A Fast and Accurate Process Variation-Aware Modeling Technique for Resistive Bridge Defects

Shida Zhong, Saqib Khursheed, and Bashir M. Al-Hashimi, *Fellow, IEEE*

***Abstract*—Recent research has shown that tests generated without taking process variation into account may lead to loss of test quality. At present, there is no efficient device-level modeling technique that models the effect of process variation on resistive bridge defects. This paper presents a fast and accurate technique to achieve this, including modeling the effect of voltage and temperature variation using the BSIM4 transistor model. To speed up the computation time and without compromising simulation accuracy (achieved through BSIM4), two efficient voltage approximation algorithms are proposed for calculating logic threshold of driven gates and voltages on bridged lines of a fault-site to calculate bridge critical resistance. Experiments are conducted on a 65 nm gate library (for illustration purposes), and results show that on average the proposed modeling technique is more than 53 times faster and in the worst case, error in bridge critical resistance is 2.64% when compared with HSPICE.**

***Index Terms*—Fault model, manufacturing test, process variation, resistive bridge fault.**

I. Introduction

RESISTIVE bridge fault (RBF) represents a major class of defects in deep-submicron (DSM) CMOS and has received increased attention on modeling and simulation [1]. Manufacturing test employs fault models for testing digital circuits, which are meant to emulate the physical behavior of a defect at device level. Accurate fault models are important for fault simulation, test generation, and fault diagnosis. The resistance of a bridge (R_{sh}, Fig. 1) is a continuous parameter which is not known in advance. Resistive bridge changes the voltage on the bridged lines (V_1, V_0, Fig. 1) from 0 V or V_{dd} to some intermediate value, which varies with R_{sh} of the bridge fault. A number of methods have been proposed in the literature to determine the behavior of the bridge fault-site in the presence of this unknown (R_{sh}) parameter. The first fault model to take into account the intrinsic resistance of a bridge is proposed in [2], which is based on the Shockley transistor model. It uses curve fitting to match results with SPICE data to achieve high accuracy. To account for DSM behavior, a more advanced transistor model (BSIM4) is used to compute bridge critical resistance [1]. These two RBF models [1], [2] are intended for designs operating in nominal conditions, however due to continuous scaling of CMOS, DSM designs are affected by process variation [3], [4]. Fabrication process variation is mainly due to sub-wavelength lithography, random dopant distribution, line edge roughness, and stress engineering [5], [6]. In a recent study, it has been shown that more than 30% error in the drive current of a transistor is observed on a 65 nm device due to process variation, when compared to a transistor operating in nominal condition [6]. Process variation also affects the behavior of a resistive bridge defect [7]. Using ISCAS'85, 89 benchmarks and a 45 nm gate library, it was shown that tests generated for nominal scenario without considering process variation can lead to as much as 10% test escapes [7]. Two important parameters are affected by process variation leading to additional logic faults and test escapes occur due to these additional logic faults. These two parameters are drive current of driving gates (Fig. 1, D_1 and D_2) and logic threshold voltage of the driven gates (Fig. 1, S_1, S_2, and S_3). Bridge defect critical resistance calculation through the Shockley transistor model with curve fitting to match SPICE data is accurate only in nominal operating conditions and it loses accuracy under the influence of process variation [8].

The first attempt at modeling the impact of process variation on bridge faults is reported in [7] using SPICE based Monte Carlo simulation. For each bridge fault-site, it uses SPICE simulation to determine the voltages (V_1, V_0, Fig. 1) at discrete bridge resistance intervals and stores the outcome in a database for subsequent use. The nominal values of V_1 and V_0 are then used to generate new set of variation-induced logic faults by Monte Carlo simulation and for this purpose four transistor parameters [threshold voltage (V_{th}), width (W), length (L), and oxide thickness (T_{ox})] are varied through 500 permutations to generate a new set of variation-induced logic faults. This method has two limitations. First, when scaling from one technology node to another, the database (with SPICE information) needs to be re-generated, since it is technology-specific. Second, the database generation (per technology node) requires long computation time. A recent study has reported that it took nearly a week with eight computers working in parallel to generate a database for ISCAS'85, 89 benchmarks [9]. See [8] for more details on limitations of available fault modeling techniques.

In this paper, we overcome these two limitations by developing a fast and accurate model of resistive bridge defects, while incorporating the effect of process, voltage and temperature (PVT) variation. The proposed modeling technique is accurate because it uses the most recent transistor model (BSIM4,

Manuscript received March 2, 2011; revised May 20, 2011; accepted July 1, 2011. Date of current version October 19, 2011. This work was supported by the Engineering and Physical Sciences Research Council, U.K., under Grant EP/H011420/1. This paper was recommended by Associate Editor K. Chakrabarty.

The authors are with the School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K. (e-mail: sz1g08@ecs.soton.ac.uk; ssk@ecs.soton.ac.uk; bmah@ecs.soton.ac.uk).

Digital Object Identifier 10.1109/TCAD.2011.2162065

0278-0070/$26.00 © 2011 IEEE

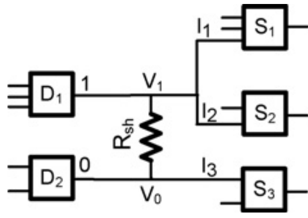

Fig. 1. Resistive bridge R_{sh} forming a potential circuit fault.

Berkeley Short-Channel IGFET Model) [10]. The proposed modeling technique is fast because it employs highly efficient voltage approximation algorithms for bridge resistance calculation and to compute logic threshold voltages of the driven gates (Fig. 1, I_1, I_2, and I_3). The effect of process variation is modeled by incorporating fluctuations in three transistor parameters: gate length (L), threshold voltage (V_{th}), and effective mobility (μ_{eff})[1] as reported in a recent study [6]. The effect of voltage variation is directly applied by changing the supply voltage; finally, the effect of temperature variation is incorporated by using temperature dependent transistor models of threshold voltage, mobility and saturation velocity using BSIM4 [10]. Experimental results verify that the proposed modeling technique is accurate (worst-case deviation of 2.64%) and leads to significant speedup (on average 53 times) in critical resistance calculation when compared with HSPICE. To the best of our knowledge, this is the first modeling technique for resistive bridge defects, that incorporates the influence of PVT variation without using HSPICE.

This paper is organized as follows. The proposed variation-aware bridge defect modeling technique is discussed in Section II. Experimental setup and results are reported in Section III, and finally Section IV concludes this paper.

II. Proposed Variation-Aware Bridge Fault Modeling Technique

The modeling technique consists of three stages. The first stage involves calculating the logic threshold voltage (L_{th}) of driven gates (Fig. 1, S_1, S_2, and S_3) of a bridge fault-site. Logic threshold voltage is defined as the gate input voltage at which the gate output voltage is equal to $\frac{V_{dd}}{2}$, while all other inputs of the gate are at non-controlling value(s) [11]. This calculation is necessary, since L_{th} is needed for critical resistance calculation of a given fault-site and is calculated using the BSIM4 transistor model. The second stage of the proposed technique computes the voltages on the bridged lines (Fig. 1, V_0, V_1) using the BSIM4 transistor model. Employing the BSIM4 transistor model through HSPICE incurs high simulation time, which is reduced by using efficient approximation algorithms (Sections II-A, II-B) for logic threshold calculation and voltages on the bridged lines. The proposed technique is faster than HSPICE because of the following two reasons. First, critical resistance calculation through HSPICE requires sweeping the resistance range from 0 Ω to (typically) 20 000 Ω [12] to observe the point where faulty value changes to fault-free value (R_{crit}). This requires two hundred HSPICE DC simulations, assuming a DC-sweep step size of 100 Ω [9], [12]. The proposed technique uses the logic threshold voltages of the driven gates and calculates R_{crit} at these specific voltage points, thereby reducing the number of iterations for calculating R_{crit}. Second, in a DC sweep, HSPICE initializes and calculates about 250 more variables than actually needed for calculating R_{crit}, the proposed technique achieves the speed up by calculating only the necessary variables for calculating R_{crit} thereby achieving speedup without compromising accuracy. The proposed technique is as accurate as HSPICE because it also uses the BSIM4 transistor model as used in HSPICE for critical resistance calculation. Through these approximation algorithms (Sections II-A, II-B), high accuracy is achieved at low computation cost as discussed in Section III. The voltage approximation algorithm (stage-2 of the proposed technique) is an improvement over the one described in [8]. These two stages therefore determine the values of all parameters needed to determine the logic behavior of a bridge fault-site in nominal operating conditions. It is recently demonstrated that bridge fault is negatively affected by process variation [7]. The final stage of the proposed technique incorporates the effect of PVT variation on the logic behavior of a bridge fault-site.

The logic threshold voltage of a fanout gate and the voltages on bridged lines of a bridge fault-site are calculated by using the BSIM4 transistor model, which accurately relates different electrical parameters with transistor device structure and takes into account various inter-dependencies between different transistor parameters. For example, scaling of the gate length results in reducing V_{th}, while increasing subthreshold swing and drain induced barrier lowering (DIBL). It is therefore well-suited to model the effect of process, voltage and temperature variations [10]. The following equation models the transistor drain current of CMOS transistor:

$$I_{ds} = \frac{I_{ds0} \cdot NF}{1 + \frac{R_{ds} \cdot I_{ds0}}{V_{dseff}}} \left[1 + \frac{1}{C_{clm}} ln \left(\frac{V_A}{V_{Asat}}\right)\right] \cdot \left(1 + \frac{V_{ds} - V_{dseff}}{V_{ADIBL}}\right) \cdot \left(1 + \frac{V_{ds} - V_{dseff}}{V_{ADITS}}\right) \cdot \left(1 + \frac{V_{ds} - V_{dseff}}{V_{ASCBE}}\right) \tag{1}$$

where I_{ds} is the drain current equation for both linear and saturation regions, I_{ds0} is the drain current valid from the subthreshold to the strong inversion regime, NF is the number of device fingers, R_{ds} is the source/drain resistance, V_{ds} is the source/drain voltage, V_{dseff} is the effective V_{ds}, C_{clm} is the channel length modulation, V_A is the early voltage, V_{Asat} is the early voltage at $V_{ds} = V_{dsat}$, V_{ADIBL} is the early voltage due to DIBL, V_{ADITS} is the early voltage due to drain induced threshold shift (DITS), and V_{ASCBE} is the early voltage due to substrate current induced body effect (SCBE). The above equation is solved by using the device parameters (per transistor) through a transistor model card [13] and other parameters such as I_{ds0} are obtained from the BSIM4 transistor model equations [10]. Leakage current including gate tunneling current, sub threshold channel current (calculated

[1]Mobility varies due to variation in effective strain in a strained silicon process [6].

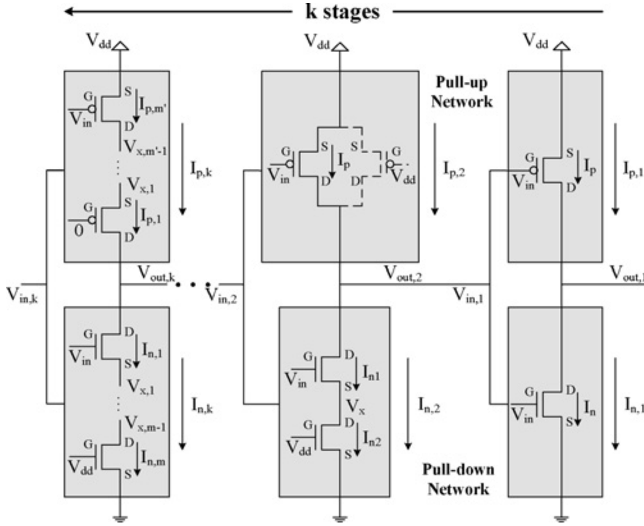

Fig. 2. General framework for logic threshold voltage calculation.

as part of I_{ds0}) and gate-induced-drain-leakage current is also included by using their respective current models from BSIM4. Note that body effect is incorporated in the BSIM4 transistor threshold voltage model, which is used in the I_{ds} equation. It has been validated by comparing the results with HSPICE using the 65 nm PTM model card (Section III).

A. Logic Threshold Voltage Calculation Algorithm

Logic threshold voltage (L_{th}) of a gate can be calculated using HSPICE, however that is a time consuming process and negatively affects the computation time of critical resistance calculation. Using 350 fault-sites, each with up to five driven gates per bridged net, it was shown in [8] that the improvement in critical resistance calculation time reduced to only 10% when using HSPICE for L_{th} generation in comparison to seven-times improvement with a pre-computed L_{th} database. This motivates the need for an efficient logic threshold generation algorithm.

Fig. 3 shows the algorithm for calculating logic threshold voltage of a gate, which is applicable to both simple and compound gates (such as OR4, AND4, and AO22). Logic gates (simple or compound) can be divided into a number of internal stages, where each stage can be sub-divided into pull-up and pull-down networks. This is illustrated in Fig. 2 that shows a compound gate with k stages ($1 \leq i \leq k$) and each stage has a pull-up and pull-down network. When calculating the logic threshold voltage of such a gate, we start with the stage connected to the gate output ($V_{\mathrm{out},1}$) and calculate its individual logic threshold voltage. For the first stage, the voltages across the pull-up and pull-down networks is set to $\frac{V_{dd}}{2}$ [11]. The logic threshold of this stage acts as the output voltage of the previous stage (Fig. 2) $V_{\mathrm{out},2} = V_{\mathrm{in},1}$. This voltage ($V_{\mathrm{out},2}$) is used to calculate the logic threshold voltage of the second stage and this calculation continues until the logic threshold voltage of the last stage ($V_{\mathrm{in},k}$) of the gate is determined.

Logic threshold of each stage is calculated by approximating the input voltage ($V_{\mathrm{in},i}$) at which the currents through the pull-up (I_p) and pull-down (I_n) networks are equal, where output

Input: V_{dd}
Output: V_{in}
1: Read the PTM model card.
  // *Model card is needed for parameters in Eq.* (1)
2: $LIMIT = 1\ \mu$A
3: Divide gate structure into k ($k \geq 1$) internal stages.
  // *Each stage has a pull-up and pull-down network.*
4: Each network is converted in to m ($m \leq 3$) transistors in series.
5: **for** $i = 1 \rightarrow k$ **do**
6:   **if** $i = 1$ **then**
7:     $V_{out,i} = V_{out} = V_{dd}/2$
8:   **else**
9:     $V_{out,i} = V_{in,(i-1)}$
10:   **end if**
11:   $V_{Max} = V_{dd}$, $V_{Min} = 0$
12:   $V_{tmp} = (V_{Max} + V_{Min})/2$
  // *This loop is used to get* $V_{in,i}$
13:   **repeat**
14:     $V_{in,i} = V_{tmp}$
15:     $I_{n,i}$ can be calculated by using the algorithm shown in Fig. 4 with input of $V_{in,i}$ and $V_{out,i}$
16:     $I_{p,i}$ can be calculated by modifying the algorithm shown in Fig. 4 with input of $V_{in,i}$ and $V_{out,i}$
17:     **if** $I_{n,i} \geq I_{p,i}$ **then**
18:       $V_{Max} = V_{tmp}$
19:     **else**
20:       $V_{Min} = V_{tmp}$
21:     **end if**
22:     $V_{tmp} = (V_{Max} + V_{Min})/2$
23:   **until** $(|I_{n,i} - I_{p,i}| \geq LIMIT)$
24: **end for**
25: **return** $V_{in} = V_{in,k}$

Fig. 3. Logic threshold voltage algorithm.

voltage ($V_{\mathrm{out},i}$) is known. The algorithm (Fig. 3) first converts a (pull-up or pull-down) network into a number of series connected transistors by calculating the effective $\frac{W}{L}$ of parallel connected transistors ($\frac{W}{L} = \frac{W_1}{L_1} + \cdots + \frac{W_n}{L_n}$). The algorithm (Fig. 3) approximates the logic threshold voltage between the two variables, V_{Max} and V_{Min}. It first assigns V_{dd} to V_{Max} and 0 V to V_{Min}, with each iteration it reduces the separation between V_{Max} and V_{Min} by half. V_{in} is set to the midpoint between V_{Max} and V_{Min} and by comparing the currents through the pull-up and pull-down networks (calculated through the algorithm shown in Fig. 4), each iteration reduces the separation between V_{Max} and V_{Min} either by reducing V_{Max} or increasing V_{Min} (step-17 to step-21 in Fig. 3). These steps are based on the principle that for $V_{\mathrm{in}} \geq L_{\mathrm{th}}$, $I_n \geq I_p$ and at $V_{\mathrm{in}} < L_{\mathrm{th}}$, $I_n < I_p$. The algorithm terminates when the difference in I_n and I_p is smaller than $LIMIT$. In this paper, $LIMIT$ is set to $1\ \mu$A, which was found empirically to achieve high accuracy. The algorithm converges using small (on average 15 or less) number of iterations for all reported results in Section III. The current through the series connected transistors in pull-up and pull-down networks is calculated by using the algorithm (Fig. 4), which can be used for up to three transistors in series ($m \leq 3$, where m is the number of transistors in series).

Input: V_{in} and V_{out}
Output: I_n
1: Read the PTM model card.
   // *Model card is needed for parameters in Eq.* (1)
2: $LIMIT = 1\ \mu A$
3: **if** Single NMOS **then**
4: $\quad I_n = I_{ds,n}(V_{ds} = V_{out},\ V_{gs} = V_{in})$
   $\quad$ // $I_{ds,n}$ *is calculated by using Eq.* (1)
5: **else if** Two NMOS transistors in series **then**
6: $\quad V_{Max} = V_{out},\ V_{Min} = 0$
7: $\quad V_x = (V_{Max} + V_{Min})/2$
8: $\quad$ **repeat**
9: $\qquad I_{n,1} = I_{ds,n}(V_{ds} = V_{out} - V_x,\ V_{gs} = V_{in} - V_x)$
10: $\qquad I_{n,2} = I_{ds,n}(V_{ds} = V_x,\ V_{gs} = V_{dd})$
11: $\qquad$ **if** $I_{n,1} \geq I_{n,2}$ **then**
12: $\qquad\quad V_{Max} = V_x$
13: $\qquad$ **else**
14: $\qquad\quad V_{Min} = V_x$
15: $\qquad$ **end if**
16: $\qquad V_x = (V_{Max} + V_{Min})/2$
17: $\quad$ **until** $(|I_{n,1} - I_{n,2}| \geq LIMIT)$
18: $\quad I_n = (I_{n,1} + I_{n,2})/2$
19: **else if** Three NMOS transistors in series **then**
20: $\quad V_{Max,1} = V_{out},\ V_{Min,1} = 0$
21: $\quad V_{x,1} = (V_{Max,1} + V_{Min,1})/2$
22: $\quad$ **repeat**
23: $\qquad V_{Max,2} = V_{x,1},\ V_{Min,2} = 0$
24: $\qquad V_{x,2} = (V_{Max,2} + V_{Min,2})/2$
25: $\qquad$ **repeat**
26: $\qquad\quad I_{n,2} = I_{ds,n}(V_{ds} = V_{x,1} - V_{x,2},\ V_{gs} = V_{dd} - V_{x,2})$
27: $\qquad\quad I_{n,3} = I_{ds,n}(V_{ds} = V_{x,2},\ V_{gs} = V_{dd})$
28: $\qquad\quad$ **if** $I_{n,2} \geq I_{n,3}$ **then**
29: $\qquad\qquad V_{Max,1} = V_{x,1}$
30: $\qquad\quad$ **else**
31: $\qquad\qquad V_{Min,1} = V_{x,1}$
32: $\qquad\quad$ **end if**
33: $\qquad\quad V_{x,1} = (V_{Max,1} + V_{Min,1})/2$
34: $\qquad$ **until** $(|I_{n,2} - I_{n,3}| \geq LIMIT)$
35: $\qquad I_{tmp} = (I_{n,2} + I_{n,3})/2$
36: $\qquad I_{n,1} = I_{ds,n}(V_{ds} = V_{out} - V_{x,1},\ V_{gs} = V_{in} - V_{x,1})$
37: $\qquad$ **if** $I_{n,1} \geq I_{tmp}$ **then**
38: $\qquad\quad V_{Max,2} = V_{x,2}$
39: $\qquad$ **else**
40: $\qquad\quad V_{Min,2} = V_{x,2}$
41: $\qquad$ **end if**
42: $\qquad V_{x,2} = (V_{Max,2} + V_{Min,2})/2$
43: $\quad$ **until** $(|I_{n,1} - I_{tmp}| \geq LIMIT)$
44: $\quad I_n = (I_{n,1} + I_{n,2} + I_{n,3})/3$
45: **end if**
46: **return** I_n

Fig. 4. Logic threshold voltage approximation algorithm for NMOS transistors in series.

Next, we explain how to approximate the current through the pull-down network (I_n), for the second stage of a gate shown in Fig. 2, with two NMOS transistors in series ($m = 2$). The steps (Step-5 to Step-18) are shown in Fig. 4. I_n is calculated by approximating the value of V_x across series connected transistors and it is used to calculate the currents through each of the two transistors in series (I_{n1} and I_{n2}). The algorithm first assigns the stage output voltage (V_{out}) to V_{Max} and 0 V to V_{Min}. It then assigns V_x the mid-point voltage of V_{Max} and V_{Min}. Through this value of V_x, it calculates the currents through each of the two transistors I_{n1} and I_{n2}, using (1). It then compares I_{n1} and I_{n2}, and reduces the separation between V_{Max} and V_{Min} until the difference between the currents I_{n1} and I_{n2} is less than $LIMIT$ ($1\,\mu$A). Once the current through the pull-down network is calculated, it is compared with I_p (current through the pull-up network of the same stage), which is calculated using an algorithm for PMOS transistors (similar to Fig. 4), where I_p will be calculated by assigning ($V_{\text{out}} - V_{dd}$) to V_{ds} and ($V_{\text{in}} - V_{dd}$) to V_{gs} (note as shown in Fig. 2, the pull-up network of second stage has just one transistor switched on). The currents through the pull-up and pull-down networks are used to adjust the logic threshold voltage of each stage $V_{\text{in},i}$ (step-17 to step-21 in Fig. 3), until the difference is less than $LIMIT$ and at that point the algorithm returns the logic threshold voltage of the given gate.

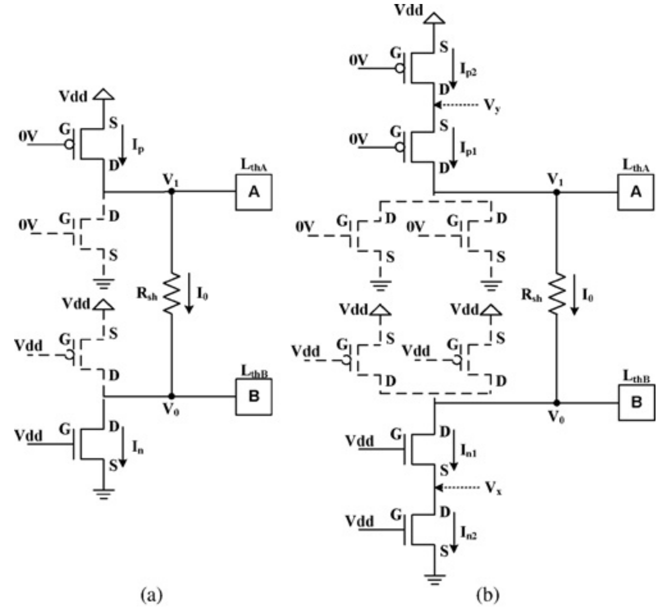

Fig. 5. Bridge resistance examples. (a) Fault-site driven by two inverters. (b) Fault-site driven by 2-input NOR and 2-input NAND.

B. Bridge Critical Resistance Calculation Algorithm

The critical resistance of a bridge is calculated through the BSIM4 transistor model and voltage approximation algorithm (Fig. 6). The algorithm approximates the voltage on bridged lines (Fig. 1, V_0, V_1), while considering both NMOS and PMOS transistors in gates driving the bridge (Fig. 1, D_1, D_2). As an example Fig. 5(a) shows a fault-site, where two inverters are driving a bridge (R_{sh}) and I_0 is the current through the resistor. The transistors drawn using dashed lines represent switched off transistors. The value of R_{sh} is

$$R_{\text{sh}} = \frac{(V_1 - V_0)}{I_0}. \tag{2}$$

Using the logic threshold voltage (L_{thA}, obtained through the algorithm discussed in Section II-A) of the driven gate "A" [Fig. 5(a)], V_1 is L_{thA}, which can be used to calculate I_0

```
Input: I_0
Output: V_0 or V_1
 1: Read the PTM model card.
    // Model card is needed for parameters in Eq. (1)
 2: V_tmp = 0
 3: I_tmp = 0, LIMIT = 0.005
 4: if NMOS then
 5:    V_Max = V_dd, V_Min = 0
 6:    repeat
 7:       V_tmp = (V_Max + V_Min) / 2
 8:       I_tmp = I_n(V_tmp); with V_bs = 0 and V_gs = V_dd
          // I_tmp is calculated by using Eq. (1)
 9:       if I_tmp >= I_0 then
10:          V_Max = V_tmp
11:       else
12:          V_Min = V_tmp
13:       end if
14:    until [ |I_0 - I_tmp| / I_0 >= LIMIT ]
15:    return (V_tmp)
       // V_0 = V_tmp
16: else
17:    V_Max = 0, V_Min = -V_dd
18:    repeat
19:       V_tmp = (V_Max + V_Min) / 2
20:       I_tmp = I_p(V_tmp); with V_bs = 0 and V_gs = -V_dd
          // I_tmp is calculated by using Eq. (1)
21:       if I_tmp >= I_0 then
22:          V_Min = V_tmp
23:       else
24:          V_Max = V_tmp
25:       end if
26:    until [ |I_0 - I_tmp| / I_0 >= LIMIT ]
27:    return (V_dd + V_tmp)
       // V_1 = V_dd + V_tmp
28: end if
```

Fig. 6. Approximation algorithm for calculating voltage on bridged lines.

through the I–V_{ds} relationship shown in (1), i.e., $I_0 = I_p(V_{\mathrm{ds},p})$, where $V_{\mathrm{ds},p} = L_{\mathrm{thA}} - V_{dd}$. The only unknown variable left in (2) is V_0, which can be approximated by using the algorithm shown in Fig. 6 [since $I_0 = I_n(V_0)$, which implies $V_0 = I_n^{-1}(I_0)$]. The algorithm shown in Fig. 6 assigns V_{dd} to V_{Max} and 0 V to V_{Min}, respectively. It then assigns the mid-point voltage value (between V_{Max} and V_{Min}) to V_0. This is used to generate I_n (represented by I_{tmp}). If $I_n \geq I_0$, that means the value of V_0 is between V_{tmp} and 0 V; otherwise it is between V_{dd} and V_{tmp} (as ideally, $I_n = I_p = I_0$). This process is repeated until the relative difference between I_0 and I_n is smaller than the specified limit, as determined by step-14 of the algorithm. In our experiments, $LIMIT$ is 0.005, as it was determined empirically that this value provides high accuracy when compared with HSPICE. The algorithm converges quickly and requires only small number (15 or less) of iterations. The value of V_0 is then used together with the other two variables (V_1, I_0) to calculate R_{sh} using (2). The same procedure can be repeated for PMOS transistor, starting with the value of V_0 as the logic threshold of gate "B," i.e., L_{thB}. In case of transistors in parallel, the effective $\frac{W}{L}$ is calculated before starting the algorithm. For transistors in series [Fig. 5(b)], the algorithm shown in our earlier publication [8, Fig. 8] is used, which is not shown due to space limitation. It should be noted that voltage (V_{ds}) approximation algorithm (Fig. 6) is an improvement over the one presented in [8] and is on average 41-times faster, while achieving the same accuracy.

TABLE I
VARIED PROCESS PARAMETERS

Parameter	Mean (μ)	Std. Deviation (σ)
L	60 nm	±4% (2.4 nm)
V_{thn}	0.423 V	±5% (21.15 mV)
V_{thp}	−0.365 V	±5% (18.25 mV)
μ_{effn}	491 cm^2/V·s	±21% (103.1 cm^2/V·s)
μ_{effp}	57.4 cm^2/V·s	±21% (12.05 cm^2/V·s)

C. Incorporation of PVT Variation

The first two stages of the proposed modeling technique (Sections II-A, II-B) are used to calculate critical resistance of a bridge fault-site in nominal operating conditions. Next, we explain how the effect of PVT variation is incorporated in the proposed modeling technique. The variation effects are incorporated in transistor model, for example (1), and therefore affect logic threshold voltage of the driven gates and voltages on bridged lines, leading to change in logic fault behavior of the bridge fault-site.

A recent study describes the parameter extraction technique (for process variation) using a 65 nm CMOS library with a PTM model [6], [13]. Three transistor parameters are recognized as the leading sources of process variation, which include gate length (L), threshold voltage (V_{th}), and mobility (μ_{eff}). These parameters follow Gaussian distribution ($\pm 3\sigma$ variation) with standard deviations of 4% for L, 5% for V_{th}, and 21% for μ_{eff}. Negligible spatial correlation is found in between these parameters, i.e., they can be treated as independent random variables following Gaussian distribution. These results are validated by comparing with the measured data using a fabricated device. Note the parameter fluctuations (correlated or otherwise) do not imply that these parameters are independent, for example as L decreases, V_{th} also decreases, this effect is also known as V_{th} roll-off [10]. Our experiments are based on a ST Microelectronics 65 nm gate library using the same PTM model cards that are used in [6], which is why we have also assumed the same parameter fluctuations. The mean and standard deviation for both NMOS/PMOS transistors are shown in Table I. Recent research has shown that it is sufficient to consider $\pm 3\sigma$ variation of process parameters, when modeling process variation for logical part of the design [7], [14], and higher variation effects ($\pm 6\sigma$ or more) are considered for (SRAM and Flash) memories [5]. This paper also deals with the logical part of the design, which is why we have also considered $\pm 3\sigma$ variation effects.

Within-die variation is modeled by varying only the gate length of different transistors using a spatial correlation

model [15]. As pointed out by several publications, gate length is a leading source of process variation and it has shown correlated variation effects due to lithography [15]–[17]. Experimental results in [8] show that 3σ variation of L, V_{th}, and μ_{eff} has a much wider spread of critical resistance than the gate length spatial correlation. That means the effect of spatial correlation is covered by considering variations due to three un-correlated parameters (L, V_{th}, and μ_{eff}). From test generation point of view, it means that considering three un-correlated parameter variations are likely to cover the complete logic fault domain due to within-die spatially correlated parameter variation (see [8] for more details).

Temperature variation is incorporated in (1) by using temperature dependent models of threshold voltage, mobility, and saturation velocity, as described in the BSIM4 transistor model [10]. The temperature dependent threshold voltage model is given by

$$V_{\text{th}}(T) = V_{\text{th}}(TNOM) + \left(KT1 + \frac{KT1L}{L_{\text{eff}}} + KT2 \cdot V_{\text{bseff}}\right) \cdot \left(\frac{T}{TNOM} - 1\right) \quad (3)$$

where $V_{\text{th}}(T)$ is the temperature dependence of threshold voltage, $V_{\text{th}}(TNOM)$ is the nominal transistor threshold voltage, T is the circuit temperature, $TNOM$ is the transistor model reference temperature and its nominal value is 25 °C, $KT1$ is the temperature coefficient for threshold voltage, $KT1L$ is the channel length dependence of the temperature coefficient for threshold voltage, $KT2$ is the body-bias coefficient of threshold voltage temperature effect, and V_{bseff} is the effective body bias voltage. The temperature dependent mobility model is given by

$$U0\,(T) = U0\,(TNOM) \cdot (T/TNOM)^{UTE} \quad (4)$$

where $U0(T)$ is the temperature dependence of mobility, $U0(TNOM)$ is the nominal transistor mobility, UTE is the mobility temperature exponent. The temperature dependent model of saturation velocity is given by

$$VSAT\,(T) = VSAT\,(TNOM) - AT \cdot (T/TNOM - 1) \quad (5)$$

where $VSAT(T)$ is the temperature dependence of saturation velocity, $VSAT(TNOM)$ is the nominal transistor saturation velocity, AT is the temperature coefficient for saturation velocity. The temperature dependent models mainly rely on the ratio of circuit temperature (T) to model reference temperature ($TNOM$), for example as shown in (3). When $T \neq TNOM$, additional values are calculated according to these equations and are used for calculating transistor drain current using (1).

As discussed at the beginning of this section, for a given fault-site, transistor drain current of the driving gate and logic threshold voltage of the driven gates are the two important parameters for calculating the bridge critical resistance. We analyzed the effect of temperature variation on critical resistance calculation by simulating the change in transistor drain current and logic threshold voltage with the change in temperature. Fig. 7 shows the effect of temperature variation on drain current of 65 nm NMOS transistor, while keeping $V_{\text{ds}} = V_{dd} = 1.2$ V and increasing V_{gs} from 0 V to V_{dd}. As can be seen, at lower values of $V_{gs} \leq 0.45$ V, current increases with temperature, however this trend reverses at higher values of V_{gs} and current reduces with increase in temperature. The crossing point (marked in Fig. 7) is also called zero temperature coefficient and its effect is examined in several publications (see [18] for more details). Similarly, when considering logic threshold voltage of a gate, it was found that it also reduces as temperature increases. For 65 nm ST Microelectronics gate library, average reduction in logic threshold voltage is about 75 mV, when the temperature increases from −40 °C to 125 °C, operating at 1.2 V V_{dd}. Temperature also affects metal resistance as it increases with temperature. To analyze the effect of temperature variation on detectable resistance range, we conducted an experiment at 0.8 V V_{dd} using 350 fault-sites at −40 °C and 125 °C. It was found that about 86% fault-sites show higher detectability at 125 °C and about 14% fault-sites show better detectability at −40 °C. This means that at higher temperatures, reduction in transistor current of driving gates and logic threshold voltages of the driven gates increases the detectable resistance range of majority of fault-sites. We also analyzed the behavior of transistor drain current using 45 nm and 32 nm NMOS transistors (PTM model cards) and found similar behavior (Fig. 7), which means that this trend of detectable resistance range will continue for these technologies as well.

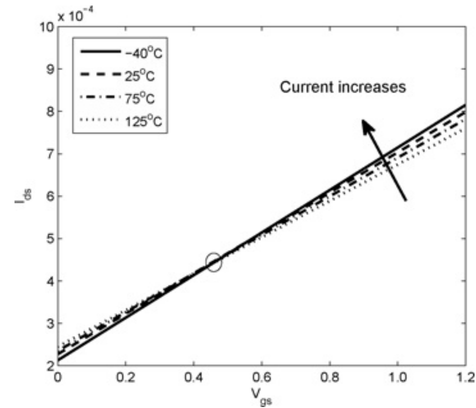

Fig. 7. Temperature dependence of NMOS transistor drain current (I_{ds}) in 65 nm technology.

The variation in supply voltage is modeled by (1) in a straightforward manner because V_{ds} and V_{gs} change with supply voltage. Fig. 8 shows the drain current under different supply voltages and temperatures using the proposed model and HSPICE using 65 nm technology. The temperature varies from −40 °C to 125 °C and the voltage varies from 0.8 V to 1.2 V, which are the operating temperatures and voltages for 65 nm ST Microelectronics gate library. As can be seen, it shows excellent correlation with HSPICE results.

III. Experimental Results

All experiments are conducted using a 65 nm ST Microelectronics gate library and PTM transistor model card [13]

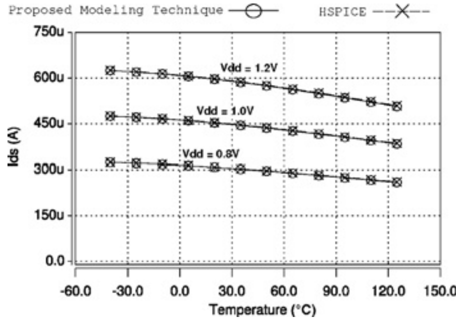

Fig. 8. Drain current under different supply voltages and temperatures using proposed model and HSPICE.

on Intel Xeon Quad Core 2.7 GHz processor with 12 GB RAM. The gate library consists of a variety of gates including simple (NAND, NOR, INV) and compound gates (AO22, OA22, and so on), each with different drive strengths. For illustration purposes 1.2 V and 25 °C are used as the nominal operating voltage and temperature in all experiments. The proposed modeling technique is based on the BSIM4 transistor model that provides detailed sets of equations for calculating each transistor parameter. The input value to each equation is provided by the PTM transistor model card and the gate library. The proposed model is compared with HSPICE and to avoid any discrepancies, we used the same gate library and transistor model card with both techniques (HSPICE and proposed). HSPICE also uses the BSIM4 transistor model as noted on the HSPICE data sheet [19]. The flow of the proposed modeling technique is shown in Fig. 9. The flow inputs are gate library and respective transistor models and the output is logic fault values of the bridge fault-site in the presence of process, voltage, and temperature variation. The flow has five main blocks as shown in Fig. 9. The bridge fault-site is generated by randomly selecting (driving and driven) gates from the gate library, using n driven gates per fanout, where $n \in [1, 5]$ and only non-feedback bridges are generated by the bridge fault-site generator. Each of the driving gates is assigned a random input, while ensuring that the two nets are driven at opposite logic values to activate the bridge fault. This setup uses 350 fault-sites for each experiment because it was shown in [7] that the average number of fault-sites per design is less than 300 with coupling capacitance based layout extraction of bridges using ISCAS'85, 89 benchmarks. The effect of process variation is incorporated by the process variation permutation generator. It varies three parameters (L, V_{th}, and μ_{eff}) using Gaussian distribution with mean and standard deviation as shown in Table I. In total, 600 permutations per fault-site are generated through Monte Carlo simulation. The number of permutations are based on a recent study, which shows that the probability of generating a unique logic fault follows the law of diminishing returns, as it reduces significantly after 500 permutations [9]. The effect of voltage variation is incorporated by varying supply voltages from 0.8 V to 1.2 V with the step size of 0.1 V using the voltage variation generator. The temperature variation generator generates three temperature values −40 °C, 25 °C, and 125 °C, which are the minimum, nominal, and maximum working temperatures for

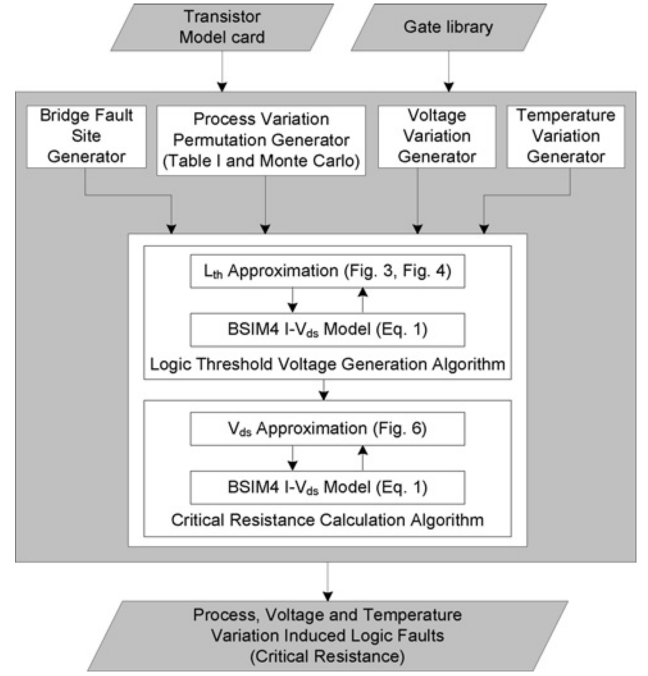

Fig. 9. Proposed PVT variation-aware bridge defect modeling flow.

65 nm ST gate library. These voltage and temperature values are used for demonstration purposes and the same flow can also be used for other values just as well. The outputs of these four blocks are fed to the main block which includes logic threshold voltage generation algorithm and critical resistance calculation algorithm. The logic threshold voltage generator uses the BSIM4 transistor drain current model (1) and the L_{th} approximation algorithm (Section II-A) to generate logic threshold voltages of a given fault-site, while including the effect of PVT variation. The outputs of these voltage values are fed to the critical resistance calculator. The critical resistance calculator also uses (1) and voltage approximation algorithm (Section II-B) to generate all PVT variation induced logic faults of bridge fault-site.

This flow is used to conduct three experiments for validating the proposed modeling technique by comparing the results with HSPICE. The first experiment (Section III-A) validates the logic threshold voltage generation algorithm. The second experiment (Section III-B) validates the critical resistance calculation algorithm. These two algorithms are separately validated to determine the loss of accuracy due to each approximation algorithm. Finally, the last experiment (Section III-C) validates the complete modeling technique (Fig. 9) including the two approximation algorithms under the influence of PVT variation.

A. Validation of Logic Threshold Voltage Calculation Algorithm

The logic threshold voltage generation algorithm (Section II-A) is validated by comparing the results with HSPICE in nominal operating conditions (1.2 V, 25 °C) and in the presence of PVT variation. When operating in nominal condition, the comparison (accuracy and speed) of various gates

TABLE II
LOGIC THRESHOLD VOLTAGE GENERATOR IN COMPARISON WITH HSPICE IN NOMINAL OPERATING CONDITIONS

Gate	Node	Lvth (V)			Time
		LG	HSP	Err (%)	$\frac{HSP}{LG}$
INV		0.5393	0.54	0.1	2730
NAND3	A	0.566	0.5738	1.4	5
	B	0.5976	0.6108	2.2	5
	C	0.6222	0.6332	1.7	6
NAND4	A	0.5648	0.5511	2.5	84
	B	0.6216	0.6417	3.1	251
	C	0.5668	0.5514	2.8	196
	D	0.6268	0.6465	3.0	135
NOR3	A	0.5535	0.5441	1.7	14
	B	0.5343	0.5225	2.3	9
	C	0.5121	0.5045	1.5	6
NOR4	A	0.5343	0.5493	2.7	198
	B	0.4869	0.4726	3.0	288
	C	0.5414	0.5495	1.5	188
	D	0.5201	0.5345	2.7	251
AND3	A	0.5801	0.5708	1.6	8
	B	0.6238	0.6403	2.6	7
	C	0.6375	0.6513	2.1	16
AND4	A	0.5695	0.5652	0.8	215
	B	0.6141	0.6318	2.8	133
	C	0.5777	0.5677	1.8	138
	D	0.6217	0.6373	2.4	165
OR3	A	0.5653	0.5534	2.2	8
	B	0.546	0.5457	0.1	9
	C	0.5261	0.5327	1.2	8
OR4	A	0.5414	0.5428	0.3	167
	B	0.5121	0.523	2.1	142
	C	0.5343	0.5426	1.5	161
	D	0.505	0.5038	0.2	167
Avg. Speedup					197

*LG → proposed logic threshold voltage generation algorithm.

TABLE III
ERROR (%) IN LOGIC THRESHOLD VOLTAGE GENERATION UNDER THE INFLUENCE OF PROCESS VARIATION IN COMPARISON TO HSPICE

Gate	Node	L_{th} Error (%)		
		Min	Max	Avg
INV		0.01	0.58	0.14
NAND3	A	0.26	3.54	1.88
	B	0.59	4.97	2.27
	C	0.64	4.66	2.11
NAND4	A	0.01	5.18	2.60
	B	0.66	5.57	4.17
	C	0.02	5.15	3.74
	D	0.20	5.19	4.31
NOR3	A	0.57	4.06	2.89
	B	0.47	3.45	2.37
	C	0.50	3.76	1.81
NOR4	A	0.01	5.74	3.88
	B	0.63	5.69	4.58
	C	0.14	5.19	4.00
	D	0.04	5.21	4.39
AND3	A	0.05	4.84	2.36
	B	0.04	4.89	2.78
	C	0.59	4.80	2.52
AND4	A	0.47	4.61	2.90
	B	0.19	5.06	3.49
	C	0.02	4.30	2.11
	D	0.10	4.26	2.88
OR3	A	0.22	5.21	3.25
	B	0.46	3.79	2.43
	C	0.08	4.25	1.90
OR4	A	0.04	5.52	2.06
	B	0.40	5.36	3.45
	C	0.02	4.73	1.90
	D	0.07	5.22	2.54

is shown in Table II. Due to space limitations, only gates expected to cause higher approximation error are shown; higher error is expected in gates with transistors in series and compound gates. It can be seen that using the proposed logic threshold calculation algorithm, the error varies from 0.1% (INV "Inverter," 0.7 mV error) to 3.1% (input-B of NAND4 "4-input NAND gate," 20.1 mV error) when compared with HSPICE. Error is highest in case of NAND4 gate (four-input NAND) as it consists of 2 two-input NAND gates connected to a two-input NOR gate followed by an INV, therefore the error in logic threshold calculation is accumulated with each stage. We also investigated the effect of leakage current, contributed by transistors that are switched off. For all the gates shown in Table II, error due to leakage current is less than 2.1 mV (0.44%) and on average it is 0.7 mV (0.13%). To analyze further, the deviation of calculated logic threshold voltages from HSPICE results, we also investigated the impact of body effect due to transistor stacking on threshold voltage, using 20 different transistor configurations. For all configurations, using the proposed model, the difference is less than 0.98 mV (0.42%) with average difference of 0.86 mV (0.31%) in comparison to HSPICE. The last column of Table II shows the relative runtime improvement when comparing the proposed logic threshold calculation algorithm and HSPICE ($\frac{HSP}{LG}$). In comparison to HSPICE, the maximum improvement is in case of INV (2730 times), and least improvement is in case of NAND3 gate (3-input NAND), which is five-times. This is because NAND3 gate has three transistors in series in the pull-down network and I_n approximation for each transistor is needed to compute logic threshold voltage of each gate input. When considering all the gates shown in Table II, average improvement is 197-times in comparison to HSPICE and in general considering all gates in the gate library, it was found that average time improvement is 257-times.

Next, we examine the influence of process variation on the accuracy of the proposed logic threshold voltage calculation algorithm. This experiment uses process variation permutation generator (Fig. 9) and the generated results are compared with HSPICE to examine the relative accuracy. The results are shown in Table III with minimum (Min), maximum (Max) and average (Avg) error per gate-input per gate. The least error is observed in case of the simplest gate (INV) with 0.14% average error over 600 permutations. Highest deviation of 4.58% is observed in case of NOR4 gate (4-input NOR), because it consists of two NOR2 gates, connected to NAND2 gate, followed by an INV and error is accumulated with each stage. In any permutation over all gates, maximum observed error is 5.69% as in case of input-B of NOR4 gate.

Finally, the logic threshold voltage generator is validated under the effect of PVT variation by using the process varia-

TABLE IV
AVERAGE ERROR (%) IN LOGIC THRESHOLD VOLTAGE GENERATION UNDER PVT VARIATION IN COMPARISON TO HSPICE

Temperature \ Voltage	0.8 V	0.9 V	1.0 V	1.1 V	1.2 V
−40 °C	2.31%	2.23%	2.18%	2.13%	2.13%
25 °C	3.11%	2.77%	2.50%	2.36%	2.23%
125 °C	3.27%	3.08%	2.96%	2.82%	2.80%

tion permutation generator, voltage and temperature variation generators (Fig. 9). The experiment is conducted by using 600 permutations of process variation at different voltage and temperature settings, when considering each gate-input per gate and average error is shown in Table IV. At a given temperature, the average error in logic threshold voltage calculation decreases as voltage increases. Similarly, at a given voltage, average error in logic threshold calculation increases as temperature increases from −40 °C to 125 °C. This is because transistor drain current reduces with supply voltage, and at a given voltage it reduces further with increase in temperature as shown in Fig. 8. Since the algorithm (Section II-A) terminates when the difference in currents through the pull-up and pull-down networks of a gate is less than 1 μA, this 1 μA difference in current becomes a bigger proportion of transistor currents at lower voltage (higher temperature) setting leading to higher accuracy error. A trivial change in termination criteria, for example reducing it further from 1 μA can improve the accuracy at lower voltage and higher temperature setting.

B. Validation of Bridge Critical Resistance Calculation Algorithm

The critical resistance calculator uses the approximation algorithm discussed in Section II-B. We first compare the generated results using the proposed critical resistance calculation algorithm with HSPICE in nominal operating condition and under the influence of PVT variation. This experiment uses the flow shown in Fig. 9, where for a given bridge fault-site, logic threshold voltage of the driven gates is calculated by using HSPICE to examine the calculation error of the proposed algorithm. Table V shows the results in nominal operating condition and when considering the influence of process variation on critical resistance calculation of bridge fault-site. The fault-sites shown in Table V include a number of cases where high approximation error is expected due to gates with transistors in series. In nominal operating condition, the difference in critical resistance varies from 0.18% (fault-site driven by INV-INV with 3 Ω difference) to 0.73% (fault-site driven by 3-input NOR and 3-input NAND gates with 11 Ω difference) when compared with HSPICE. Generally, the difference in critical resistance calculation increases with higher number of transistors in series in the pull-up and pull-down networks as in case of fault-site driven by NOR3-NAND3 gates. This is because each transistor requires voltage (V_{ds}) approximation using the algorithm discussed in Section II-B to calculate the critical resistance of the bridge. We also analyzed the error contribution due to leakage current and for all the fault-sites shown in Table V, max difference is 0.13%

TABLE V
HSPICE RESULTS IN COMPARISON WITH THE CRITICAL RESISTANCE CALCULATOR IN NOMINAL OPERATING CONDITIONS AND UNDER THE INFLUENCE OF PROCESS VARIATION

Driving Gates (D_1, D_2)	Tech.	Nom (Ω)	±3σ Variation L (Ω)	±3σ Variation H (Ω)	Time (s)	Nom Err. %
INV (0)	CRC	1658	95	7559	0.37	
INV (1)	HSP	1661	99	7574	301	0.18
IVN (0)	CRC	1223	408	9758	0.74	
NAND2 (1, 1)	HSP	1232	406	9762	307	0.73
NAND2 (0, 0)	CRC	3732	2550	14 001	0.82	
NAND2 (1, 1)	HSP	3752	2569	13 900	312	0.53
NOR2 (0, 0)	CRC	2094	62	11 998	1.20	
NAND2 (1, 1)	HSP	2104	56	12 080	319	0.47
NOR2 (0, 0)	CRC	1649	59	13 540	1.50	
NAND3 (1, 1, 1)	HSP	1655	67	13 440	317	0.36
NOR3 (0, 0, 0)	CRC	1511	76	13 157	1.86	
NAND3 (1, 1, 1)	HSP	1500	73.6	13 090	316	0.73

*CRC → proposed critical resistance calculation algorithm, HSP → HSPICE, L → low, H → high, Nom → nominal.

leading to 2 Ω difference in critical resistance calculation, as in case of a fault-site driven by NOR3-NAND3 gates. This clearly demonstrates the accuracy of the proposed critical resistance calculation algorithm in nominal operating conditions.

Table V also shows the minimum (low) and maximum (high) values of bridge critical resistance, as a result of process variation across ±3σ range. The minimum and maximum differences are 2 Ω (fault-site driven by INV-NAND2) and 101 Ω (fault-site driven by NAND2-NAND2) respectively, which is also the maximum difference observed for all 350 fault-sites. It should be noted that bridge fault is detected over a range of resistance values and a test is not for a specific (discrete) resistance, as shown in [12]. This means that the difference in resistance values (Table V) between the proposed model and HSPICE does not necessarily mean loss of test coverage. The second last column of Table V shows the simulation time using the two methods (proposed and HSPICE). The proposed method is approximately 376-times faster than HSPICE and in general, 287-times faster for 350 fault-sites. This is a significant speedup in comparison to the algorithm proposed in [8], which is on average 7-times faster than HSPICE. Note that the simulation time of logic threshold generation is excluded to examine the relative speedup, in comparison to HSPICE, using the proposed algorithm for critical resistance calculation.

The combined effect of PVT variation using the proposed bridge critical resistance calculation algorithm is considered next and the results are shown in Table VI. This shows the average error in bridge critical resistance calculation using the proposed algorithm and HSPICE when considering 350 fault-sites and it shows the same trend as observed from Table IV and related discussion (in the last paragraph of Section III-A), at a given temperature, the average error in bridge critical resistance calculation decreases as voltage increases and at a given voltage, the average error increases as temperature increases.

C. Validation of the Proposed Modeling Technique

We show experimental results to validate the complete modeling flow (Fig. 9) using the two proposed algorithms for

TABLE VI
AVERAGE ERROR (%) IN CRITICAL RESISTANCE CALCULATIONS UNDER PVT VARIATIONS IN COMPARISON TO HSPICE

Temperature \ Voltage	0.8 V	0.9 V	1.0 V	1.1 V	1.2 V
−40 °C	0.65%	0.61%	0.59%	0.57%	0.56%
25 °C	0.69%	0.69%	0.66%	0.62%	0.60%
125 °C	1.73%	1.47%	1.31%	1.19%	1.11%

logic threshold voltage generation (Section II-A) and critical resistance calculation (Section II-B) and compare the results with HSPICE to examine the net effect on accuracy using the proposed technique. In this experiment, the proposed algorithm for logic threshold (L_{th}) voltage generation is used for each fault-site and L_{th} is then used to calculate the critical resistance of each of 350 different bridge fault-sites.

We first show the results in nominal operating condition (1.2 V, 25 °C) and then incorporate the effect of process variation. The results are shown in Table VII for a selected number of fault-sites for which high approximation error is expected due to gates with transistors in series. For all fault-sites, Table VII shows the comparison in nominal operating condition (marked "Nom"), it can be seen that when compared with HSPICE, the difference in critical resistance varies from 8 Ω (bridge driven by INV-INV) to 31 Ω (bridge driven by NOR3-NAND3) leading to calculation error of 0.48% to 2.07%, respectively. High error (in case of NOR3-NAND3) is because three transistors are in series in the pull-up and pull-down networks of the two gates driving the bridge, and each transistor requires voltage (V_{ds}) approximation using the algorithm proposed in Section II-B, for bridge critical resistance calculation. Error due to leakage current for all the fault-sites shown in Table VII was also analyzed and max difference is 0.2% leading to 3 Ω difference in critical resistance calculation, as in case of a fault-site driven by NOR3-NAND3 gates.

When considering the effect of process variation, Table VII shows the minimum (low) and maximum (high) values of bridge critical resistance (R_{crit}). The minimum and maximum differences are 3 Ω (fault-site driven by INV-NAND2) and 132 Ω (fault-site driven by NOR3-NAND3) respectively, which is also the maximum difference observed for all 350 fault-sites. Fig. 10 shows the effect of process variation on critical resistance calculation of a bridge fault-site driven by two NAND2 gates (with inputs [0, 0] and [1, 1]). It can be observed from these results (Table VII, Fig. 10) that the proposed modeling technique achieves high accuracy (worst case error of 2.07%) when compared with HSPICE. When considering the combined effect of PVT variation over all (350) fault-sites, the worst case error is 2.64% when operating at 0.8 V and 125 °C.

From the results presented in this section, we observe that the following two observations can further improve the proposed modeling technique. First, since the proposed technique utilizes BSIM4, the upper bound in accuracy is that of the BSIM4 transistor model. However, this technique (Section II) does not depend on a specific transistor model and can be updated using a more accurate transistor model to achieve higher accuracy. Second, it is observed from the experimental results

TABLE VII
HSPICE RESULTS IN COMPARISON WITH THE PROPOSED TECHNIQUE IN NOMINAL OPERATING CONDITIONS AND UNDER THE INFLUENCE OF PROCESS VARIATION

Driving Gates (D_1, D_2)	Tech.	Nom (Ω)	±3σ Variation Low (Ω)	±3σ Variation High (Ω)	Nom Err. %
INV (0)	PM	1653	92	7555	
INV (1)	HSPICE	1661	99	7574	0.48
IVN (0)	PM	1223	409	9754	
NAND2 (1, 1)	HSPICE	1232	406	9762	0.73
NAND2 (0, 0)	PM	3732	2551	14 022	
NAND2 (1, 1)	HSPICE	3752	2569	13 900	0.53
NOR2 (0, 0)	PM	2080	63	11 991	
NAND2 (1, 1)	HSPICE	2104	56	12 080	1.14
NOR2 (0, 0)	PM	1631	62	13 548	
NAND3 (1, 1, 1)	HSPICE	1655	67	13 440	1.45
NOR3 (0, 0, 0)	PM	1531	79	13 222	
NAND3 (1, 1, 1)	HSPICE	1500	73.6	13 090	2.07

*PM → proposed modeling technique.

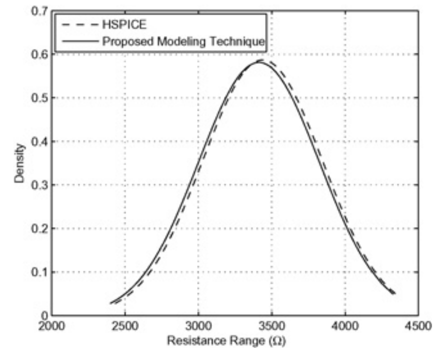

Fig. 10. Effect of process variation on the critical resistance of a bridge driven by two NAND2 gates.

that in comparison to HSPICE, the accuracy of the proposed technique reduces with higher number of transistors in series. For example, the difference in critical resistance calculation increases (compared to HSPICE) with higher number of transistors in series in the pull-up and pull-down networks as in case of fault-site driven by NOR3-NAND3 gates (the last row of Table VII). The accuracy can be improved further by reducing the value of the parameter $LIMIT$ (Figs. 3, 4, 6), however that will increase the computation time of the algorithm.

To demonstrate the runtime improvement of the proposed modeling technique in comparison to HSPICE, Fig. 11 shows the simulation time of the two modeling techniques (proposed and HSPICE) using 50 randomly generated fault-sites. The minimum and maximum times for the proposed technique are 5.2 s and 171.7 s, respectively. In case of maximum simulation time (171.7 s) the fault-site comprises of three NAND3 (3-input NAND) gates as the fanout gates, which requires longer logic threshold calculation time. Table II shows that logic threshold calculation is slowest in case of NAND3 gate when compared to other gates and it is only five-times faster than using HSPICE. However, using HSPICE the minimum and maximum times for critical resistance calculation are 1743.9 s and 2853.5 s, respectively. In general, when considering 350 fault-sites, the proposed technique is 53-times faster than HSPICE. A

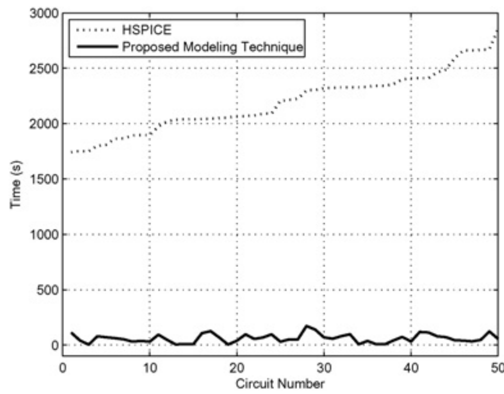

Fig. 11. Computation time improvement: proposed technique versus HSPICE.

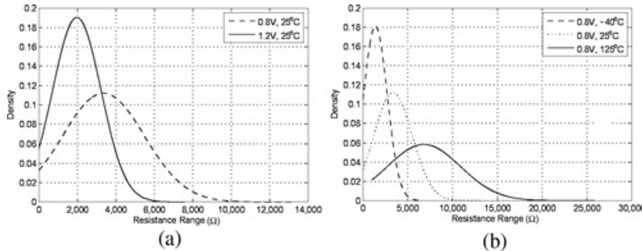

Fig. 12. Resistance range coverage for a fault-site driven by two inverters. (a) Effect of voltage variation. (b) Effect of temperature variation.

recent study has reported that it took nearly a week with eight computers working in parallel to generate a database for ISCAS'85, 89 benchmarks [9]. Using the proposed technique, we were able to re-generate the database for the same set of benchmarks in just over 3 h with approximately the same accuracy. These results clearly show that the proposed technique is fast and accurate when compared with HSPICE. The proposed technique can be incorporated into an ATPG process through database generation, for example as in [12]. However, instead of generating database through HSPICE, the proposed technique can be used for efficient database generation. The only downside of database generation is that it is technology specific and a new database will be needed for every technology node. Similarly, for fault simulation, the proposed technique can be used instead of HSPICE for very efficient fault simulation.

Next, we show the effect of PVT variations on resistance range coverage of bridge defect. Fig. 12(a) shows the effect of process and voltage variation on critical resistance of a bridge driven by two inverters. It can be seen that higher resistance range is detectable at lower voltage, which is in line with results reported in recent publications, for example see [12] and [20] for more details, note that temperature is constant at both voltage settings. For the same fault-site, we used the lowest voltage (0.8 V that covers highest resistance range) and changed the temperature to observe the effect of temperature variation. The simulation results are shown in Fig. 12(b), which shows the effect of process and temperature variation on critical resistance of a bridge. It can be seen that maximum resistance is covered at highest temperature and lowest voltage setting, which is in line with the discussion in Section II-C.

IV. Conclusion

This paper presented a fast and accurate technique to model the effect of process, voltage and temperature variation on resistive bridge defects by employing BSIM4 (I–$V_{\rm ds}$) transistor model. The effect of process variation is modeled by using three transistor parameters L, $V_{\rm th}$, and $\mu_{\rm eff}$, using Gaussian distribution. Variation in supply voltage is modeled by varying the supply voltage and temperature dependent transistor models are used to model the effect of temperature variation. The effect of voltage and temperature variation are incorporated by varying their respective values within prescribed (gate library) operating ranges. The proposed modeling technique employs an approximation algorithm for logic threshold calculation of gates' inputs driven by the bridge fault-site and a voltage ($V_{\rm ds}$) approximation algorithm for critical resistance calculation. The proposed modeling technique is extensively validated through comparison with HSPICE and it is shown that the worst-case error for logic threshold generation algorithm is 3.1% and $V_{\rm ds}$ (critical resistance) approximation algorithm is 0.73%, respectively, when operating in nominal (1.2 V, 25 °C) condition. Combining the two approximation algorithms together for critical resistance calculation, under the influence of PVT variation over 350 fault-sites, the worst-case error is 2.64%, when compared with HSPICE. In terms of run-time improvement, it is shown that on average over 350 fault-sites, the proposed modeling technique is 53-times faster than HSPICE (Fig. 11). The proposed modeling technique has been demonstrated on a 65 nm gate library, and it can be used for evaluating the impact of PVT variation on bridge defect using other technology nodes. The modeling flow (Fig. 9) will require a gate library with respective transistor model card, appropriate values of mean and standard deviation for the three transistor parameters (Table I) and voltage and temperature variation ranges through the gate library. This paper represented the first step toward efficient test generation and diagnosis of deep submicron bridge defects under the influence of process variation.

Acknowledgment

The authors would like to thank Dr. H. Oldham (ARM Fellow) and the anonymous reviewers for their useful comments.

References

[1] I. Polian, P. Engelke, B. Becker, S. Kundu, J.-M. Galliere, and M. Renovell, "Resistive bridge fault model evolution from conventional to ultradeep submicron," in *Proc. IEEE VLSI Test Symp.*, May 2005, pp. 343–348.

[2] M. Renovell, P. Huc, and Y. Bertrand, "The concept of resistance interval: A new parametric model for realistic resistive bridging fault," in *Proc. VLSI Test Symp.*, Apr. 1995, pp. 184–189.

[3] G. Roy, A. R. Brown, F. Adamu-Lema, S. Roy, and A. Asenov, "Simulation study of individual and combined sources of intrinsic parameter fluctuations in conventional nano-MOSFETs," *IEEE Trans. Electron Devices*, vol. 53, no. 12, pp. 3063–3070, Dec. 2006.

[4] *The International Technology Roadmap for Semiconductor*. (2007) [Online]. Available: http://www.itrs.net/links/2007itrs/home2007.htm

[5] D. Reid, C. Millar, G. Roy, S. Roy, and A. Asenov, "Analysis of threshold voltage distribution due to random dopants: A 100 000-sample 3-D simulation study," *IEEE Trans. Electron Devices*, vol. 56, no. 10, pp. 2255–2263, Oct. 2009.

[6] W. Zhao, F. Liu, K. Agarwal, D. Acharyya, S. Nassif, K. Nowka, and Y. Cao, "Rigorous extraction of process variations for 65 nm CMOS design," *IEEE Trans. Semicond. Manuf.*, vol. 22, no. 1, pp. 196–203, Feb. 2009.

[7] U. Ingelsson, B. Al-Hashimi, S. Khursheed, S. Reddy, and P. Harrod, "Process variation-aware test for resistive bridges," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 28, no. 8, pp. 1269–1274, Aug. 2009.

[8] S. Khursheed, S. Zhong, B. Al-Hashimi, R. Aitken, and S. Kundu, "Modeling the impact of process variation on resistive bridge defects," in *Proc. IEEE ITC*, Nov. 2010, pp. 1–10.

[9] U. Ingelsson, "Investigation into voltage and process variation-aware manufacturing test," Ph.D. dissertation, School Electron. Comput. Sci., Univ. Southampton, Southampton, U.K., Jul. 2009 [Online]. Available: http://eprints.soton.ac.uk/67440

[10] *BSIM4.6.5, Manual*, Univ. California, Berkeley, Feb. 2011 [Online]. Available: http://www-device.eecs.berkeley.edu/bsim3/bsim4_get.html

[11] J. Segura, J. Rossello, J. Morra, and H. Sigg, "A variable threshold voltage inverter for CMOS programmable logic circuits," *IEEE J. Solid-State Circuits*, vol. 33, no. 8, pp. 1262–1265, Aug. 1998.

[12] S. Khursheed, U. Ingelsson, P. Rosinger, B. Al-Hashimi, and P. Harrod, "Bridging fault test method with adaptive power management awareness," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 27, no. 6, pp. 1117–1127, Jun. 2008.

[13] Arizona State University. (2011, Feb.). *Predictive Technology Model* [Online]. Available: http://ptm.asu.edu

[14] Q. Liu and S. Sapatnekar, "A framework for scalable post-silicon statistical delay prediction under process variations," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 28, no. 8, pp. 1201–1212, Aug. 2009.

[15] P. Friedberg, Y. Cao, J. Cain, R. Wang, J. Rabaey, and C. Spanos, "Modeling within-die spatial correlation effects for process-design co-optimization," in *Proc. 6th ISQED*, Mar. 2005, pp. 516–521.

[16] M. Orshansky, L. Milor, P. Chen, K. Keutzer, and C. Hu, "Impact of systematic spatial intra-chip gate length variability on performance of high-speed digital circuits," in *Proc. IEEE/ACM ICCAD*, Nov. 2000, pp. 62–67.

[17] Y. Cao and L. Clark, "Mapping statistical process variations toward circuit performance variability: An analytical modeling approach," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 26, no. 10, pp. 1866–1873, Oct. 2007.

[18] J. C. Ku and Y. Ismail, "On the scaling of temperature-dependent effects," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 26, no. 10, pp. 1882–1888, Oct. 2007.

[19] HSPICE. (2011, May). *Data Sheet* [Online]. Available: http://www.synopsys.com/Tools/Verification/AMSVerification/CircuitSimulation/HSPICE/Documents/hspice_ds.pdf

[20] P. Engelke, I. Polian, M. Renovell, S. Kundu, B. Seshadri, and B. Becker, "On detection of resistive bridging defects by low-temperature and low-voltage testing," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 27, no. 2, pp. 327–338, Feb. 2008.

Shida Zhong received the B.S. degree with a distinction from the College of Information Engineering, Shenzhen University, Shenzhen, China, in 2008, and the M.S. degree with a distinction from the School of Electronics and Computer Science, University of Southampton, Southampton, U.K., in 2009. He is currently pursuing the Ph.D. degree with the School of Electronics and Computer Science, University of Southampton.

His current research interests include design for testing, fault modeling, process variation, and resistive bridge/open fault.

Saqib Khursheed received the B.E. degree in computer engineering from NED University, Karachi, Pakistan, in 2001, the M.S. degree in computer engineering from King Fahd University (KFUPM), Dhahran, Saudi Arabia, in 2004, and the Ph.D. degree in electronics and electrical engineering from the University of Southampton, Southampton, U.K., in 2010.

From 2005 to 2007, he was a Lecturer with KFUPM. Currently, he is a Research Fellow with the School of Electronics and Computer Science, University of Southampton. His current research interests include fault modeling, diagnosis, and test generation for deep submicrometer defects.

Bashir M. Al-Hashimi (M'99–SM'01–F'09) received the B.S. degree (with first-class classification) in electrical and electronics engineering from the University of Bath, Bath, U.K., in 1984, and the Ph.D. degree from York University, York, U.K., in 1989.

In 1999, he was with the microelectronics design industry. Then, he joined the School of Electronics and Computer Science, University of Southampton, Southampton, U.K., where he is currently a Professor of computer engineering and the Director of the Pervasive Systems Center. He has authored one book on SPICE simulation (Boca Raton, FL: CRC Press, 1995) and co-authored two books, *Power Constrained Testing of VLSI Circuits* (New York: Springer, 2002) and *System-Level Design Techniques for Energy-Efficient Embedded Systems* (New York: Springer, 2004). He edited the book *System-on-Chip: Next Generation Electronics* (Piscataway, NJ: IEEE Press, 2006). He has published over 240 papers in journals and refereed conference proceedings.

Prof. Al-Hashimi is a fellow of the British Computer Society. He is the Editor-in-Chief of the IEEE PROCEEDINGS: COMPUTERS AND DIGITAL TECHNIQUES, and an Editorial Board Member of the *Journal of Electronic Testing: Theory and Applications*, the *Journal of Embedded Computing*, and the *Journal of Low Power Electronics*. He was the General Chair of the 11th IEEE European Test Symposium in 2006, the Technical Program Chair of DATE in 2009, and the General Chair of DATE in 2011. He is the co-recipient of two Best Paper Awards: the James Beausang Award at the ITC 2000, relating to low power BIST for RTL data paths, and at the CODES-ISSS Symposium 2009, relating to low-energy fault-tolerance techniques. He is co-author of a paper in test data compression which has been selected for a Springer book featuring the most influential work over the ten years of the DATE conference.