References

[1] V.H. Champac, A. Rubio, and J. Figueras. Electrical model of the floating gate defect in CMOS IC's: Implications on IDDQ testing. *IEEE Transactions on Computer-Aided Design*, pages 359–369, March 1994.

[2] H. Cox and J. Rajski. Stuck-open and transition fault testing in CMOS complex gates. In *Proceedings of International Test Conference*, October 1988.

[3] C. Di and J.A.G. Jess. On accurate modeling and efficient simulation of CMOS opens. In *Proceedings of International Test Conference*, October 1993.

[4] J.A. Waicukauski et al. Fault simulation for structured VLSI. In *VLSI Systems Design*, Dec. 1985.

[5] Z. Barzilai et al. Efficient fault simulation of CMOS circuits with accurate models. In *Proceedings of International Test Conference*, October 1986.

[6] M. Favalli, M. Dalpasso, P. Olivo, and B. Ricco. Modeling of broken connections faults in CMOS ICs. In *Proceedings of European Design and Test Conference*, 1994.

[7] C.F. Hawkins, J.M. Soden, A.W. Righter, and F.J. Ferguson. Defect classes - an overdue paradigm for CMOS IC testing. In *Proceedings of International Test Conference*, October 1994.

[8] Alvin Jee and F. Joel Ferguson. Carafe: An inductive fault analysis tool for CMOS VLSI circuits. In *Proceedings of the IEEE VLSI Test Symposium*, 1993.

[9] N.K. Jha and J.A. Abraham. Design of testable CMOS logic circuits under arbitrary delays. *IEEE Transactions on Computer-Aided Design*, pages 264–269, July 1985.

[10] K.-J. Lee and M.A. Breuer. On the charge sharing problem in CMOS stuck-open fault testing. In *Proceedings of International Test Conference*, October 1990.

[11] W.M. Maly, P.K. Nag, and P. Nigh. Testing oriented analysis of CMOS ICs with opens. In *Proceedings of International Conference on Computer-Aided Design*, November 1988.

[12] G. Massobrio and P. Antognetti. *Semiconductor Device Modeling with SPICE*. McGraw-Hill, 1993.

[13] Meta-Software. *HSPICE User's Manual: Elements and Models*. 1992.

[14] S.M. Reddy, M.K. Reddy, and J.G. Kuhl. On testable design for CMOS logic circuits. In *Proceedings of International Test Conference*, October 1983.

[15] M. Renovell and G. Cambon. Electrical analysis and modeling of floating-gate fault. *IEEE Transactions on Computer-Aided Design*, pages 1450–1458, November 1992.

[16] Jeffrey Rogenski and F. Joel Ferguson. Characterization of opens in logic circuits. In *Proceedings of IEEE ASIC Conference*, 1994.

[17] B.J. Sheu, W.-J. Hsu, and P.K. Ko. An MOS transistor charge model for VLSI design. *IEEE Transactions on Computer-Aided Design*, pages 520–527, April 1988.

[18] B. Underwood, S. Kang, W.-O. Law, and H. Konuk. Fastpath: Robust path-delay test generator for standard scan designs. In *Proceedings of International Test Conference*, October 1994.

[19] J.-F. Wang, T.-Y. Kuo, and J.-Y. Lee. A new approach to derive robust tests for stuck-open faults in CMOS combinational logic circuits. In *Proceedings of Design Automation Conference*, June 1989.

5. Summary and Future Work

We presented a new fault simulation algorithm for realistic CMOS network breaks. We showed that Miller feedback and feedthrough effects can invalidate a test for a network break just as charge sharing can. We presented a new charge-based approach, in contrast to previous capacitance-based approaches, that can accurately and efficiently predict the worst case effects of Miller capacitances and charge sharing together. Our experimental results with ISCAS85 circuits showed that pretty high fault coverage values can be achieved with even random patterns and assuming the worst case test invalidation effects of transient paths, Miller capacitances and charge sharing. We also tried fault simulating the uncompacted SSA test set for each circuit, but that produced lower fault coverage figures than using 1024 random vectors. This shows the necessity to produce test patterns for network breaks. Our results in Table 4.2 shows that static hazard identification makes a significant difference in fault coverage numbers. Table 4.2 also shows that even though transient paths play a bigger role in test invalidation, the effect of Miller capacitances and charge sharing is big enough that they should not be ignored.

Potential future work includes more careful analysis of charge transfer from/to transistor bulks via forward biased p-n junctions during the floating period; including interwire capacitances in addition to wire-to-GND capacitances; associating test invalidation probability to a potential test, instead of always assuming the worst case scenario for test invalidation; and conducting fault coverage experiments on full-scan and non-scan sequential circuits.

on" column, static hazard identification is on. In Table 4.2, "charge off" means that the computation of ΔQ_{wiring} is turned off, that is, Miller effects and charge sharing are ignored. The "paths off" term means that transient paths to Vdd or GND are ignored. Note that when all of Miller effects, charge sharing, and transient paths are ignored, detection of a network break is only determined by single-stuck-at (SSA) detection in time frame 2 and the value of the cell output in time frame 1, so static hazards have no relevance. That is why the last column in Table 4.2 does not have an "SH on" or an "SH off" part. A fault coverage value in this last column might be greater than the SSA coverage of the circuit. For instance, the coverage value for c6288 in the last column is 99.9% while the SSA coverage for this circuit is 99.4%, because most of the undetectable SSA faults in c6288 are on fanout branches, and SSA detectability of fanout branches are not relevant in network break detection, only the SSA detectability of fanout stems are important.

Comparing the "SH on" and the "SH off" columns in both the "charge off" and the regular case shows that static hazard identification makes a quite significant difference on fault coverage values especially for circuits with high degree of reconvergent fanouts, such as c6288. Another observation from Table 4.2 is that in both the "SH on" and the "SH off" cases, disabling transient path identification has a larger impact on fault coverage than disabling Miller effects and charge sharing.

for c880 is much higher than the one for c1355 even though the percentage of short wires for c880 is much higher than the one for c1355. This shows that other factors in the circuit, such as the number of reconvergent fanouts, types of cells used, etc. can also significantly affect the fault coverage.

Because we use only six voltage levels for our charge difference computations, a look-up table can be constructed for all combinations of these voltages in the charge equations 3.3 to 3.8. We constructed such a look-up table only for the $(1 + V_r/\phi_j)^{(1-m_j)}$ and $(1 + V_r/\phi_j)^{(1-m_{jsw})}$ terms in equation 3.8, since taking the power of a real number is computationally expensive. Even though we did not construct such look-up tables for other equations, we ended up with reasonable CPU times as shown in Table 4.1, in fact, our CPU times per vector are always better than the ones reported by Di and Jess [3], where they used an HP-9000/700.

Circuit	# of network breaks	# of vectors simulated	% of breaks covered	% of short wires	CPU time (msec) per vector
c432	931	4000	87.8	27.7	3.8
c499	1403	5856	63.4	44.0	7.3
c880	1337	7360	94.8	20.6	2.0
c1355	2174	9120	74.5	4.9	9.4
c1908	2235	22528	75.5	34.0	9.0
c2670	3427	17920	78.2	16.7	6.2
c3540	4947	29984	91.6	17.0	13.1
c5315	7607	70528	94.0	20.3	15.1
c6288	10760	138624	87.4	7.9	128.2
c7552	9955	90912	86.5	23.2	22.3

Table 4.1: Random pattern simulation results for ISCAS85 circuits

Circuit			charge off		charge off
	SH on	SH off	SH on	SH off	paths off
c432	84.0	89.5	88.0	92.6	98.7
c499	60.4	80.8	73.0	90.1	99.5
c880	89.3	90.6	92.4	93.3	98.6
c1355	69.6	83.3	77.6	87.8	96.9
c1908	54.8	63.5	63.6	70.9	86.5
c2670	71.2	76.5	75.1	79.6	85.7
c3540	77.1	85.6	81.7	88.7	96.6
c5315	83.7	91.0	87.6	93.9	98.9
c6288	76.8	96.0	82.8	97.2	99.9
c7552	72.0	80.7	76.9	84.4	89.9

Table 4.2: Fault coverage results using 1024 random patterns with varying accuracy levels

Table 4.2 shows our fault coverage results using 1024 random vectors for each circuit. The fault coverage numbers in an "SH off" column are obtained by turning the static hazard identification off, that is, every 00 is treated as S0, and every 11 is treated as S1. In an "SH

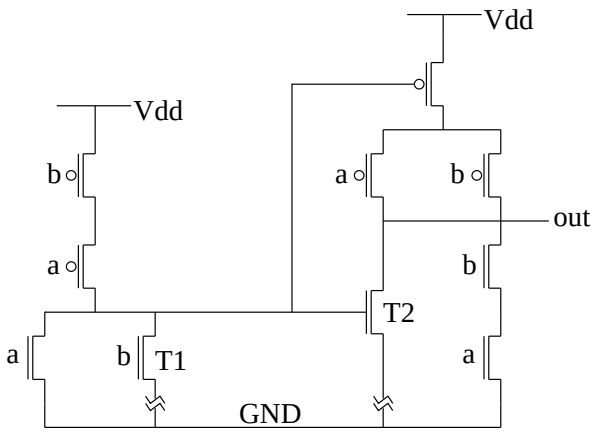

Figure 4.1: Two network breaks in an XOR gate caused by a single contact break in the layout

using the MCNC cell library, all the circuits but c1355 and c6288 end up having XOR or XNOR gates in their implementations. An XOR gate is implemented using a NOR gate and an AOI21 gate, and an XNOR gate is implemented using a NAND gate and an OAI21 gate in the MCNC library. Figure 4.1 shows an XOR gate with two n-network breaks in it. In the layout of this gate, transistors $T1$ and $T2$ share a diffusion contact to connect to the GND terminal. A break in this diffusion contact causes the two network breaks shown in Figure 4.1. Because we assumed a single network break in our fault simulation algorithm described in previous sections, we handle this case as follows. One possible solution is to exercise the AOI21 gate in a fault-free manner so that we can assume the network break exists only in the NOR gate. The only two-vector sequence that might detect the NOR gate network break is $a = S0$ and $b = 01$. But, this sequence activates the broken path in the AOI21 gate in both time frames 1 and 2, therefore we cannot use this sequence. The other solution is to exercise the NOR gate in a fault-free manner so that we can assume the network break exists only in the AOI21 gate. In this case, $a = 10$ and $b = S0$ is the only potential test, and will detect this break fault if the XOR output is observable in time frame 2, and the wire driven by the XOR gate is big enough to handle Miller effects. Two simultaneous breaks in the p-networks of an XOR gate, and two simultaneous network breaks in an XNOR gate are treated similarly.

The results of running our fault simulator with the ISCAS85 benchmark circuits on a DECstation 5000/240 with 128Mb of memory are shown in Tables 4.1 and 4.2. In the experiments shown in Table 4.1, we kept generating random patterns until a certain number of successive random patterns do not detect any further network break, and that number is proportional to the number of cells used in the circuit. A two-vector pattern is formed by using two successive random vectors $v1$ and $v2$, and the next two-vector pattern is formed by using $v2$ and $v3$, where $v3$ is the next random vector after $v2$. We call a wire in a circuit **short wire** if its capacitance to GND is less than or equal to 35fF. We chose 35fF arbitrarily mostly because the wiring capacitance we used in Figure 2.2 was also 35fF. All circuits but c1355 and c6288 have double digit short wire percentages, because all these circuits have XOR or XNOR gates in them, and such a gate consists of two primitive gates with about 10fF wiring between them. Note that it is easier for a test to be invalidated by Miller effects and charge sharing as the wiring capacitance gets smaller. But, fault coverage

4. Implementation and Experimental Results

We implemented the fault simulation algorithm described in the previous section, and used the ISCAS85 benchmark circuits implemented with the MCNC standard cell library for our experiments. For charge difference computations, we obtained the BSIM model parameters from MOSIS for the 1.2μ Orbit n-well fabrication process. We extracted the wiring capacitance of each wire in a circuit using *Magic* with this 1.2μ technology. We took L0_th to be 1.8V and L1_th to be 3.2V leaving a guardband of 1.4V wide.

For every standard cell used in the ISCAS85 benchmark circuits, we performed the following tasks. We used the public domain *ext2spice* program to determine the area and the perimeter of the diffusion region for the drain and source terminals of each transistor in the cell. We used *Carafe* to get a list of realistic break faults in the cell, and eliminated the breaks that are not network breaks.

For each internal node in each faulty cell, our program generates the connection function between the internal node and the faulty cell output, where the **connection function** between two nodes in a cell denotes a sum-of-products expression, where each product term describes the condition to activate a transistor path between the two nodes, and a product term exists for every possible transistor path between the two nodes. This function is used in determining the initial and final voltages in the faulty cell as described in Section 3.2. Actually, we first generate the described connection function for each internal node of the fault-free cell. And, for every faulty cell produced from this fault-free cell with a network break, we list the faulty cell internal nodes that are identical to the ones in the fault-free cell, and then we list the new internal nodes with their connection functions. This way, we save memory by generating a connection function only for a new internal node in a faulty cell.

Again for each faulty cell, our program generates the connection function between the cell output and either Vdd or GND depending on whether the break is in the p-network or in the n-network. This function is used to determine whether the faulty cell output will float in time frame 2, and whether a transient path to Vdd or GND can exist to invalidate a test.

For each internal node in a fault-free cell, our program generates the connection function to the Vdd or GND node depending on whether the internal node is in the p-network or in the n-network. This function together with the connection function to the cell output is used in determining the initial and final voltages for Miller feedback effect as described in Section 3.2.

The standard cells are processed as described above only once, not every time before a circuit is fault simulated. Our program performs parallel pattern simulation using our eleven-value logic algebra to determine the logic value on each wire in time frames 1 and 2 in the fault-free circuit. Then, we perform PPSFP (parallel pattern single fault propagation) [4] only in time frame 2 to determine the stuck-at-0 and stuck-at-1 detectability of the wires. If a stuck-at-0 on a wire is detectable in time frame 2 and the wire is logic-0 in time frame 1, then our program checks for possible transient paths to Vdd and computes the ΔQ_{wiring} in equation 3.1 for the p-network breaks in the cell that drives the wire. The n-network breaks are processed similarly.

Even though only c432 and c499 have XOR or XNOR gates in their gate level descriptions among all the ISCAS85 benchmark circuits, when these circuits are technology mapped

```
GetNodeInitFinal( V_ds,init, V_ds,final, static_current_possible )
BEGIN
    static_current_possible = TRUE;
    IF (O is initialized to GND) THEN
        IF (there is at least one path of transistors from ds to Vdd such that
                the gates of all these transistors are S1) THEN
            V_ds,init = max_n;
            V_ds,final = max_n;
        ELSE
            V_ds,init = GND;
            IF (ds is connected to GND at the end of time frame 2) THEN
                V_ds,final = GND;
            ELSE
                V_ds,final = max_n;
                IF (ds is disconnected from the cell output at the end of time frame 2 OR
                        the cell output is logic-0 at the end of time frame 2) THEN
                    static_current_possible = FALSE;
                ENDIF
    ELSE
        /* O is initialized to Vdd */
        IF (there is at least one path of transistors from ds to GND such that
                the gates of all these transistors are S1) THEN
            V_ds,init = GND;
            V_ds,final = GND;
        ELSE
            V_ds,init = max_n;
            IF (ds is connected to Vdd at the end of time frame 2) THEN
                V_ds,final = max_n;
            ELSE
                V_ds,final = GND;
END

Get_MFB_InitFinal()
BEGIN
    GetNodeInitFinal( V_drain,init, V_drain,final, drain_SCP );
    GetNodeInitFinal( V_source,init, V_source,final, source_SCP );
    IF (O is initialized to GND) THEN
        IF ( drain_SCP == FALSE AND V_source,final == GND ) THEN
            V_drain,final = GND;
        ELSE IF ( source_SCP == FALSE AND V_drain,final == GND ) THEN
            V_source,final = GND;
END
```

Figure 3.1: Determining drain/source initial/final voltages for the Miller feedback effect

feedthrough effect around fcn, but when the fcn voltage exceeds max_n, charge cannot be transferred from O to fcn.

For any transistor t connected to fcn, if the logic value at t's gate gt is neither S0 nor S1, then we take the initial voltage for gt as Vdd, and the final voltage as GND. We do this even when gt's logic value is 01, because a 01 can create a falling transition between two rising transitions, and during the falling transition the voltage at fcn may be max_n or lower thus enabling charge transfer from O to the drain or source (whichever is connected to fcn) of t, but during the rising transitions the voltage at fcn may be max_n or higher thus preventing charge transfer onto O. When gt's logic value is S0 or S1, then the initial and final gt voltages are both GND or both Vdd, respectively. This completes Subcase 2.2.

So far in Section 3.2, we assumed that fcn was an internal node. We now describe how to determine the initial and final voltages when fcn is the same as node O. When O is initialized to GND, we determine the initial and final gate voltages of all the transistors, either in the n-network or p-network, connected to O as shown in Table 3.1. Obviously, $V_{fcn,init} = GND$ and $V_{fcn,final} = L0\_th$ in this case. The case when O is initialized to Vdd is similar.

In order to estimate the worst case Miller feedback effects, we need to compute $\Delta Q_{g,f}$ in equation 3.1 for each fanout transistor f of O. For this, the initial and final voltages at the gate, drain, and source terminals of f are needed. There are four cases to consider depending on whether f is an nMOS or a pMOS transistor, and whether O is initialized to GND or Vdd. Due to lack of space, we only discuss the two cases where f is an nMOS transistor. The other two cases where f is a pMOS transistor are similar.

Let $\mathbf{V_{g,f,init}}$ and $\mathbf{V_{g,f,final}}$ denote the initial and final voltages at f's gate. Obviously, $V_{g,f,init} = GND$ and $V_{g,f,final} = L0\_th$ when O is initialized to GND, and $V_{g,f,init} = Vdd$ and $V_{g,f,final} = L1\_th$ when O is initialized to Vdd. Let **ds** denote the drain or the source terminal of f. Let us assume that ds is an internal node, that is, it is neither GND nor the output of cell **fc** in which f is located, then routines GetNodeInitFinal and Get_MFB_InitFinal in Figure 3.1 show how we determine the initial and final voltages $\mathbf{V_{ds,init}}$ and $\mathbf{V_{ds,final}}$ for f's drain and source. In the case O is initialized to GND, when O reaches L0_th at the end of time frame 2, the nMOS transistor f will be weakly turned on. If the output of fc is sensitized to O, then a static current will be flowing in fc as we discussed in Section 3.1. The flag **static_current_possible** in routine GetNodeInitFinal is used to determine when it is impossible for fc's output to be sensitized to O due to the logic values at the side-inputs of fc. When ds is fc's output, then the max_n terms in Figure 3.1 will be replaced by Vdd.

Logic value at gt	$V_{g,t,init}$	$V_{g,t,final}$
11, X1, 10, 1X, X0, XX	Vdd	GND
S0, 00, 0X	GND	GND
S1	Vdd	Vdd
01	GND	Vdd

Table 3.3: The worst case gate initial and final voltages for Subcase 1.2, $max\_n < L1\_th$

CASE 2 : The condition for CASE 1 is not satisfied, and there is no path of transistors from fcn to O such that the gates of all these transistors are S1 if fcn is in the p-network, and S0 if fcn is in the n-network. This case is for intermittent connections between fcn and O during the floating period. As in CASE 1, there are four subcases depending on whether fcn is in the p-network or in the n-network, and whether O is initialized to GND or Vdd. Due to lack of space, we only discuss two subcases where fcn is in the n-network. The other two subcases where fcn is in the p-network are similar.

Subcase 2.1 : Node fcn is in the n-network, and O is initialized to GND. In this case, if fcn is connected to GND at the end of time frame 1, then $V_{fcn,init} = GND$, otherwise $V_{fcn,init} = max\_n$. Note that if the p-network break disconnects the whole p-network from O, only the Miller feedthrough and feedback mechanisms can create an initial voltage of max_n at fcn, therefore max_n is a pretty pessimistic initial voltage, but not impossible. If fcn is connected to O at the end of time frame 2, then $V_{fcn,final} = L0\_th$, otherwise $V_{fcn,final} = GND$ because even when $V_{fcn,init} = max\_n$, fcn might be connected to O while O is still at GND voltage, and this may pull down the fcn voltage very close to GND because the total capacitance of O might be much larger than the capacitance of fcn, and fcn may never connect to O again in the rest of time frame 2.

For any transistor t connected to fcn, if the logic value at t's gate gt is S0 or S1, then the initial and final gt voltages are both GND or both Vdd, respectively. Otherwise, we take the initial voltage for gt as GND, and the final voltage as Vdd. This might sound counter-intuitive for the case when 10 is the logic value for gt, but consider the following scenario during the floating period. While the voltage of fcn is GND, a falling transition arrives at gt. This will bring in more charge to the drain or source terminal (whichever is connected to fcn) of t. But, this charge will be coming from the bulk of transistor t due to the forward biased p-n junction between fcn and the bulk. In order to make our charge conservation assumption we made in Section 3.1 hold, we treat even a 10 at gt as 01, because a 10 can create a rising transition between two falling transitions, and the falling transitions may cause charge transfer from the bulk. Repetitive falling and rising transitions at gt coupled with connections of fcn to O at appropriate times can create an effect of pumping charge from the bulk to node O. But, we ignore this seemingly unlikely effect, and leave its detailed discussion to another paper. In fact, a similar phenomenon can also happen in Subcase 1.1.

Subcase 2.2 : Node fcn is in the n-network, and O is initialized to Vdd. If fcn is connected to O at the end of time frame 1, then $V_{fcn,init} = max\_n$, otherwise $V_{fcn,init} = GND$. If fcn is connected to O at the end of time frame 2, and $L1\_th < max\_n$, then $V_{fcn,final} = L1\_th$, otherwise $V_{fcn,final} = max\_n$. If fcn is disconnected from O at the end of time frame 2, the actual fcn voltage might be larger than max_n due to Miller

Logic value at gt	$V_{g,t,init}$	$V_{g,t,final}$
01, 11, 0X, X1, XX, 1X	GND	Vdd
S0, 00, 10, X0	GND	GND
S1	Vdd	Vdd

Table 3.1: The worst case gate initial and final voltages for Subcase 1.1

Subcase 1.1 : Node fcn is in the n-network, and O is initialized to GND. In this case, $V_{fcn,init} = GND$, and $V_{fcn,final} = L0\_th$. Table 3.1 shows how the worst case $V_{g,t,init}$ and $V_{g,t,final}$ values are determined for each transistor t connected to node fcn, depending on the logic value at t's gate **gt**.

The non-obvious cases in Table 3.1 are when the logic values at gt are 11 and 10. When the logic value is 11, it is possible due to a glitch that the voltage at gt is GND at t_{init}. Even when the voltage of gt at t_{init} is Vdd, the following scenario might occur after t_{init}: While O is at GND voltage, a glitch causes a falling transition at gt, which forces the voltage at fcn below GND, which makes the p-n junction between fcn and the bulk of t forward-biased, because the bulk of an nMOS transistor is connected to GND. This way, positive charge is transferred from t's bulk to node fcn. Note that this charge transfer is happening during the floating period, which will violate our charge conservation assumption of Section 3.1 during the floating period. So, by assuming $V_{g,t,init}$ to be GND, we are effectively moving the beginning of the floating period from t_{init} to the point this charge transfer is completed, this way we can still assume charge conservation. The reason we take $V_{g,t,init}$ to be GND when the logic value at gt is 10 is the same.

Subcase 1.2 : Node fcn is in the n-network, and O is initialized to Vdd. In this case, $V_{fcn,init} = max\_n$. Consider the case where $max\_n \geq L1\_th$. Then, $V_{fcn,final} = L1\_th$, and Table 3.2 shows how the worst case $V_{g,t,init}$ and $V_{g,t,final}$ values are determined for transistor t connected to node fcn, depending on the logic value at t's gate **gt**.

Logic value at gt	$V_{g,t,init}$	$V_{g,t,final}$
10, 1X, X0, XX	Vdd	GND
S0, 00, 0X	GND	GND
S1, 11, X1	Vdd	Vdd
01	GND	Vdd

Table 3.2: The worst case gate initial and final voltages for Subcase 1.2, $max\_n \geq L1\_th$

When $max\_n < L1\_th$, then $V_{fcn,final} = max\_n$, and Table 3.3 shows the worst case initial and final voltages for the gate of transistor t. The difference between Tables 3.2 and 3.3 is that the initial and final gate voltages for 11 and X1 were both Vdd in Table 3.2, but they changed to Vdd and GND in Table 3.3. The reason is as follows. Due to a glitch during the floating period, gt can make a falling transition absorbing charge from floating O. Because the voltage of O may never go below L1_th during the floating period, and $max\_n < L1\_th$, the charge absorbed may not be transferred back to O when gt rises back to Vdd. Another difference with the case $max\_n < L1\_th$ is that if ΔQ_{fcn} in equation 3.2 comes out to be a negative value implying that net positive charge will be transferred from fcn to O, we make ΔQ_{fcn} equal to zero, because charge transfer from fcn to O is not guaranteed since O may never go below L1_th during the floating period.

$$C_{junction} = \frac{C_j \cdot A_{diff}}{(1 + V_r/\phi_j)^{m_j}} + \frac{C_{jsw} \cdot P_{diff}}{(1 + V_r/\phi_j)^{m_{jsw}}}$$

where C_j and C_{jsw} are the capacitances at zero-bias voltage, for unit area and for unit perimeter of the diffusion; m_j and m_{jsw} are the substrate-junction and perimeter capacitance grading coefficient; and ϕ_j is the junction potential. All of these parameters have constant values for the nMOS and pMOS transistors depending on the fabrication process used. Finally, A_{diff} and P_{diff} denote the area and the perimeter of the diffusion. Integrating $C_{junction}$, we obtain the charge expression for the p-n junction as follows:

$$\begin{aligned} \Delta Q_{junction} &= \int_{V_{r,init}}^{V_{r,final}} C_{junction} \cdot dV_r \\ &= \frac{C_j \cdot A_{diff} \cdot \phi_j}{1 - m_j} \cdot \left(1 + \frac{V_r}{\phi_j}\right)^{(1-m_j)} \Bigg|_{V_{r,init}}^{V_{r,final}} + \\ &\quad \frac{C_{jsw} \cdot P_{diff} \cdot \phi_j}{1 - m_{jsw}} \cdot \left(1 + \frac{V_r}{\phi_j}\right)^{(1-m_{jsw})} \Bigg|_{V_{r,init}}^{V_{r,final}} \end{aligned} \tag{3.8}$$

The $\Delta Q_{junction,fcn}$ term in equation 3.2 is computed using equation 3.8 for node fcn.

3.2 Initial and Final Voltages for Charge Computations

In this section, we describe how we determine the worst case voltage values at transistor terminals at t_{init} and at t_{final} in order to compute ΔQ_{wiring} in equation 3.1. We use only six voltage values as the initial and final voltages of transistor terminals to compute the charge differences given by equations 3.3 to 3.8. These values are Vdd, GND, L0_th, L1_th, max_n, and min_p, where **max_n** is the maximum voltage an internal node in an n-network can achieve through a path to Vdd without any Miller feedthrough effect, and **min_p** is the minimum voltage an internal node in a p-network can achieve through a path to GND without any Miller feedthrough effect. For the 1.2μ process we used, max_n was around 3.3V, and min_p was around 1.2V with Vdd equal to 5V.

In order to compute $\Delta Q_{ds,t}$ and $\Delta Q_{junction,fcn}$ in equation 3.2, we need the gate voltages at t_{init} and at t_{final} for every transistor t connected to fcn, which we denote as $\mathbf{V_{g,t,init}}$ and $\mathbf{V_{g,t,final}}$, and we need the initial and final voltages of fcn, which we denote as $\mathbf{V_{fcn,init}}$ and $\mathbf{V_{fcn,final}}$. Let us assume that node fcn is an internal node in the faulty cell, and not the output node. There are two cases to consider:

CASE 1 : There is at least one path of transistors from fcn to O such that the gates of all these transistors are S0 if fcn is in the p-network, and S1 if fcn is in the n-network. Under this, there are four subcases depending on whether fcn is in the p-network or in the n-network, and whether O is initialized to GND (p-network break) or Vdd (n-network break). Due to lack of space, we only discuss the two subcases where fcn is in the n-network. The other two subcases where fcn is in the p-network are similar.

$$g = 1 - \frac{1}{1.744 + 0.8364 \cdot (zphi + V_{sb})}$$

$$V_{DSAT} = \frac{V_{gs} - V_{th}}{\alpha_x}$$

Any term that starts with "z" in the equations above such as $zvfb$ or $zphi$ is a BSIM electrical parameter taking the transistor size into account, and we compute it as follows [12]:

$$zP = P + \frac{P_L}{L - DL} + \frac{P_W}{W - DW}$$

where P is a process parameter such as vfb or phi, P_L and P_W are the length and width sensitivities of parameter P, W and L are the drawn transistor width and length, and DW and DL are the size changes to W and L due to various fabrication steps. The values of P, P_L, P_W, DL, and DW are all determined by the fabrication process. We obtained the values of all the BSIM parameters from MOSIS for the 1.2μ Orbit n-well fabrication process.

Finally, $cap = C_{ox} \cdot (W - DW) \cdot (L - DL)$ where C_{ox} is the gate-oxide capacitance per unit area.

We assumed V_{ds} to be zero in equation 3.5, which is used for computing the gate charge of a fanout transistor from the faulty cell. The static current might be non-zero in a fanout cell when O reaches L0_th or L1_th, but this static current will not cause a substantial voltage drop across the drain and source of a transistor in triode region. Let us show this for the case when O is initialized to 0V. The final value for O is L0_th, thus the nMOS transistor O is connected to in the fanout cell **fc** will be turned on. If the output of fc is sensitized to O, then some static current will be flowing through the nMOS transistor which is now in saturation region. The output of fc is now at logic 1, because O is at logic 0 even with L0_th voltage on it. Therefore, the pMOS transistor O is connected to in fc is in triode region, and the voltage drop across its channel is about Vdd minus the voltage at fc's output. Since fc's output is at logic 1, we ignore this voltage drop.

We also assumed V_{ds} to be zero in equation 3.6, which is used for computing the drain or source part of the channel charge for a transistor in the faulty cell. If this transistor is in triode region at the beginning or end of the floating period, we do not expect a drain current flowing through this transistor, since there is no conducting path from Vdd to GND. We did not need an equation for the drain or source part of the channel charge for a transistor in saturation region, because no transistor in the faulty cell will be in saturation at the boundaries of the floating period.

To compute $\Delta Q_{g,f}$ in equation 3.1 we use equations 3.3, 3.5, and 3.7 depending on the region the fanout transistor f is in with the initial and final voltages at its terminals. To compute $\Delta Q_{ds,t}$ in equation 3.2 we use equations 3.4 and 3.6 depending again on the region transistor t is in. We also include in $\Delta Q_{g,f}$ and $\Delta Q_{ds,t}$ the charge difference due to the gate-diffusion overlap capacitances.

The reverse biased p-n junction between the diffusion region and the bulk of a transistor forms the capacitance $C_{junction}$. The diffusion region is either the source or the drain of a transistor. From Massobrio and Antognetti [12], $C_{junction}$ can be expressed as a function of the reverse bias voltage V_r as follows:

in time frame 1, implying a p-network break, then we assume that O will reach L0_th at the end of time frame 2, because $L0\_th$ is the maximum tolerable voltage without test invalidation. Similarly, if O is initialized to Vdd, implying an n-network break, we assume that O will be reduced to L1_th at the end of time frame 2. The test becomes invalidated if

$$C_{O,wiring} * L0\_th < \Delta Q_{wiring} \quad \text{when } O \text{ is initialized to GND, and}$$

$$C_{O,wiring} * (Vdd - L1\_th) < -\Delta Q_{wiring} \quad \text{when } O \text{ is initialized to Vdd.}$$

Otherwise, the test is declared to be valid if there are no transient paths to Vdd or GND that will invalidate the test.

The following equations, 3.3 through 3.7, are taken from Sheu, Hsu, and Ko [17] to express the charge stored on a transistor gate, denoted by Q_g, and the charge stored by the source and the drain terminals in the channel of a transistor, denoted by Q_d and Q_s. The interested reader can find the derivations of these equations in the Sheu, Hsu, and Ko [17] paper. Additionally, we included the sensitivity of model parameters to transistor lengths and widths. These equations are for an nMOS transistor. For a pMOS transistor, the right hand sides of equations 3.3 to 3.7 need to be negated together with the interterminal voltages.

Subthreshold region, $V_{gs} \leq V_{th}$ and $V_{gb} > zvfb$:

$$Q_g = \frac{cap \cdot zk1^2}{2} \cdot (-1 + \sqrt{1 + \frac{4 \cdot (V_{gb} - zvfb)}{zk1^2}}) \tag{3.3}$$

$$Q_d = Q_s = 0 \tag{3.4}$$

Triode region, $V_{gs} > V_{th}$ and $V_{ds} \leq V_{DSAT}$:

$$Q_g = cap \cdot (V_{gs} - zvfb - zphi) \quad \text{with } V_{ds} = 0 \tag{3.5}$$

$$Q_d = Q_s = -0.5 \cdot cap \cdot (V_{gs} - V_{th}) \quad \text{with } V_{ds} = 0 \tag{3.6}$$

Saturation region, $V_{gs} > V_{th}$ and $V_{ds} > V_{DSAT}$:

$$Q_g = cap \cdot (V_{gs} - zvfb - zphi - \frac{V_{gs} - V_{th}}{3 \cdot \alpha_x}) \tag{3.7}$$

The terms V_{th}, α_x, and V_{DSAT} used in the preceding equations are defined as follows [17] [12], but in these definitions we assumed the BSIM model parameters $k2$, η, and $U1$ [12] to be zero in order to match the definitions in HSPICE [13].

$$V_{th} = zvfb + zphi + zk1 \cdot \sqrt{zphi + V_{sb}}$$

$$\alpha_x = 1 + \frac{g \cdot zk1}{2 \cdot \sqrt{zphi + V_{sb}}}$$

1. Each transistor drain or source terminal **ds** connected to fcn stores charge in the intrinsic, or channel, area of the transistor when the transistor is on [17]. This charge is a function of the voltages at the terminals of the transistor **t** and the size of t. Some charge is also stored on ds due to the gate overlap capacitance, which is a linear function of the gate-drain or gate-source voltage and the width of t. We denote the charge on ds of t as $\mathbf{Q_{ds,t}}$.
2. Charge is stored in the diffusion regions that make up the transistor terminals connected to fcn, because of the reverse biased p-n junctions between these diffusion regions and the transistor bulks. This charge is a function of the reverse bias voltage and the size of the p-n junctions, and we denote it as $\mathbf{Q_{junction,fcn}}$.

The following two charge components exist only for the faulty cell output O:

3. Charge is stored on each transistor gate connected to O. This charge is a function of the voltages at the terminals of the fanout transistor **f** and the size of f. We denote this charge as $\mathbf{Q_{g,f}}$.
4. Charge is stored on the metal wire that connects the faulty cell to its fanout cells, due to the linear capacitance to GND and to nearby wires. In this paper, we ignore the interwire capacitances, and consider only the capacitance to GND, which we refer to as the **wiring capacitance**. We denote this charge as $\mathbf{Q_{wiring}}$.

Let us assume for now that the total charge stored at the nodes in FCN at t_{init} is the same as the charge stored at t_{final}, where $\mathbf{t_{init}}$ denotes the beginning of the floating period, and $\mathbf{t_{final}}$ denotes the end of the floating period, which is also the end of time frame 2. So, we will assume that the net charge difference in the nodes of FCN is zero, that is, charge is conserved during the floating period. We are interested in the worst case charge difference on the wiring capacitance $\mathbf{C_{O,wiring}}$, because this charge difference ΔQ_{wiring} will give us the worst case voltage change on O. Because the net charge difference in the nodes of FCN is zero, any charge difference on the wiring capacitance, which represents only component 4 of the charge stored on O, must come from the charge differences on the remaining three charge components of O and from the charge differences in the nodes of I. Therefore, ΔQ_{wiring} can be expressed as follows.

$$\Delta Q_{wiring} = -\left(\sum_{fcn \in FCN} \Delta Q_{fcn} + \sum_{f \in F} \Delta Q_{g,f} \right) \tag{3.1}$$

$$\Delta Q_{fcn} = \Delta Q_{junction,fcn} + \sum_{t \in T_{fcn}} \Delta Q_{ds,t} \tag{3.2}$$

where F is the set of transistors whose gates are connected to O, and T_{fcn} is the set of transistors whose drain or source terminals are connected to fcn. Given a circuit, the worst case charge differences are determined only by the worst case voltage differences from t_{init} to t_{final}. Section 3.2 describes how we obtain these worst case voltages at t_{init} and at t_{final} from the elements of our eleven-value logic algebra described in Section 2. In equation 3.2, the $\Delta Q_{junction,fcn}$ term is for charge sharing between nodes fcn and O, and the summation term is for the Miller feedthrough effect of the transistors in T_{fcn}. In equation 3.1, the second summation term is for the Miller feedback effect.

If ΔQ_{wiring} creates a sufficient voltage difference on O, then the test will be invalidated. Let **L0_th** and **L1_th** denote the maximum voltage that is still a logic 0 and the minimum voltage that is still a logic 1, respectively. If the faulty cell output O is initialized to 0V

3. The Fault Simulation Algorithm

Our fault simulation algorithm declares a two-vector sequence a test for a network break if the sequence cannot be invalidated by transient paths to Vdd or GND, Miller feedback and feedthrough effects, and charge sharing. The first thing we do with a two-vector sequence is to perform gate level simulation using our eleven-value logic algebra. We assume that if a circuit input has the same logic value in time frames 1 and 2, then that input has no static hazard, that is, it is glitch-free. For an AND gate to have an S0 value at its output, at least one of its inputs must be S0, and to have an S1 at its output, all of its inputs must be S1. An OR gate is processed similarly.

In order to guarantee that no transient path to Vdd invalidates a test for a p-network break, all the paths from the faulty cell output to Vdd in the p-network must have at least one transistor with S1 value at its gate. This is a necessary condition, because if a path has no transistor with an S1 at its gate, then that path can be momentarily activated causing current flowing from Vdd to the faulty cell output, making the faulty cell behave like the fault-free one. It is also a sufficient condition, because having at least one pMOS transistor turned off for every possible path in the p-network of the faulty cell throughout time frame 2 guarantees that no current can flow from Vdd to the faulty cell output. Similarly, in order to guarantee no transient path to GND for an n-network break, all the paths from the faulty cell output to GND must have at least one transistor with S0 value at its gate.

In order to guarantee that a test will not be invalidated by Miller effects and charge sharing, our fault simulator uses a charge-based approach that computes the worst case charge difference on the floating faulty cell output. This approach is described next.

3.1 A Charge-Based Approach

When a test for a network break is applied, the faulty cell output becomes floating at some point during time frame 2, and stays floating in the rest of time frame 2. We refer to this time period as the **floating period**. We assume that time frame 2 is short enough so that the transistor leakage currents can be ignored. During the floating period, voltage changes at the gates of the transistors in the faulty cell can displace charge from, or bring in more charge to, the drain and source terminals (Miller feedthrough effect); the output may be connected to some internal nodes in the faulty cell resulting in charge sharing; and voltage changes at the internal nodes of the fanout cells can displace charge from, or bring in more charge to, the gate terminals of the transistors fed by the floating output (Miller feedback effect). Assuming constant values for the Miller and diffusion capacitances would be too pessimistic or too optimistic, because the Miller capacitances change by more than a factor of five as shown in Section 2.1, and the p-n junction diffusion capacitances at internal nodes change by more than a factor of two. So, our approach is based on computing the worst case changes in electrical charge as a function of the worst case voltage changes at the inputs of the faulty cell and its fanout cells.

Let us now identify the components of the charge stored at the faulty cell output O, and at a faulty cell internal node. Let $\mathbf{I}$ denote the set formed by the faulty cell internal nodes that might be connected to O during the floating period, and $\mathbf{FCN} = I \cup \{O\}$ where FCN stands for the set of Faulty Cell Nodes. The following two components exist for the charge stored on any faulty cell node $fcn \in FCN$.

2.2 Charge Sharing

We assume that the next transition in time frame 2 is at line $a3$ between 9ns and 10ns due to a glitch. Now, *out* is connected to internal nodes $p1$ and $p2$ in the OAI31 cell. Since $p1$ and $p2$ were initialized to 5V during time frame 1 by starting $a1$ at 0V, charge transfer from $p1$ and $p2$ to *out* raises the *out* voltage to 2.3V from 9ns to 12ns as shown in Figure 2.3. The p-n junction capacitance of node $p2$ changes from 26.7fF to 14.9fF when the voltage at $p2$ changes from 5V to 2.3V. When the voltage at $p2$ drops to 1V, its capacitance drops to 13.2fF.

2.3 Miller Feedthrough Effect

The next event is a rising transition at line $a2$ between 12ns and 13ns. Due to the gate-drain and gate-source (Miller feedthrough) capacitances of the pMOS transistor $a2$ is connected to, this transition raises the voltages on $p1$ and $p2$. Please note that the Miller feedthrough capacitance is not only due to the gate-diffusion overlap, but it can go up to half of the total gate capacitance when the transistor is on as in the case of Miller feedback. The voltage increase on $p2$ enables additional charge transfer from $p2$ to *out* between 12ns and 14ns. The final event is a rising transition at line $a3$ between 14ns and 15ns, which bumps up the *out* voltage to its final value of 2.63V. At this point, the output of the second inverter in Figure 2.2 is a perfect 0V, the same value as in the fault-free circuit, so the test is completely invalidated.

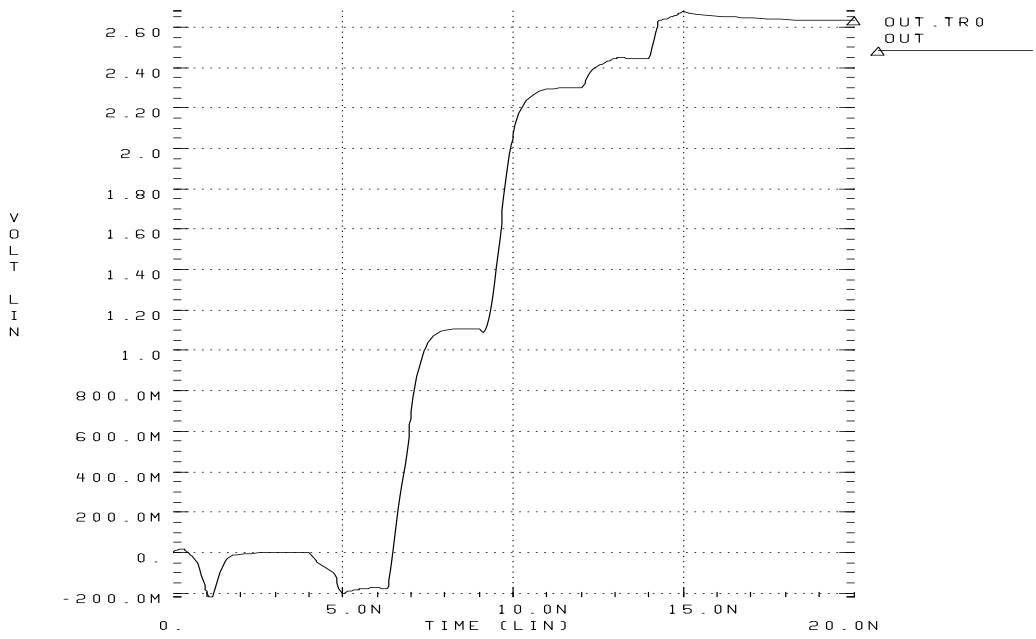

Figure 2.3: Test invalidation by Miller feedback, charge sharing, and Miller feedthrough

2.1 Miller Feedback Effect

We now show that the voltage changes on the drain/source terminals of the Miller feedback capacitances can significantly change the voltage of a floating node. We want to emphasize that a Miller feedback capacitance is not only due to the overlap between the gate and diffusion regions of a transistor, but it is also due to the charge stored in the channel region, and it can go up to half of the total gate capacitance when the transistor is on. For the pMOS transistor connected to *out* in the NOR gate in Figure 2.2, the Miller feedback capacitance changes from 4.1fF to 20.8fF according to HSPICE when the transistor gate voltage changes from 5V to 0V with drain and source voltages held at 5V.

Consider the proposed test shown in Figure 2.2. Table 2.1 shows the simulated behavior of all the cell input signals in time frame 2 and in part of time frame 1. We assume that the circuit in Figure 2.2 is embedded in a larger circuit, and the cell inputs are not the primary inputs. The first transition in time frame 2 happens at line b making the OAI31 output floating with a slightly negative initial voltage as shown in Figure 2.3. The next transition is at x between 6ns and 7ns. Just before this transition, the NOR output m was at 0V, and the internal node $p3$ in the NOR gate was at around 1.2V, which is about the minimum voltage an internal p-diffusion node can acquire in the process we used. After x becomes 0V turning on the pMOS transistor it is connected to, $p3$ and m both rise to around 5V. These rising transitions on $p3$ and m raise the *out* voltage due to Miller feedback to 1.1V from 6ns to 9ns as shown in Figure 2.3.

In time frame 1 we started x at 0V in order to first charge up $p3$ to 5V, and then let it drain down to 1.2V at the time b becomes high impedance.

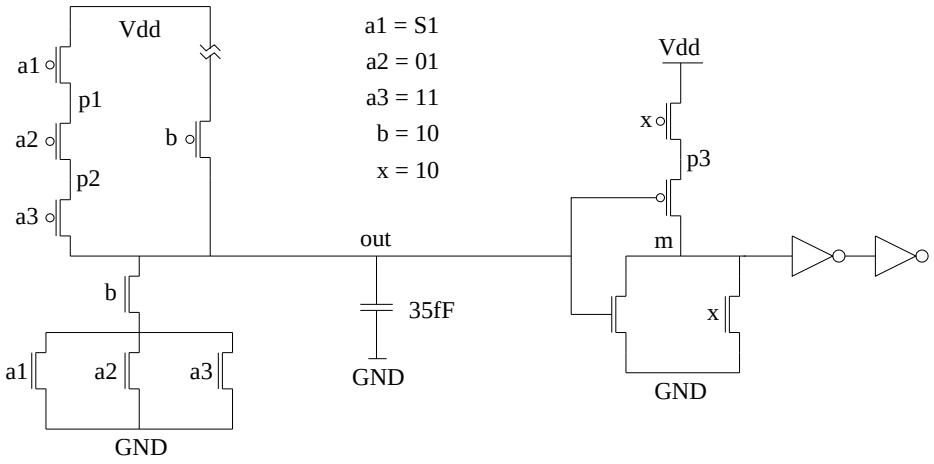

Figure 2.2: The circuit to demonstrate test invalidation for a network break

Other researchers studied the effect of transient paths to Vdd or GND on test invalidation extensively [14], [9], [19], [5], and we now show this effect with an example. Consider the p-network break in Figure 2.2. The cell input assignments shown form a proposed test for this break. Time frame 1 initializes line *out* to 0V, and time frame 2 attempts to charge up *out* to Vdd only through the broken path. In this test, if $a1$ was 11 instead of S1, then $a1$, $a2$, and $a3$ could be logic-0 at the same time momentarily due to glitches on $a1$ and $a3$ after *out* starts floating with b at logic-0. This would momentarily establish a conducting path from Vdd to *out*, and could raise the *out* voltage to a logic-1 value, thus invalidating the test.

In this paper, our emphasis is on how Miller feedback and feedthrough effects, and charge sharing can invalidate a test. We use the circuit in Figure 2.2 to demonstrate these test invalidation mechanisms. The cell on the left in Figure 2.2 with a p-network break in it is an OAI31 in the MCNC cell library, and the cell on the right is a 2-input NOR gate again from the MCNC cell library. We used HSPICE to simulate this circuit. We used level 13 (the BSIM model) in HSPICE, because this model guarantees charge conservation. We obtained the BSIM model parameters from MOSIS for the 1.2μ Orbit n-well fabrication process. The 35fF capacitance shown in Figure 2.2 is used to model a metal-1 wire that is around 160μ long in this 1.2μ process.

	part of time frame 1		time frame 2									
	initializing $p1$, $p2$, $p3$		out starts floating		Miller feedback		charge sharing		Miller feedthrough			
	0ns	1ns	4ns	5ns	6ns	7ns	9ns	10ns	12ns	13ns	14ns	15ns
x	**0V**	**5V**	5V	5V	**5V**	**0V**	0V	0V	0V	0V	0V	0V
$a1$	**0V**	**5V**	5V	5V	5V	5V	5V	5V	5V	5V	5V	5V
$a2$	0V	0V	0V	0V	0V	0V	0V	0V	**0V**	**5V**	5V	5V
$a3$	5V	5V	5V	5V	5V	5V	**5V**	**0V**	0V	0V	**0V**	**5V**
b	5V	5V	**5V**	**0V**	0V	0V	0V	0V	0V	0V	0V	0V

Table 2.1: The simulated behavior of the cell input signals in Figure 2.2

2. Detection of Network Breaks

To guarantee the detection of a network break with voltage measurements, a two-vector test is necessary. Without loss of generality, let us assume that the break is in the p-network. Then, the first vector should initialize the cell output to GND, and the second vector should activate only the broken paths in the p-network and no other path. **Activating a path** means applying ON voltages to the gates of all the transistors on the path. The second vector will make the fault-free cell output Vdd, but the faulty cell output will be high impedance with GND as its initial voltage. If the faulty cell output keeps its logic 0 value until the circuit outputs are sampled, and the second vector is a test for the cell output stuck-at-0 fault, then the network break will be detected. If certain mechanisms, which can raise the high-impedance cell output voltage from GND to a higher value that might be interpreted as logic 1, are not taken into account, a two-vector sequence may be incorrectly classified as a test for the break.

Two mechanisms that may invalidate a test, *transient paths to Vdd or GND* and *charge sharing*, have been studied in the context of transistor stuck-open faults and CMOS opens by many researchers [14], [9], [19], [2], [5], [10], [3]. In this paper, we show that the gate-drain or the gate-source capacitances of the CMOS transistors can modify the voltage of the faulty cell output when it is at high impedance. We refer to these capacitances as **Miller feedthrough** [13] when they are inside the faulty cell, and as **Miller feedback** [13] when they are inside the fanout cells of the faulty cell. We describe these mechanisms in detail, and show how our fault simulator handles them efficiently and accurately using a charge-based, instead of a capacitance-based, approach that solves all of the Miller feedthrough, Miller feedback, and the charge sharing problems together.

We now introduce some terminology that will be used in the rest of the paper. Using the path-delay fault testing terminology, let **time-frame 1** denote the time interval beginning with the application of the first vector and ending with the application of the second vector, and let **time-frame 2** begin with the application of the second vector and end with the sampling of the circuit outputs. We assume that all the signals in the circuit will be stable by the end of time-frames 1 and 2.

We use an **eleven-value logic algebra** to denote the logic values of wires in the two time frames. Let ab denote one of the nine values of our logic algebra, where $a, b \in \{0, 1, X\}$, and a and b are the final values of a wire in time frame 1 and 2, respectively. Thus, 00 on wire l means that the final value of l is 0 in both time frames. Due to multiple paths from circuit inputs to line l, the value on l may temporarily change to 1 and change back to 0 again, which is called a *static hazard* in logic design terminology. As the other two values of our eleven-value logic algebra, we use **S0** to represent a 00 with no static hazard, and **S1** to represent a 11 with no static hazard, and refer to them as stable 0 and stable 1 [18], respectively. Figure 2.1 shows two cases for an output assignment of 00 and S0 for an AND gate.

Figure 2.1: An AND gate output with and without a static hazard

3. We identify static hazards on the circuit wires, and this enables us, among other things, to determine whether a faulty cell internal node has an intermittent or a stable connection to the cell output during charge sharing. This makes a difference, because the resulting voltage when a group of capacitors are sharing charge at the same time is different from the case where the same group of capacitors connect with each other in a certain sequence but not at the same time. This also makes a difference for the worst case Miller feedthrough effects as shown in Section 3.2. The static hazard information is crucial in determining whether transient paths to Vdd or GND can occur, and also crucial in determining Miller effects.
4. Our fault simulation performance degradation is very small since we use only six voltage levels to compute the worst case charge differences, as described in Section 3.2, so the charge equations can be precomputed into a look-up table. We also perform parallel pattern simulation at the gate level identifying static hazards. Section 4 shows that our CPU times are very competitive with previous less accurate fault simulation methods. Using more look-up tables will improve performance further.
5. We use realistic network breaks that are identified from the layouts of the cells by an inductive fault analysis tool, *Carafe* [8] [16]. A realistic network break may correspond to multiple network breaks identified from the transistor schematic of a cell, one example is a defect that breaks a diffusion contact in the layout where two transistors were connected to Vdd or GND using the same broken diffusion contact.

1. Introduction

Defects that occur during the IC manufacturing process can be categorized into three classes according to Hawkins et al. [7]. These classes are bridge, open circuit, and parametric defects. Open circuit defects cause breaks in the conducting materials in the layout, and contacts are particularly susceptible to such breaks. Breaks can be divided into two categories: those that physically disconnect one or more transistor gates from their drivers, and those that disconnect transistors from each other in the p-network or n-network of a CMOS cell [11]. We define a **network break** as a break fault in the p-network or in the n-network of a cell that breaks one or more transistor paths between the cell output and Vdd or GND. A **transistor path** is a sequence of transistors physically connected through their drain and source terminals. Note that transistor stuck-open faults form a subset of network break faults. Renovell and Cambon [15], and Champac et al. [1] showed that a transistor stuck-open test set can detect some of the breaks that create floating transistor gates. So, a network break test set is useful not only for detecting network breaks but also other breaks that cause floating transistor gates.

Detection of a network break with voltage measurements requires a two-vector test. Reddy et al. [14] showed that transient paths to Vdd or GND can invalidate a two-vector test in transistor stuck-open testing, and Barzilai et al. showed that charge sharing between the internal nodes of the faulty cell and the high impedance faulty cell output can also invalidate a test. Lee and Breuer [10] proposed a scheme for handling charge sharing in transistor stuck-open fault testing using both IDDQ and voltage measurements, but measuring both current and voltage may not be feasible during testing. Barzilai et al. [5] described a fault simulator for transistor stuck-open and stuck-on faults. For handling charge sharing, they partitioned all the nodes in every cell into two classes. Nodes in the first class were assumed to have small enough capacitances so that they could be ignored. If a node in the second class can share charge with the floating cell output, then the test is declared invalidated. Di and Jess [3] developed a fault simulator for network breaks, but they ignored static hazards, and their detecting conditions considered charge sharing only with the nodes on the broken paths. Favalli et al. [6] proposed a set of detection conditions for network breaks, but they considered neither transient paths to Vdd or GND, nor charge sharing.

In this paper we present a fault simulation algorithm for network breaks that takes into account the transient paths to Vdd or GND, charge sharing, Miller feedback effect, and the Miller feedthrough effect. The major contributions of this paper are as follows:

1. We demonstrate in Section 2 that Miller feedback and Miller feedthrough capacitances can invalidate a two-vector test for a network break just as charge sharing can. To the best of our knowledge, there is no published work that handles the effects of Miller capacitances on test invalidation in network break or transistor stuck-open fault simulation. We describe a charge-based approach in Section 3.1 that considers the worst case effects of Miller capacitances and charge sharing together on test invalidation.
2. Because we have a charge-based approach, the non-linear nature of Miller capacitances and internal node capacitances are accurately modeled compared to previous capacitance-based approaches. In Sections 2.1 and 2.2, we show that a Miller capacitance and an internal node capacitance can change by more than a factor of five and a factor of two, respectively.

Accurate and Efficient Fault Simulation of Realistic CMOS Network Breaks

Haluk Konuk
F. Joel Ferguson

UCSC-CRL-94-40
18 October 1994

Board of Studies in Computer Engineering
University of California, Santa Cruz
Santa Cruz, CA 95064

ABSTRACT

We present a new fault simulation algorithm for realistic break faults in the p-networks and n-networks of static CMOS cells. We show that Miller effects can invalidate a test just as charge sharing can, and we present a new charge-based approach that efficiently and accurately predicts the worst case effects of Miller capacitances and charge sharing together. Results on running our fault simulator on ISCAS85 benchmark circuits are provided.