fabrication [2]. The main symptoms of TDDB are increased gate leakage current and loss of the Ids vs Vgs relationship. For the new Preamp process being used at LSI, TDDB is strongly correlated to Vgs, temperature, and device area.

HCA, which is also commonly referred to as HCI (Hot Carrier Ionization), occurs in a FET during inversion as high velocity carriers (i.e. hot-carriers) accelerate through the pinch-off region. When a hot carrier collides with an atom near the drain depletion region, it can produce an electron-hole pair in an impact ionization event. These scattered carriers can then become trapped charge in the gate oxide interface. The side-effects of HCA are diminished carrier mobility and reduced transconductance. The lifetime of a device due to HCA decreases with increasing Vds. In addition, both temperature and device length can be variables in modeling HCA.

The last major reliability effect that is important to the Preamp group is NBTI, in which temperature induced stress under DC conditions causes the generation of interface traps between the gate oxide and the silicon substrate. Because holes interact

more readily with oxide states, PMOS devices are generally more susceptible to NBTI effects [3]. The symptoms of NBTI are threshold voltage shifts and reduced drive current. For the new Preamp process used at LSI, NBTI is only modeled for 1.5V PMOS devices, for which lifetime decreases with increasing Vgs and increasing temperature.

2. Previous Method vs New Method

The original method of determining reliability was to post-process the warnings from a fixed-test over-voltage methodology. During the post-processing, each over-voltage duration (for each device instantiation) was accumulated. Then, the accumulated duration was scaled relative to total simulation time and applied to generic (technology independent) reliability equations. Furthermore, the duty cycling (which is the accumulated duration divided by total simulation time) could be adjusted by another multiplier (from 0 to 1). This methodology was undesirable for several reasons: (1) a large number of warnings were generated; (2) effective IC failure rate was not available; and (3) reliability results were approximate.

The primary reason that large amounts of warnings got generated was that the over-voltage limits had to be set low enough to get an adequate number of data points (i.e. warnings) for the reliability post-processing. Because of the high number of warnings generated, it became too cumbersome to inspect each over-voltage condition manually. Because of this, it was easy to ignore individual over-voltage conditions for a device when no long-term reliability risk existed (based on the post-processing report). By not inspecting individual warnings, one cannot guarantee that critical effects of over-voltage conditions are caught before tapeout (e.g. incorrect connections, device breakdown, etc). Another side effect to the vast number of warning statements was that it caused designers to miss other important simulator or circuit warnings in the log file.

A second disadvantage of the previous methodology was that there was no way to easily predict effective IC reliability from the results of a block-level simulation (for TDDB). If a designer can get an estimated IC failure rate (based on a scaling of the simulated block), then total reliability concerns can be addressed early in the design phase, before chip-level simulations begin. In this manner, chip-level simulations can be used for reliability validation, and not as a means for first-pass reliability inspection.

The last major disadvantage of the previous method was the accuracy of the results. The accuracy was limited for a given device because reliability calculations only occurred during over-voltage conditions, and not at every time-point along the voltage waveform. Furthermore, generic reliability equations are generally not as reliable as technology-specific equations formulated form empirical silicon measurements.

During the formulation of a new over-voltage and reliability methodology, it was noted that some tools, like Ultrasim, have the ability to model MOS reliability effects. However, we want a methodology that works across the various tools we use for block-level and chip-level simulations, namely Spectre and AMS(Spectre). It is critical that we have the ability to run reliability analysis in Spectre because it is our high-accuracy, proven block-level simulator.

In order to overcome the limitations of the previous methodology and meet the needs of the Preamp group at LSI moving forward, a flow was developed that incorporates the following key aspects:

- Over-voltage and reliability are separated into two distinct flows
- A fixed over-voltage flow is used for "sanity" checks only, and thus voltage limits are set at reasonable values
- Reliability analysis integrates over the entire waveform for accurate results of HCI, NBTI, and TDDB
- The effective IC failure rate (for TDDB) is calculated for every block-level and chip-level simulation
- A full methodology exists for over-voltage management and signoff

3. Flow Overview

As can be seen in figure 1, the over-voltage and reliability checking flow is divided into four main parts: block-level over-voltage, block-level reliability, chip-level over-voltage, and chip-level reliability. The differences in block-level and chip-level simulation environments and accuracy/performance tradeoffs necessitated the need for distinct flows. Furthermore, because over-voltage and reliability checking are inherently different mechanisms, there was a natural choice to make separate flows for that as well.

Both over-voltage flows and the block-level reliability flow rely on VerilogA modules to be instantiated in parallel with each device in the circuit.

These VerilogA modules continuously monitor the node voltages of each device (at each time step). In the case of over-voltage, the mosov.va module provides fixed voltage testing. For reliability, the mosrel.va module calculates and accumulates reliability at each time step. In the block-level flows, a simControl spectre view allows the user to enable or disable the VerilogA module instantiation (for performance reasons). It should be noted that the mosov.va modules are used by default (when no simControl module is used). The simControl module is also used for customizing the reliability report (see section 4 for more detail).

The chip-level flow relies on TCL functions to process every device voltage at a specified time-step. For this reason, reliability checking at the chip-level is limited to steady-state conditions only. Due to the number of FET devices in a chip-level simulation, it is not currently feasible to use VerilogA module instantiations to continuously monitor voltages.

4. Detailed Block-level Flow

The detailed CAD flow for block-level over-voltage and reliability is shown in figure 2. The green blocks represent aspects of the flow that were developed by the local Preamp CAD group and released as part of a technology Independent Design Kit (IDK). The purple blocks on the right-side of figure 2 indicate items supplied by the centralized DPO CAD team and are contained as part of the process-specific design kit (PDK). Finally, the blue-colored blocks in figure 2 represent Cadence-specific programs and tools.

The flow begins in a standard way by invoking the Cadence OSS Direct Netlister to produce a netlist from schematics. As mentioned previously, a special control element, called simControl, can be instantiated in the chip-level testbench to control and configure the level of over-voltage and reliability checking (see figure 3). By setting CDF parameters, the user can control whether over-voltage, reliability, or both types of checks are enabled. Furthermore, the summary report can be configured (if reliability is enabled) with the simControl CDF parameters. Note the summary report is discussed in more detail in the "Reliability Summary Reporting" section. During the netlisting process, a netlist procedure creates a setup file (in the netlist directory) that is later sourced when the simulator is invoked. If no simControl element is present, the default environment enables over-voltage checking only.

The next phase is for the netlist, models and VerilogA check modules from the PDK to be read into Spectre. The VerilogA check modules include mosov.va for fixed over-voltage checking; mosrel.va for continuous reliability checking; and rellife.fun which contains reliability equations for various device types and operating conditions (e.g. accumulation and inversion). The models are set up as subcircuits, which include the actual FET transistor model and a master check module instantiated in parallel. The master check module can include a mosov.va instantiation, a mosrel.va instantiation, or both; as a result, three different types of master check modules are available. The specific master check module to be instantiated is determined by an environment variable, called $CKTSIM_CHECK, that is used in include statements in the models. Note that a re-simulation is needed to change the type of checking (mosov, mosrel, or both) because the type of master check module is determined by evaluating $CKTSIM_CHECK during elaboration (or circuit read-in).

After circuit read-in and elaboration, the next step is to simulate the circuit. In this phase of the flow, Spectre is invoked through a wrapper script for several reasons. The primary reason is that the VerilogA check modules for reliability (e.g. mosrel.va) will output reliability messages to stdout for every device in the circuit. Note that for over-voltage, having the warnings print directly to stdout (via mosov.va) is desirable. However, for reliability (mosrel.va), a stdout filter is needed to parse and capture the messages into a special file (lifetime.csv) for later post processing (note that the details of this will be discussed later in the "Reliability Summary Reporting" section). The second reason a wrapper is needed is to have post-exec functions occur automatically at the end of a Spectre simulation. The first post-exec function to run is the reliability summary reporting program, ovSummary. In short, ovSummary will sort and organize reliability messages that meet a given criteria. In addition, it will perform extra block-level calculations for TDDB, by utilizing equations present in rellife.tcl. The rellife.tcl file contains the TCL equivalent of the VerilogA equations that are contained in the rellife.fun file. As it runs, ovSummary prints to the end of the spectre.log file. The second post-exec function, simMailer, emails the log file to the user. It should be noted that the vast majority of simulations in the LSI Preamp group are run through batch submission via LSF. Thus, having the jobs auto-emailed is highly desirable so that users do not have to wade through a lengthy UNIX directory structure

to see the reliability output. In order to avoid unnecessary emails, the simMailer program detects whether the job was run in batch or interactive mode (based on the directory structure used and the environment variables that are set). The simMailer program was developed in order to avoid problems with the default mechanism present in the analog design environment (ADE) distributed job control [4]. The final feature of the wrapper, which was not a required feature, allows pre-exec functions before the spectre executable is called. The only program being called in the pre-exec hook is simPrintEnv, which dumps the environment at the time of the simulation to a text file.

Because mosov.va and mosrel.va modules get instantiated for every device in the circuit, simulation performance will often be degraded. From simulations over a variety of circuits, the CPU run time increased anywhere from 10% to 100% when reliability checking was enabled. For block-level simulations, the run time degradation was an acceptable tradeoff for the increased verification we gained. This is in contrast to the chip-level, for which a new method was needed and is discussed in the next section.

5. Detailed Chip-level Flow

Figure 4 presents the chip-level over-voltage and reliability flow in detail. As with the block-level flow chart, the green, purple, and blue blocks represent aspects of the IDK, the PDK, and Cadence toolsets respectively. The chip-level flow deviates significantly from the block-level flow in that we don't use the mosrel.va check modules, due to its simulation overhead (both CPU time and memory usage). Thus, we have developed a method for calculating steady-state (DC) device reliability using the TCL command line interface available in the Cadence IUS toolset.

The flow begins by invoking Cadence's amsdesigner program to netlist each cellview in the design into a use5x structure under a directory named amstmp. After netlisting, the next step is to catalog devices for reliability analysis during simulation. As the first step in the cataloging process, a custom program called amsPrep traverses the configured hierarchy in the dfII environment by means of an "icfb –nograph" process. In traversing the hierarchy, an intermediate file is created (not shown in figure 4), that contains each cellview in the design, along with each hierarchical instance path (i.e. scope) at which the cellview occurs in the design. Then, all kit-level devices (e.g. FET's, resistors, capacitors, BJT's, etc.) in the design are cataloged by parsing the netlist for each cellview listed in the intermediate file. After each cellview's netlist is parsed, the full hierarchical path for each kit device found is written to a .device_list.txt file in the amstmp directory. After all cellview's are cataloged, the .device_list.txt represents a flattened view of the analog portion of the design. Note that in addition to cataloging devices for reliability analysis, amsPrep performs additional pre-simulation operations specific to the Preamp infrastructure, which are beyond the scope of this paper.

After netlisting and creation of the .device_list.txt file, the design is compiled and elaborated. During elaboration, the PDK model files and VerilogA check modules are read in. Again, note that for the chip-level flow we only use the mosov.va (i.e. over-voltage) check modules.

After elaboration, the simulation phase begins and the user-controlled TCL input file (userRunCmd.tcl) is executed. In this file, a custom TCL command called ovReport can be called at any time during the simulation. A typical Preamp part has four modes of operation: SLEEP, IDLE, READ, and WRITE. At the end of each mode, a steady-state condition is reached during which a snapshot is saved, power analysis is done, and the reliability reporting (via ovReport) can be performed. Figure 5 shows a portion of the userRunCmd.tcl in which sleep mode steady-state is reached. The ovReport command reads in the .device_list.txt file and extracts the value of every node of every device. Then, the device node values get passed to an ovCheck TCL procedure that evaluates the device type. For valid devices (currently, this includes FET's only), the ovCheck procedure applies the appropriate reliability equations from rellife.tcl. Then, the reliability information is returned to ovReport which outputs to a uniquely named CSV file. The name of the CSV file will depend on the arguments to the ovReport function call. For example, the ovReport call in figure 5 results in a lifetime_sleep.csv file being created. After all the devices in .device_list.txt have been evaluated, the ovReport function executes the ovSummary reporting program, in which the device lifetimes (extracted from the CSV file) are sorted and organized based on the options given. Finally, the ovSummary program prints all output to the ncsim.log just after the ovReport call. It should be noted that the ovSummary program is the same as that used in the block-level flow, and is discussed thoroughly in the next section.

6. Reliability Summary Reporting

The ovSummary program used in the block-level and chip-level flows takes three inputs: (1) the CSV file with all reliability data; (2) the summary report type; and (3) an option to the summary report type.

The CSV file contains the reliability data for every device in simulated circuit. Each row in the CSV file corresponds to a single device and has the following twelve fields:

- full device instance-path
- device type
- reliability type
- dc voltage (if applicable)
- temperature
- length
- width
- lifetime (or 63% lifetime for TDDB)
- beta (used in reliability equations)
- ppm (used in reliability equations)
- device type code (used internally in code)

The full device instance path in field one is a dot-delimited path from the testbench level down to the device level. It is equivalent to the %M formatting code in VerilogA. Device type is given in the second field and is a string representing the oxide voltage limit, the majority carrier type, and the specific device junction tested. In the preamp process the two oxide types are 1.5V and 3.3V. Field three gives the reliability type, which can be one of the three main failure mechanisms described earlier (TDDB, HCA, and NBTI). The fourth field lists the dc voltage when the mosrel.va modules are enabled under a "dc" analysis. For transient analysis, field four will be blank since the entire waveform is integrated and a single dc-like value is not applicable. Field five indicates the temperature over which the simulation was run. Because our version of Spectre cannot vary the temperature between devices in a single simulation, field five will always be the same value (in every row) for a single CSV file. The next two fields indicate the length and width of the device in microns. In field eight, the lifetime of the device (for the given reliability type) is shown. For HCA and NBTI this is the full device lifetime, but for TDDB the number represents only 63% of the total lifetime. From the 63% lifetime number, the ovSummary program calculates the ten year reliability and 0.1% failure rate for the block-level and effective chip-level (note: this is described in more detail later). The remaining fields are device and process-specific values that are used internally by the reliability equations (in rellife.tcl) and in the ovSummary program code. If a CSV file is not supplied to the ovSummary program, then a default of filename of "./lifetime.csv" is used.

After the CSV filename, the next two inputs to the ovSummary program are the report type and report option. The report type can have the value of "lifemin" or "worstcase"; and the meaning of the report option value depends which of these is given. For the "lifemin" option, the report type is the minimum lifetime in years. In this manner, indicating a "lifemin" option instructs ovSummary to report every device with a lifetime of less than or equal to the lifetime given in the report option. When the "worstcase" option is given, the report type indicates the number of devices to report with the shortest lifetimes. For both the "lifemin" and "worstcase" options, the devices are sorted first by reliability type, then in ascending order with respect to lifetime. Thus, the worst (i.e. shortest life) devices are shown first within each reliability type grouping.

A sample reliability report summary is split between figures 6, 7 and 8. The start of the summary report (figure 6) indicates the temperature at which the simulation was run, which is extracted from field five of the CSV file. The first reliability types to get reported (due to alphabetical sorting) are HCA and NBTI. From Figure 6, it can be seen that for each device that meets the summary criteria (in this case lifemin less than or equal to 10 years) the following is reported: full instance path, device type, DC voltage (when applicable), length, width, and lifetime. The ovSummary program reads this information directly from the CSV file, and thus the program merely sorts and reports each device that meets the summary criteria. Note that for NBTI, only the 1.5V PMOS devices have reliability information (because equations have not been formulated for other device types).

As indicated earlier, extra reporting is done for TDDB, which varies statistically with gate area [5]. Figure 7 shows an example of the actual TDDB summary. The first thing that gets reported is the reference gate area in square microns. The reference gate area is measured from a representative preamp part in the same process and stored in a setup file that the ovSummary program reads in at startup. The gate area is partitioned into the different oxide types which for our process is 1.5V or 3.3V. This is done so that the total reliability for each type can be reported separately. The next item to be reported is the block gate area, which is calculated on-the-fly by the ovSummary program from the length and width of every device in the CSV file (i.e. every device in

the simulated block). Like the reference gate area, the block gate area is split into 1.5V and 3.3V bins. The next section of the summary gives the total, actual TDDB ten year reliability for each device type. The ten year reliability number (for actual TDDB) represents the percentage of integrated circuits (ICs) that do not fail due to TDDB after ten years, where the entire IC is assumed to be no more than what is in the simulated block. The ten year reliability number for actual TDDB can be calculated from the 63% lifetime as follows:

$$10\ \ Year\ \ \mathrm{Re}liability\ \ (ActualTDDB) = \left[\prod_{i=0}^{N} f_{10rel_1.5v_device_i}(A_{actual}, T_{63})\right] * \left[\prod_{j=0}^{M} f_{10rel_3.3v_device_j}(A_{actual}, T_{63})\right]$$

Equation 1. Ten year reliability for actual TDDB as a function of 63% lifetime and area

where A_{actual} is calculated as the device length (L) multiplied by the device width (W); N is the total number of 1.5V devices in the simulated block, M is the total number of 3.3V devices in the simulated block; and parameters L, W and 63% lifetime (T_{63}) are read in from the CSV file (from fields six, seven, and eight respectively). Under the oxide-specific ten year reliability number in the summary report, the 0.1% lifetime for actual TDDB is shown. The 0.1% lifetime number (for actual TDDB) represents the amount of time it would take 0.1% of the ICs to fail, where the entire IC is assumed to be no more than what is in the simulated block. Note that the 0.1% lifetime for actual TDDB can be calculated directly from the ten year reliability number. Next, the total ten year reliability for actual TDDB is shown, which is the combined reliability for the 3.3V and 1.5V device types (i.e. the result of Equation 1). For convenience, the ten year failure rate is also given, which is simply one minus the ten year reliability. Finally, the individual devices that meet the summary criteria are given.

The scaled TDDB data shown in figure 8 is arranged similarly to that of the actual TDDB data just described. The key difference (between actual and scaled) is that for the scaled TDDB equations, an effective area is used and is defined as the ratio of reference gate area to block gate area relative to oxide type. The effective area equation for any given device is as follows:

$$A_{effective}^{oxide} = \left[\frac{A_{refchip}^{oxide}}{A_{block}^{oxide}}\right] * L_{device}^{oxide} * W_{device}^{oxide}$$

Equation 2. Effective area used in lifetime calculations for scaled TDDB.

where the oxide superscript in each variable can be either 1.5V or 3.3V (for our technology). The $A_{refchip}$ is the total reference gate area, A_{block} is the calculated gate area for the simulated block, and L_{device} and W_{device} are the device length and width (from fields six and seven in the CSV file).

The ten year reliability number for scaled TDDB is the percentage of ICs that do not fail due to TDDB if the (simulated) block were increased to the size of the reference gate area. The ten year reliability for scaled TDDB can be calculated from the scaled 63% scaled lifetime as follows:

$$10\ \ Year\ \ \mathrm{Re}liability\ \ (EffectiveTDDB) = \left[\prod_{i=0}^{N} f_{10rel_1.5v_device_i}(A_{effective_1.5v}, T_{63})\right] * \left[\prod_{j=0}^{M} f_{10rel_3.3v_device_j}(A_{effective_3.3v}, T_{63})\right]$$

Equation 3. Ten year reliability for scaled TDDB as a function of 63% lifetime and effective area

where $A_{effective}$ is the oxide-specific effective area, as given in Equation 2; N is the total number of 1.5V devices in the simulated block; M is the total number of 3.3V devices in the simulated block; and 63% lifetime (T_{63}) is read in from the CSV file (from field eight). Under the oxide-specific ten year reliability number in the summary report, the 0.1% lifetime for scaled TDDB is shown. The 0.1% lifetime (for scaled TDDB) represents the time it would take 0.1% of the ICs to fail if the (simulated) block were increased to the size of the reference gate area. Note that the 0.1% lifetime for scaled TDDB can be calculated directly from the ten year reliability number. At the end of the scaled TDDB report, the total ten year reliability, failure rate, and individual device lifetimes (for the given summary criteria) are printed. Note that the scaled lifetimes for each device are calculated using the effective area (from Equation 2).

The ovSummary program provides a concise and useful reliability overview of the simulated circuit. It is the responsibility of the circuit designer to interpret

the results and signoff on or correct each offending device. Furthermore, by giving effective IC reliability numbers (for TDDB) at the block-level, the designer has a responsibility to ensure that the simulated block is safely within the chip-level specification. In this manner, reliability problems can be addressed early in the design process. Then, chip-level simulations can be used for reliability validation, and not as means for first-pass reliability inspection.

7. Over-voltage Management and Signoff

In the preamp group at LSI, designers are required to inspect the individual over-voltage conditions that occur in a simulation, even if they pose no long-term reliability risk (i.e. are not listed in the reliability summary report). As stated earlier, voltage spikes, glitches, or incorrect connections can cause over-voltages and thus each occurrence must be approved or corrected. Because the approval process is a manual and time-consuming task, it is desirable to remove previously approved warnings during re-simulations of the same block or in a simulation of a block farther up in the design hierarchy.

The over-voltage management and signoff flow we developed is shown in figure 9. It begins by running a block-level or chip-level simulation in either Spectre or AMS(Spectre). For an AMS simulation, the user can optionally add logEvent procedure calls to the TCL run file. The logEvent procedure maps a mnemonic name to an event transition at the time of the logEvent call. Mnemonics are a key aspect of the approval database and correspond to a specific transition or mode of the Preamp part. An example mnemonic might be called "SLEEP" to signify the sleep mode transition. Another example would be "RH2H" to designate the read mode head-to-head switching transition (in a 2 channel part). By assigning mnemonics to different transitions, the exact transition time does not have to be aligned with each warning in the approval database; which may vary somewhat with design changes, dynamic simulation timesteps, and different testbench setups.

After the simulation completes, the user invokes a custom GUI (figure 10) to begin the over-voltage management and signoff process. Once the testbench, path of an instance within the testbench (i.e. the DUV), and simulation log file are selected, all over-voltage warnings are parsed and loaded into the GUI by clicking the "Load Approval Database" button. Note that the DUV can be any instance under the testbench for which warnings should be approved. When multiple DUV instances exist the approval process can be repeated many times within the same GUI session by simply updating the DUV field value. Next, the user can assign any missing mnemonics that did not get applied during the simulation (via the logEvent TCL proc). Currently for block-level Spectre simulations, the user must manually assign mnemonics in the GUI. However, a future methodology may allow assignment of mnemonics during Spectre-only simulations (see the "Conclusions and Future Work" section). Every warning must have an associated mnemonic before approval occurs (either via loading an approval database or by manually clicking the "Approve Selected Items" button).

In order to aid in the approval of warnings, a database from any sub-cell in the hierarchy can be loaded and applied to the existing warnings in the "unapproved" tab. Warnings transfer from the "unapproved" tab to the "approved" tab in one of two ways: (1) after the "Approve Selected Items" button is clicked or (2) after a pre-approved database is loaded (by clicking the "Load Approval Database" button). By clicking the "Create Approval Database" button, an XML approval database is created from all warnings under the "approved" tab.

The XML approval database is stored as a unique cellview under the DUV cell in the DFII use5x directory structure, and thus can be managed with the cell via third party versioning software. The database is an XML file that contains an element for each approved warning with the following child elements: an instance string, the event mnemonic, the over-voltage duration, and the peak voltage value (if available). It should be noted that the instance string is a concatenation of the full instance path and the junction for which the over-voltage condition occurs. When a database is loaded into the GUI, warnings with the same instance strings are approved if all of the following conditions are true:

- The mnemonics are equal
- The unapproved duration is less then or equal to the approved duration
- The unapproved peak voltage is less than or equal to the approved peak voltage; if it exists

If the peak voltage is missing from one of the approved warnings or unapproved warnings then no approval takes place. However, if the peak voltage is absent from *both* warnings, it is not considered as a criterion for approval.

The XML file also contains elements for the library, cell, and view of the DUV selected. An example approval database file is shown in figure 11.

8. Conclusions and Future Work

A methodology has been presented that detects over-voltage conditions and calculates reliability for every device during block-level and chip-level simulations. Furthermore, a summary program (ovSummary) automatically runs to give summarized reliability results directly in the simulation log file. For TDDB, ovSummary calculates effective IC reliability so that the designer can address reliability concerns early in the design phase (instead of later during chip-level verification). Over-voltage warnings that are approved can be saved in a database for later application to re-simulations of the same block or simulations at an upper level of the hierarchy. Moreover, an over-voltage warning management and signoff GUI offers designers an easy way to sort through over-voltage warnings and approve them. Through the GUI, designers can load previously approved databases (for sub-blocks), manually approve new warnings, and create a new approval database.

Despite being a robust, feature-rich and user-friendly methodology, there are a few enhancements we plan to make.

First, we would like to offer summary reporting across corners (and other parametric simulation types). This would probably be implemented as a post-processing script that the user could manually invoke at the end of a corner, parametric, or statistical run.

Another enhancement would be the ability to define event mnemonics during block-level Spectre simulations (akin to what is implemented for the AMS flow). One possible implementation is to use the netlist procedure of the simControl block to instantiate VerilogA code to strobe event statements at desired times. Then, the instance CDF could be used to add various time/mnemonic pairs (similar to the CDF of a PWL element).

A further enhancement to the over-voltage management and signoff flow is to automatically split event transitions (i.e. mnemonics) into smaller partitions. Event durations can range anywhere from 5us to 15us and thus a single mnemonic may not have adequate resolution for future approval of over-voltages (within the event duration). One possible solution is to automatically split each event-duration into smaller partitions of a default resolution. This could easily occur when the warnings are initially gathered (see figure 9). For each sub-range of a given event duration, mnemonics could be created by appending unique identifiers to the original name. For example, if RH2H has a 5us duration, and the default resolution is 1us, then five new mnemonic names would automatically be created by the GUI as follows: RH2H.1, RH2H.2, RH2H.3, RH2H.4, and RH2H.5. Note that in this example, the dot should be a reserved character to avoid name collisions (with other user-defined mnemonics).

A final improvement to the over-voltage management and approval flow would be to offer formatting of the XML approval database with extensible style sheets (XSL); which is touted as the preferred style sheet language by the World Wide Web Consortium (W3C) [6]. By formatting with XSL, the XML would have a user-friendly layout that could be read by any (modern) web browser.

As an ongoing effort, we are continually evaluating Ultrasim as an option for both block-level and chip-level simulations. Thus, at some future date we may consider using the built-in over-voltage and reliability analysis that Ultrasim offers. However, in the near term, our plan is to use our internally developed mosov.va and mosrel.va modules with the Spectre solver.

In conclusion, the over-voltage and reliability methodology implemented for the Preamp group at LSI helps ensure high reliability parts, which reduces field failures, maintains LSI's standing as a high quality provider, and ultimately helps to maximize Preamp revenue.

9. References

[1]. "Testing for Excessive MOS Voltages", Agere Technical Memo, D. Averill Bell, Kausar Banoo, Shawn Boshart, Scott Dickinson, Peg French, Chester Leung, Ed Morgan, Phil Mason, Bonnie Weir, Randy Wolf, December 15, 2006

[2]. "Reliability in CMOS IC Design: Physical Failure Mechanisms and their Modeling", MOSIS Technical Note, www.mosis.org

[3]. "NBTI: A Growing Threat to Device Reliability", Laura Peters, Senior Editor, Semiconductor International, March 1 2004

[4] "DP Email Notifier Broken in 5033USR1", PCR 683428, Eric Braun, Jan 2004

[5] "Excessive Voltage Checking", D. Averill Bell, Internal Agere Presentation, July 28, 2004

[6] "XML Tutorial", W3 Schools, www.w3schools.com/xml/default.asp

10. Acknowledgments

The methodology presented in this paper could not have been realized without the efforts of many talented and hard-working individuals who came together to form a cross-functional and cross-department team.

First, the author would like to thank individuals in the LSI Device Modeling group, including Bonnie Weir, Kausar Banoo, and Averill Bell. **Bonnie Weir** was responsible for developing the technology-specific lifetime equations, which are the heart and soul of the reliability flow. Without her efforts, a lifetime-based reliability flow would not have been possible. **Kausar Banoo** created the check modules (mosov.va and mosrel.va) that apply the over-voltage limits and the reliability equations. Through her diligence and hard-work, the modules are both accurate and efficient. Furthermore, Kausar filled a critical role as primary modeling contact for the Preamp group and collaborated on overall flow implementation. **Averill Bell**, who consulted on the check module development and methodology, was an invaluable resource. Averill developed the original fixed-test over-voltage methodology on which the second-generation check modules were based. Furthermore, his experience and expertise were critical throughout the development of the entire flow.

Next, there are many people in the LSI Preamp Design group that helped make the flow a reality, including Dave Kelly, Steve Kuehne , Brad Natzke and Jason Brenden. **Dave Kelly** offered an experienced design perspective in forming the requirements of the reliability flow. He steered the entire team toward a correct implementation and ensured necessary features were present for making Preamp design successful. **Steve Kuehne** brought a wealth of process expertise and oversight to the project, and as a result, was also critical in developing the flow requirements and ensuring Preamp's continued success. **Jason Brenden** was one of the primary beta testers of the flow and helped debug many of the initial problems we encountered during the beta phase of the project. The over-voltage management and approval flow would not have been possible without the work of **Brad Natzke**. Not only did Brad help define flow requirements; he also developed the initial feasibility scripts that proved the methodology.

Finally, the author would like to thank **Vicki Nelson**, by providing the opportunity to work on such a critical project. Without her support and guidance, the development of new methodologies (such as the reliability flow) would not be possible within the Preamp Group at LSI.

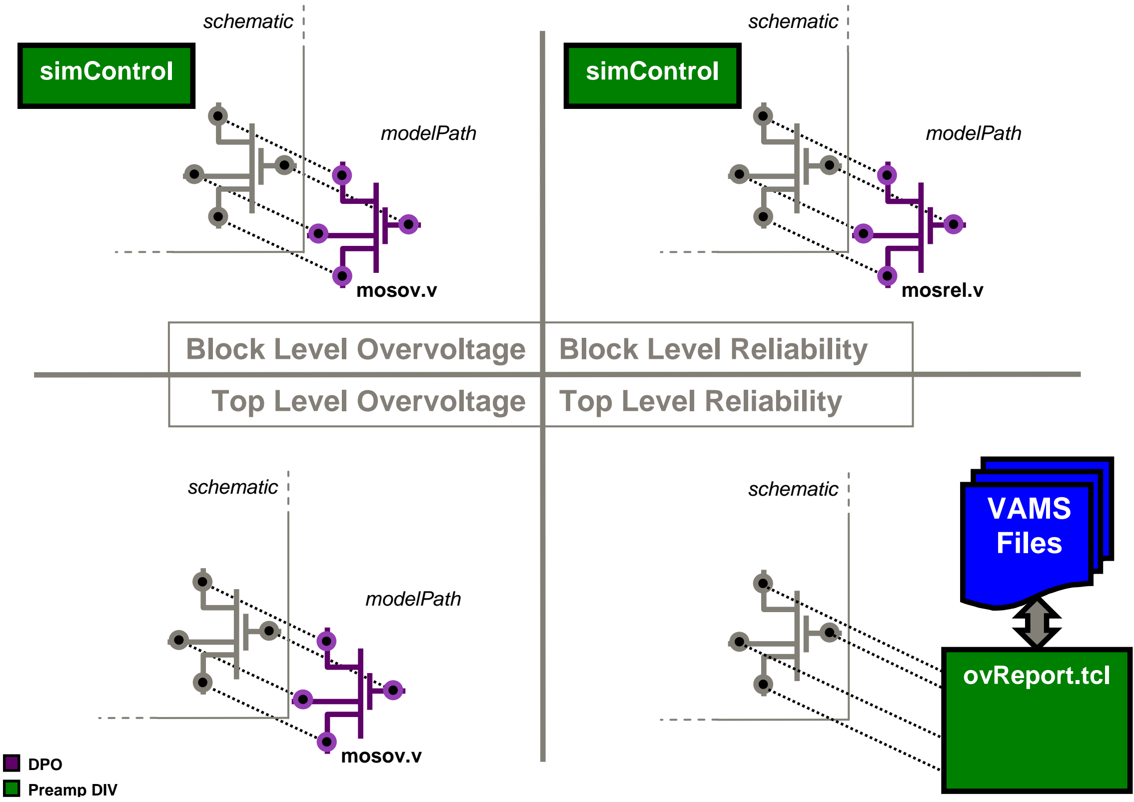

Figure 1. Over-voltage and Reliability Flow Overview

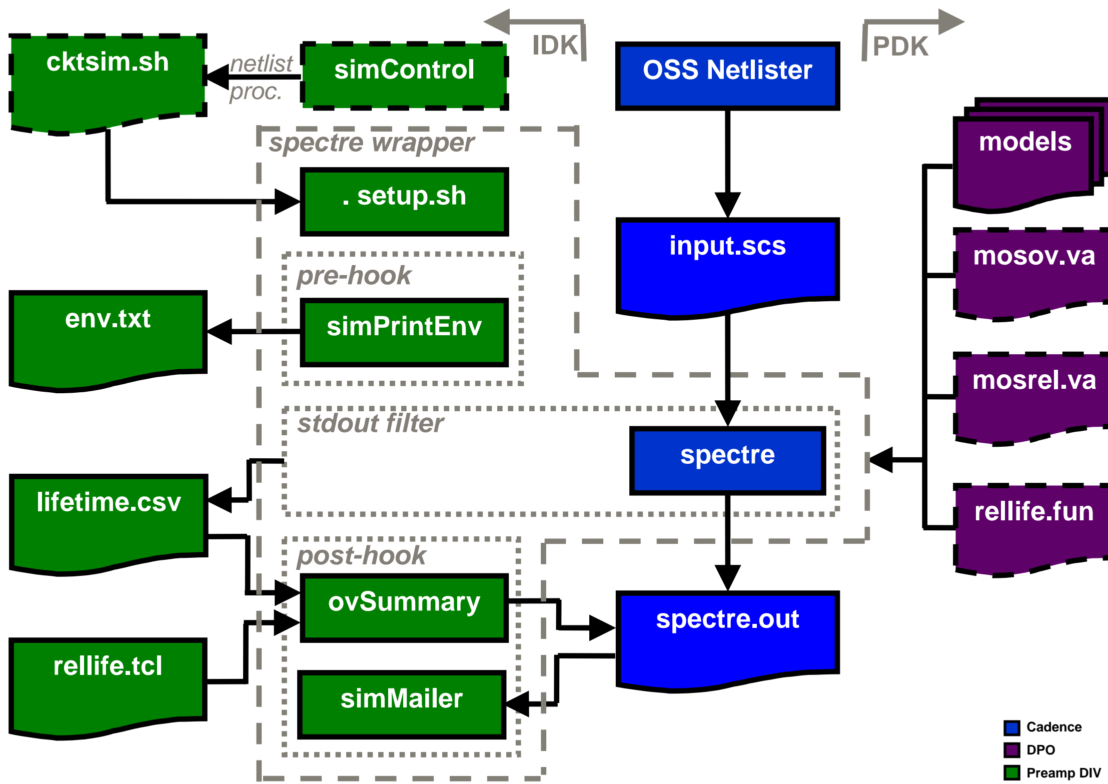

Figure 2. Detailed Block-level CAD Flow

SIMCONTROL0
simControl
Check Type = Reliability
Check Summary Type = lifemin
Check Summary Option = 100

Figure 3. Example simControl Element Instantiation

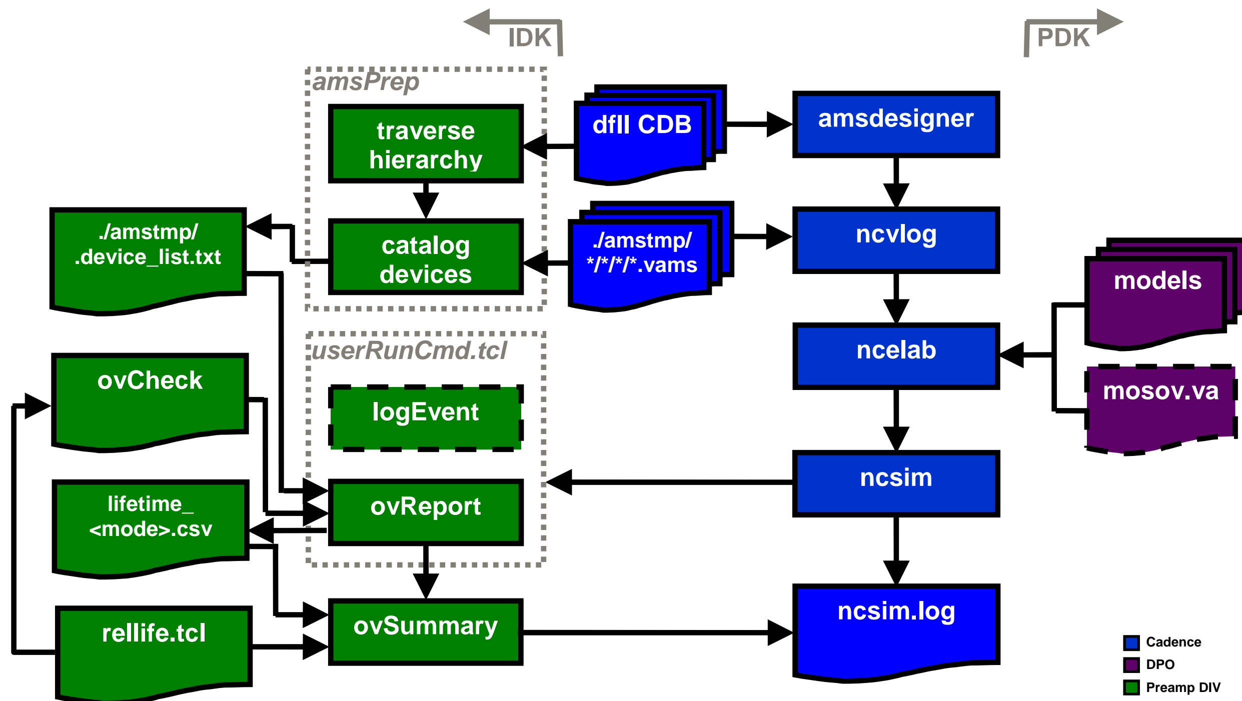

Figure 4. Detailed Chip-level CAD Flow

```
S2I2R2W.tcl - /synchronicity/workspaces/PA_COM2_Mob/COM2_Mob/symko/
File  Edit  Search  Preferences  Shell  Macro  Windows  Help

puts "The part is in SLEEP mode"

logEvent "SLEEP"

setCRbit mode 00

run 401.5us -absolute

save -simulation -overwrite :sleep

puts "ICC sleep mode is [value [scope].PWRVCC.Ipwr]A"
puts "IEE sleep mode is [value [scope].PWRVEE.Ipwr]A"
puts $simLog "ICC sleep mode is [value [scope].PWRVCC.Ipwr]A"
puts $simLog "IEE sleep mode is [value [scope].PWRVEE.Ipwr]A"

pwrReport sleep $pwrpath
ovReport sleep
```

Figure 5. Using the ovReport TCL function (userRunCmd.tcl screen shot)

```
************* Reliability Summary for com2 sg @ 70.00C **************

======================================================================
                              HCA Summary:
======================================================================

   TEST_OV.NMOS_3p3V.chk.chk1
    Device Type = 3.3V nmos
    L           = 3.200000e-01 um
    W           = 1.800000e-01 um
    Life [HCA] = 3.45 years
  TEST_OV.NMOS_3p3V_LOWTH.chk.chk1
    Device Type = 3.3V nmos
    L           = 3.200000e-01 um
    W           = 1.800000e-01 um
    Life [HCA] = 3.45 years
…

======================================================================
                             NBTI Summary:
======================================================================

 TEST_OV.CPMOS_1p5V.chk.chk1
    Device Type = 1.5V pmos
    L           = 1.005002e+01 um
    W           = 1.005002e+01 um
    Life [NBTI] = 2.38 years
  TEST_OV.CPMOS_1p5V.chk.chk1 (DC)
    Device Type = 1.5V pmos
    Vgs         = -2.500000 VDC
    L           = 1.005002e+01 um
    W           = 1.005002e+01 um
    Life [NBTI] = 2.38 years
…
```

Figure 6. Example HCA and NBTI Summary Reporting (summary type=lifemin, summary option = 10 years)

```
==============================================================
                     TDDB_Actual Summary:
==============================================================

Reference Gate Area (pa2540) :
  3.3  Devices : 3.989080e+05 um^2
  1.5  Devices : 3.054050e+05 um^2

Block Gate Area (calculated) :
  3.3  Devices : 3.192714e+02 um^2
  1.5  Devices : 2.044593e+02 um^2

Total TDDB_Actual Reliability :
  3.3  Devices :    10 Year Reliability = 0.999996
                    0.1% Lifetime       = 63.48 years
  1.5  Devices :    10 Year Reliability = 0.999980
                    0.1% Lifetime       = 29.13 years

Total 10 Year Reliability  [TDDB_Actual] : 0.9999757153
              Failure Rate [TDDB_Actual] : 0.0000242847

  TEST_OV.CNMOS_1p5V.chk.chk1
    Device Type = 1.5V nmos
    L           = 1.016002e+01 um
    W           = 1.016002e+01 um
    Life [TDDB_Actual] = 63.21 years
  TEST_OV.CNMOS_1p5V.chk.chk1 (DC)
    Device Type = 1.5V nmos
    Vgs         = -2.500000 VDC
    L           = 1.016002e+01 um
    W           = 1.016002e+01 um
    Life [TDDB_Actual] = 63.21 years

...
```

Figure 7. Example TDDB (actual) Summary Reporting (summary type=lifemin, summary option = 10 years)

```
==================================================================
                      TDDB_Scaled Summary:
==================================================================

Reference Gate Area (pa2540) :
  3.3  Devices : 3.989080e+05 um^2
  1.5  Devices : 3.054050e+05 um^2

Block Gate Area (calculated) :
  3.3  Devices : 3.192714e+02 um^2
  1.5  Devices : 2.044593e+02 um^2

Total TDDB_Scaled Reliability :
  3.3  Devices :    10 Year Reliability = 0.995057
                   0.1% Lifetime        = 5.89 years
  1.5  Devices :    10 Year Reliability = 0.970105
                   0.1% Lifetime        = 81.13 days

Total 10 Year Reliability  [TDDB_Scaled] : 0.9653098458
              Failure Rate [TDDB_Scaled] : 0.0346901542

  TEST_OV.CNMOS_1p5V.chk.chk1
    Device Type = 1.5V nmos
    L           = 1.016002e+01 um
    W           = 1.016002e+01 um
    Life [TDDB_Scaled] = 176.08 days
  TEST_OV.CNMOS_1p5V.chk.chk1 (DC)
    Device Type = 1.5V nmos
    Vgs         = -2.500000 VDC
    L           = 1.016002e+01 um
    W           = 1.016002e+01 um
    Life [TDDB_Scaled] = 176.08 days
…

********************* End Reliability Summary *******************
```

Figure 8. Example TDDB (scaled) Summary Reporting summary type=lifemin, summary option = 10 years)

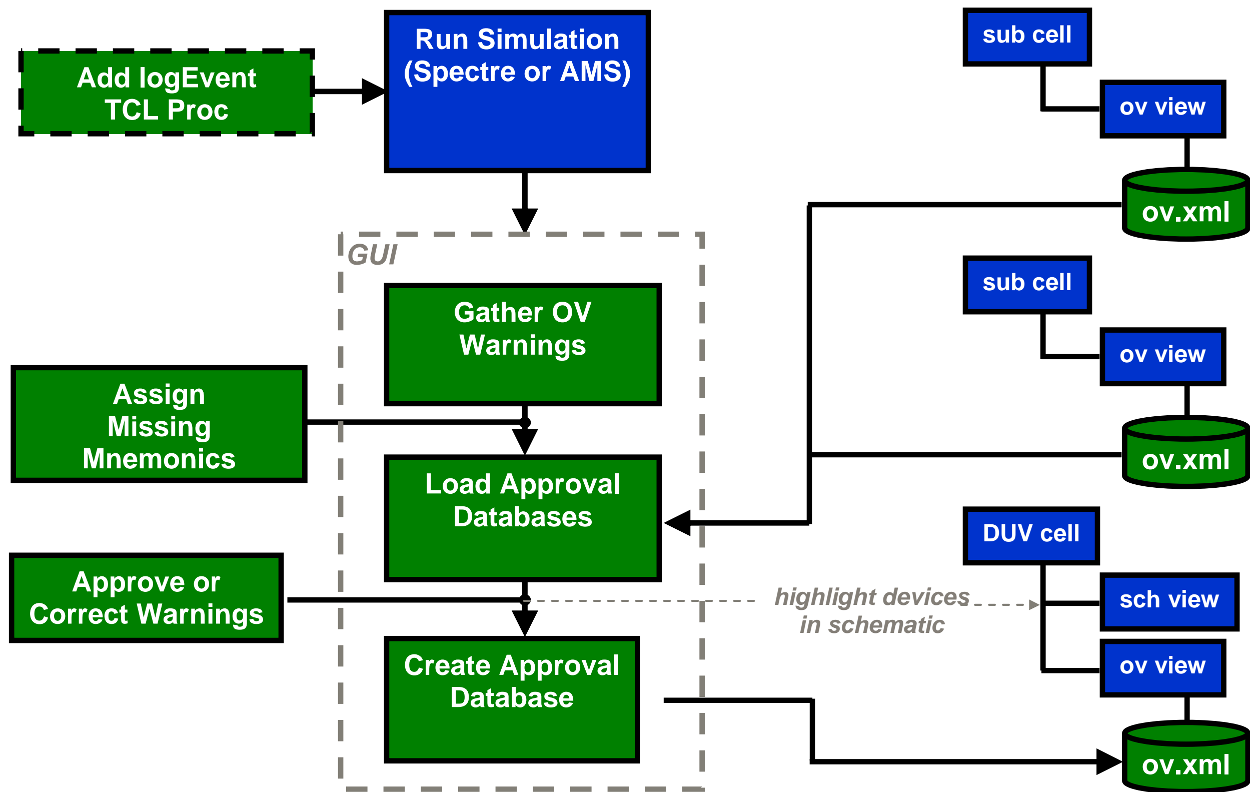

Figure 9. Over-voltage management and signoff flow.

Xcheck - OV Warning Management and Approval Form

Close | Help

Testbench Lib.Cell:View: pa7840.pa784002_b1:mixed_sch [browse]

DUV Instance Path: tb_pa784002_b1.DUV [select]

Log File Path: /synchronicity/workspaces/PA_COM2_Mob/COM2_Mob/symko/amsRunDir/pa784002_b1_run_FHC_H2H_single/ncsim.log [browse]

Load Approval Database -------------->> Approve Selected Items -------------->> Create Approval Database

Clear | Highlight | Define Mnemonic [apply]

Unapproved | Approved

<table>
<tr><th>Instance</th><th>Violation</th><th>Start Time</th><th>Duration Time</th><th>Peak Value</th><th>Mnemonic</th></tr>
<tr><td>AnalogCore.ReaderAnlg.MR_BiasControl.MR_BiasModeSwitch.C7_0.m1</td><td>|Vgd|>4</td><td>1.02418e-06</td><td>1.978e-9</td><td></td><td>RH2H</td></tr>
<tr><td>AnalogCore.ReaderAnlg.MR_BiasControl.MR_BiasModeSwitch.C4_0.m1</td><td>|Vgs|>4</td><td>1.02418e-06</td><td>1.978e-9</td><td></td><td>RH2H</td></tr>
<tr><td>AnalogCore.ReaderAnlg.MR_BiasControl.MR_BiasModeSwitch.C4_0.m1</td><td>|Vgd|>4</td><td>1.02418e-06</td><td>1.978e-9</td><td></td><td>RH2H</td></tr>
<tr><td>AnalogCore.ReaderAnlg.GainStage2.HeadCombing.M10</td><td>|Vgs|>4</td><td>1.02431e-06</td><td>2.590e-10</td><td></td><td>RH2H</td></tr>
<tr><td>AnalogCore.ReaderAnlg.GainStage2.HeadCombing.M10</td><td>|Vgd|>4</td><td>1.02431e-06</td><td>2.590e-10</td><td></td><td>RH2H</td></tr>
<tr><td>AnalogCore.ReaderAnlg.GainStage2.HeadCombing.M11</td><td>|Vgs|>4</td><td>1.02431e-06</td><td>2.590e-10</td><td></td><td>RH2H</td></tr>
<tr><td>AnalogCore.ReaderAnlg.GainStage2.HeadCombing.M11</td><td>|Vgd|>4</td><td>1.02431e-06</td><td>2.590e-10</td><td></td><td>RH2H</td></tr>
<tr><td>AnalogCore.ReaderAnlg.GainStage2.HeadCombing.M12</td><td>|Vgs|>4</td><td>1.02431e-06</td><td>2.590e-10</td><td></td><td>RH2H</td></tr>
<tr><td>AnalogCore.ReaderAnlg.GainStage2.HeadCombing.M12</td><td>|Vgd|>4</td><td>1.02431e-06</td><td>2.590e-10</td><td></td><td>RH2H</td></tr>
<tr><td>HeadCellTop.HeadCell0.ZML.M29.chk</td><td>|Vds|>1.725</td><td>1.76899e-06</td><td>2.000e-09</td><td>1.94</td><td>HtrVOn</td></tr>
<tr><td>HeadCellTop.HeadCell0.ZML.M30.chk</td><td>|Vds|>1.725</td><td>1.76899e-06</td><td>2.000e-09</td><td>1.94</td><td>HtrVOn</td></tr>
<tr><td>HeadCellTop.HeadCell1.FHC_HeadCell.FHC_Amp.M71</td><td>|Vgs|>4</td><td>1.76890e-06</td><td>1.500e-8</td><td></td><td>HtrVOn</td></tr>
<tr><td>HeadCellTop.HeadCell1.ZML.M29.chk</td><td>|Vds|>1.725</td><td>1.76899e-06</td><td>2.000e-09</td><td>1.94</td><td>HtrVOn</td></tr>
<tr><td>HeadCellTop.HeadCell1.ZML.M30.chk</td><td>|Vds|>1.725</td><td>1.76899e-06</td><td>2.000e-09</td><td>1.94</td><td>HtrVOn</td></tr>
<tr><td>HeadCellTop.VRegVP17.M12.chk</td><td>|Vds|>3.795</td><td>1.76899e-06</td><td>2.000e-09</td><td>3.89</td><td>HtrVOn</td></tr>
<tr><td>HeadCellTop.VRegVP17.M12.chk</td><td>|Vgd|>3.795</td><td>1.76899e-06</td><td>2.000e-09</td><td>3.90</td><td>HtrVOn</td></tr>
</table>

Figure 10. Over-voltage management and signoff GUI

```
<approved_ov_db>
    <library>pa7840</library>
    <cell>pa784002_b1</cell>
    <view>schematic</view>
    <warning>
        <instance>HeadCellTop.VRegVP17.M12 Vds</instance>
        <mnemonic>HtrVOn</mnemonic>
        <duration type="time" units="s">2e-9</duration>
        <peak type="voltage" units="V">3.89</peak>
    </warning>
    <warning>
        <instance>HeadCellTop.VRegVP17.M12 Vgd</instance>
        <mnemonic>HtrVOn</mnemonic>
        <duration type="time" units="s">2e-9</duration>
        <peak type="voltage" units="V">3.9</peak>
    </warning>
    …
</approved_ov_db>
```

Figure 11. Example XML approval database file