Measuring dynamic input capacitance of CMOS logic gates

G. Dalton Bentley

https://stackexchange.com/users/7947131/dalton-bentley

(Dated: December 27, 2021)

Abstract

As a retired electronic engineer, my attention was attracted by a question posted online at Stack Exchange, How find input capacitance output R CMOS use SPICE, asking how to measure the input capacitance (and output resistance) of a CMOS NAND gate using SPICE (or descendants of that electronic circuit simulation software). This paper is a detailed discussion of the subject and includes `Ngspice-27` code, as well as discussion of standard cell libraries, CMOS layout, appropriate test frequencies, and analysis and repair of the CBCM (charge-based capacitance measurement method).

CONTENTS

I. INTRODUCTION

We happened upon the following question posted online at the electronics forum at Stack Exchange[1] How find input capacitance output R CMOS use SPICE:

> I have a 2-input NAND gate spice netlist (generated from a Tanner Ledit layout) where I have to find each input's capacitance and the output resistance. I am to use a 1nF load capacitor and a 10 Kohm for the calculations.

The OP (original posting) defined the circuit netlist as

```
* 1 = Vdd, 2 = Gnd. 3 = Nand_out (F), 4 = A (NAND in), 5 = B (NAND in)
M3 1 4 3 1 PMOS L=2u W=10u AD=80p PD=38u AS=110p PS=62u
* M3 DRAIN GATE SOURCE BULK (30.5 27 32.5 37)
M4 3 5 1 1 PMOS L=2u W=10u AD=110p PD=62u AS=80p PS=38u
* M4 DRAIN GATE SOURCE BULK (39.5 27 41.5 37)


M1 6 4 3 2 NMOS L=2u W=10u AD=70p PD=34u AS=55p PS=31u
* M1 DRAIN GATE SOURCE BULK (30.5 5 32.5 15)
M2 2 5 6 2 NMOS L=2u W=10u AD=70p PD=36u AS=70p PS=34u
* M2 DRAIN GATE SOURCE BULK (39.5 5 41.5 15)
```

We corrected the wiring for the MOSFET devices in the netlist above consistent with normal electronics practice:

[1] Building on the popularity of the initial Stack Overflow question-and-answer website for programmers, Stack Exchange now offers sites dedicated to various fields, e.g., physics (where at least one Nobel Prize winner contributed for a time), mathematics, and electronics, the latter the context here. Though losing some of the most prolific participants in late 2019 after new management annoyed many, the site still remains a source of high-quality information, subject to the same critical evaluation one should apply to any information.

```
Circuit nodes: 1=vdd, 2=gnd, 3=F Nand out,4=A Nand in,5=B Nand in
Order of MOSFET leads: drain, gate, source, bulk/well
electronically correct M1 6 5 2 2 NMOS cf. OP: M1 6 4 3 2 NMOS (d/s is reversed)
electronically correct M2 3 4 6 2 NMOS cf. OP: M2 2 5 6 2 NMOS (d/s is reversed)
electronically correct M3 3 4 1 1 PMOS cf. OP: M3 1 4 3 1 PMOS (d/s is reversed)
electronically correct M4 3 5 1 1 PMOS cf. OP M4 3 5 1 1 PMOS
```

Using the posted node numbers, but with the electrically correct connections, we created a schematic for that layout in Fig. 1:

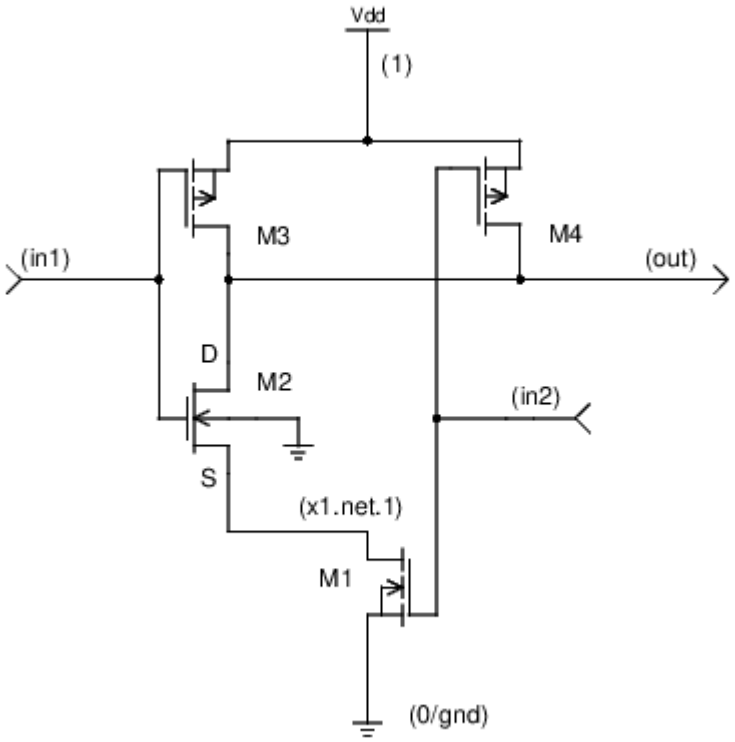

FIG. 1. The correct schematic for the posted NAND layout (using the OP node labels). M2 is an enhancement-mode NMOS transistor like M1 (required ability to indicate bulk connection to ground rather than source and symbol choices were limited).

Our goal in this paper will be to answer the question (albeit using a different CMOS process, but more on that in Section IV), in considerably more detail than would be possible in a formal answer post at Stack Exchange, with suitable adjustment of the load capacitance on the NAND output (and discarding the 10 kΩ as irrelevant). We will provide technical

background and Ngspice-27 circuit code, among other things we used in the simulations and methods developed. We consider many related matters of interest (to us and possibly to you), but more on that below.

II. ORGANIZATION OF THIS PAPER

We will begin in Section III with a brief discussion of MOSFET drain-source symmetry (because the posted circuit configuration apparently assumed that) and identify our terms NMOS and PMOS for our purposes here.

In Section IV we will set out the reasons for our selection of CMOS process and PDK (physical design kit) to use in the simulations. That will include a brief look at CMOS integrated circuit layout to provide some context for our work. We specify SPICE model cards and standard library cells used also.

We discuss the relevant equations for capacitance in Section V and follow with the details of the CBCM[1] or charge-based capacitance measurement method in Section VI.

Our choice of drive impedance and test signal frequency is described in Section VII and Section VIII. The latter section discusses the relevant standard cell library parameters and several methods of estimating bandwidth and operating frequency for amplifiers generally, as well as the MOSFET and CMOS in particular.

In Section IX we run simulations in Ngspice-27 using our implementation of the original CBCM method, looking at the accuracy in measurement of a discrete capacitor of 1.5 fF before continuing to the measurement of NAND2_X1 input capacitance. We follow in Section X with an analysis of why the accuracy was not as good as expected in the initial validation run, finding that there was a known problem with the method causing a small error.

In Section XI we propose our own solution (a software fix more or less) to the problem discovered in the original implementation of CBCM. We include an alternate strategy involving user participation rather than solely automated measurement.

We make several simulation runs with the revised CBCM method in Section XII, measuring the discrete validation capacitor first (improving accuracy to better than 1%), then following with measurements of the in1 input capacitance of the NAND2_X1 gate at 1 GHz, 100 MHz and 2 GHz achieving correspondence of 4-8% with the Nangate library suggested input capacitance value of 1.599 fF.

We include a non-CBCM method of CMOS gate input capacitance measurement in Section XIII, for those who might prefer a solution not involving multiple CMOS circuits (and can accept the deterioration in accuracy).

Though not strictly concerned with capacitance measurement (except perhaps as a check on the operation of the circuits used in those measurements), in Section XIV we measure the propagation delay observed in our simulation of the NAND2_X1 gate, comparing with the Nangate library suggested values.

We estimate the output resistance of the NAND2_X1 gate in Section XV, using two different methods for cross-validation.

Appendices include a brief examination of the BSIM4 MOSFET capacitance model (Appendix A), with subsections looking at the intrinsic core model (presenting the Ward-Dutton transcapacitance matrix), the BSIM4 RF (high-speed) settings we used, and for the curious, a look under the hood as it were, graphing some of the internal BSIM4 capacitance and charge variables (which hopefully lend some reality to the abstractions of the model). Ngspice-27 circuit code used in the simulations is given in remaining appendices. The code is well-commented with a combination of explicit comment and literate programming, i.e., naming variables in such a way as to identify their purpose, so it is hoped that the reader can quickly adapt the methods to whatever SPICE environment is available.

We also include in the appendices the BSIM4 SPICE model cards for the PMOS and NMOS transistors we used in the Ngspice simulations. Finally, for those who would like to read more about MOSFET and CMOS, we include a non-exhaustive list of textbooks and lecture notes we used in Appendix I. We tried to include a url when we had one available for all references cited also. Hopefully the reader has obtained a hypertext active pdf of this paper, which will facilitate quick cross-referencing of ideas encountered (along with text search within the pdf reader).

III. MOSFET DRAIN-SOURCE SYMMETRY

From[2] we find that the MOSFET (metal-oxide-semiconductor field effect transistor) is a physically symmetric device and that in the BSIM4[2] (based on BSIM3) device model the

[2] The BSIM3 and BSIM4 or Berkeley Short-Channel IGFET Models are industry standards for the simulation of CMOS processes down to 0.15 μm (BSIM3) and below (BSIM4).[3]

drain to source current (I_{ds}) and the terminal charges should remain the same when the source terminal and the drain terminal are swapped, either by virtue of properties of the core physical model and/or by internal swapping mechanisms of the model. It is therefore probably not an error to wire the MOSFET without particular regard to the normal drain and source assignments as we see in the original L-edit[3] extracted circuit posted and in some of the standard cell library circuits. Nevertheless, we will restore normal electronic schematic wiring conventions for PMOS and NMOS drain and source leads in our discussion, simulations and schematics.

By NMOS we mean the integrated circuit MOSFET with n^+-doped drain and source regions embedded in a p-type substrate, where gate to source voltage $V_{GS} > 0$ induces an effective n-channel bridging source and drain such that current of electrons I_{ds} can flow if voltage between drain and source V_{ds} is greater than zero.[4] Farther below we present a figure depicting the NMOS device cross section in simplified form Fig. 11. The PMOS transistor (we follow Sah and use the term "transistor" as a "generic name for solid-state electron devices with three or more terminals"[5] [4]) can loosely be considered to invert the dopings (p^+ source and drain embedded in n-type well) and voltages of the NMOS, i.e., source is the more positive terminal, $V_{gs} < 0$ turns on the device, current primarily holes.[6]

IV. SELECT CMOS PROCESS MODEL

If the NMOS transistors in the NAND circuit are in minimum L (channel length) configuration in the posted question, i.e., L=2u where u denotes μm, as they should be[5] (leaving aside questions about equalizing transition times in relation to other circuits or the like[8]), that implies the minimum feature size is a relatively huge $2\,\mu$m. That large a process has not been used in integrated circuit design for thirty years or more. For example, the largest gate length in a 1988 table of high speed logic devices was $1\,\mu$m[5]). To put this feature size in perspective, as of 2021 Intel is already shipping 10 nm FinFET processors, about as small as technologies recognizable in our context here can get.[9]

Being unable to locate parameter sets (model cards) applicable to the $2\,\mu$m channel

[3] Simply put, Tanner EDA L-edit, now Siemens EDA L-edit, is a tool for drawing the two-dimensional geometries (aka "pushing polygons") destined for a semiconductor mask.

[4] The Bell Lab definition required that the device also have power gain greater than unity.[5]

[5] The current drive strength of a FET is inversely proportional to channel length so as a general integrated circuit sizing rule the minimum channel length is used.[7]

length and hesitant to use defaults on some of the oldest MOSFET simulation models, we decided, therefore, to use a process size for which we could obtain reliable standard cell model parameters, circuit configurations, and SPICE model characterizations, specifically, Beta Version released on 2/22/06 45nm BSIM4 model card for bulk CMOS: V1.0[10] and INV_X1 and NAND2_X1 from NCSU FreePDK 45nm[11]. FreePDK45 is the NCSU (North Carolina State University) kit for a generic 45 nm process.[12] . This is not cutting-edge technology (early versions of the 45 nm kit were released c. 2002)[6], but it was available. With some effort we managed to locate files required to implement our CMOS circuits, but even this old 45nm FreePDK is increasingly difficult to obtain without formal member organization or academic affiliation.[14]. After some initial testing we updated our model cards to the nominal corner (models_nom) (a corner is "a characterization of the standard cell library and technology with specific assumptions about the process, temperature, and voltage or PVT"[15]), i.e., HSPICE cards for NMOS_VTL and PMOS_VTL built for FreePDK 45nm verion 1.4 (2011-04-07) Subversion Repository revision 173 .[16]

The NMOS channel length in a FreePDK45 45 nm NAND2_X1 is L=0.05u, i.e., it is roughly the minimum for the 45 nm process. "NAND2_X1" is terminology in standard CMOS cell libraries meaning "a 2-input NAND gate, NAND2, with drive strength 1 transistor sizing X1."

A. Brief view of CMOS layout

Though our work in this document is oriented towards SPICE simulation of a CMOS gate input capacitance and we will not be considering layout in detail, perhaps it would be helpful to the reader in the context of discussion of gate width W and length L to see a screen shot of a standard cell library NAND2_X1 CMOS circuit we opened in a layout editor (LayoutEditor[17]) Fig. 2:

Recall from Section I above that the original question about measuring input capacitance that sparked our interest in the subject was in reference to a circuit encountered in a CMOS layout session with the layout editor, L-edit. Most layout editors will generate a schematic

[6] There is a more advanced 15nm library based on the FreePDK15 process design kit from NC State University used in conjunction with the Predictive Technology Model from Arizona State University[13], but you must be a member of the Silicon Integration Initiative, Inc. or associated with a university to gain access.[14]

from a layout and facilitate simulation with several SPICE implementations. However, we hand-coded Ngspice "cards" (the lines of SPICE code in a circuit file, e.g., Appendix B [7]) and ran Ngspice circuit simulations from a Linux terminal session not associated with a layout editor. We used the open source schematic editor *gschem*[20] to manually draw electronic schematics Fig. 1 and Fig. 5.

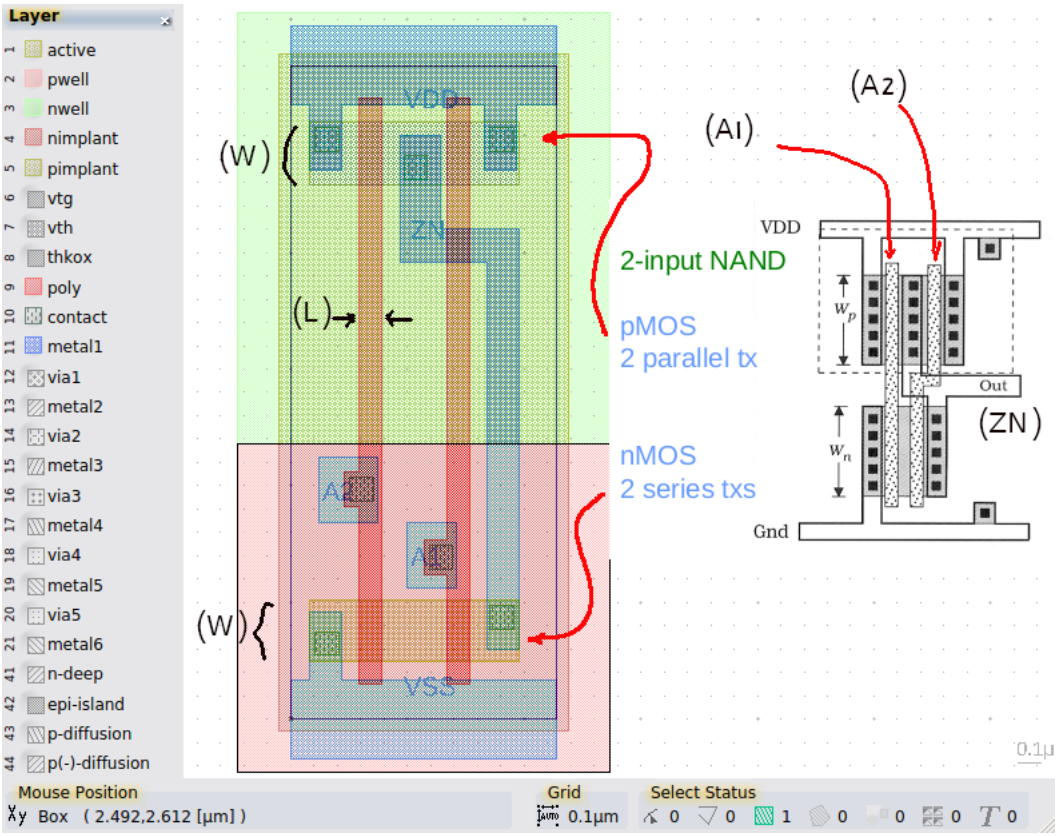

FIG. 2. A standard library CMOS NAND2_X1 circuit is open in LayoutEditor[17]. The additional markings and smaller inset figure[21] were added by the author.

To make a more direct connection of the above layout Fig. 2 with the NAND2_X1 electronic schematic, we offer Fig. 3.

Finally, one more figure presents detail of a MOSFET transistor (NMOS), comparing cross section in silicon to a view of the corresponding typical view in a layout editor: Fig. 4

For those who would like to read more about MOSFETs and CMOS and/or the layout of CMOS integrated circuits, see Appendix I.

[7] In 1973 when SPICE1 emerged, much of computer input was provided via punch (Hollerith) cards. SPICE element instance and control line lines were batch submitted with decks of these cards.[18]][19]

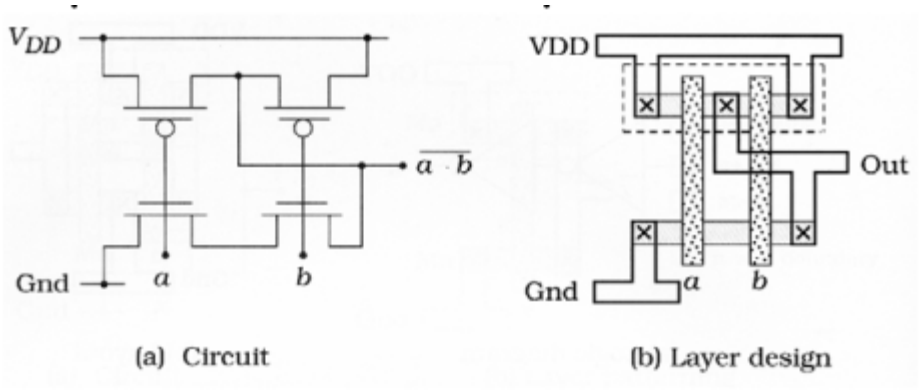

FIG. 3. A NAND2_X1 is shown in schematic at left (a) (compare our earlier schematic Fig. 1) and at right (b) in superior view of the corresponding (simplified) CMOS horizontal transistor layout.[21] The width W, i.e., poly gate and connecting strip, runs vertically as in Fig. 2.

B. Sizing continued

The NAND2_X2 would therefore have increased current drive (more transistors while arranged to keep the resistance about the same), but higher input capacitance, the inputs now seeing more parallel gates. Recall that, as a general matter, resistance is inversely proportional to the cross-sectional area, $A = \text{Height} \cdot \text{Width}$, of a rectangular conducting region, $\text{Resistance} = \text{Length}/(\sigma A)$, with σ being the conductivity of the material (in turn a function of charge density ρ and carrier mobility μ).[22] Capacitance is directly proportional to the area of the planes separated by dielectric[22]. The electrical effort required of logic gates to drive other gates is then usually expressed a ratio of transistor widths, where it is assumed that they have the same minimum length[23], although in our specific case we see parallel devices used to accomplish effective increase of width and decrease of effective length. The sizes of the transistors in standard cells may be optimized in application for delay minimization and fall/rise time matching, among other performance goals.[8]

C. Device model cards utilized

We used the VTL 45nm BSIM4 model card for bulk CMOS because our standard library cells used NMOS_VTL and PMOS_VTL. "VTL" identifies SPICE model characterization for low threshold voltage (V_t), high-performance devices in this PDK.[12]

Our copy of the documents[11] for this CMOS design kit included the stdcells-databook.pdf

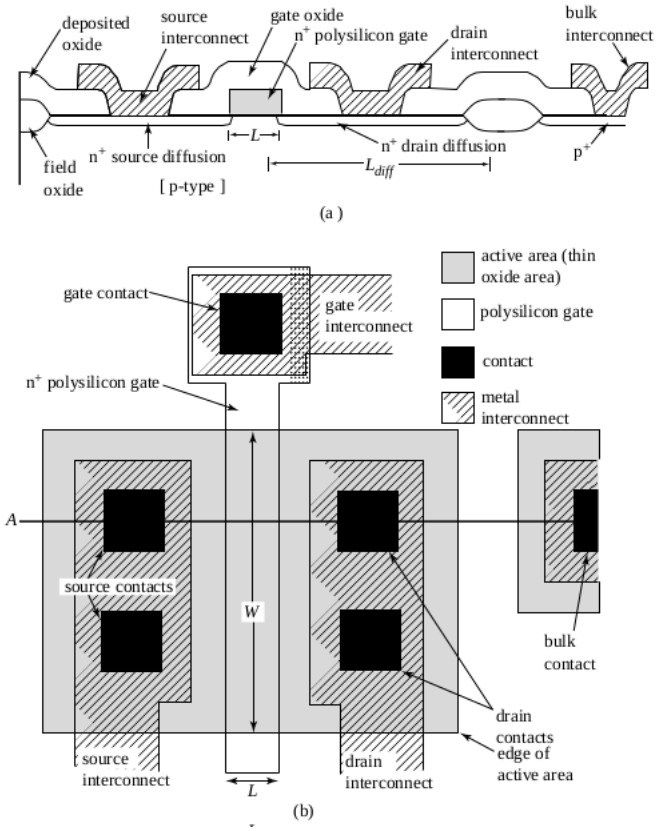

FIG. 4. Shown is an NMOS (n-type MOSFET) in silicon cross section at the top and in superior layout view below. From a pdf file "EECS 6.012 Spring 1998 Lecture 13" at the Electrical Engineering and Computer Sciences department of the University of California, Berkeley. Compare to any of the four transistors in Fig. 2, being mindful of the difference between NMOS and PMOS drain and source diffusions (and PMOS n^+ well vs the p^+ bulk of the NMOS).

automatically generated when Nangate built this library, NangateOpenCellLibrary. Nangate sells tools (e.g., Library Creator) to automatically generate standard cell libraries, but as a demonstration of their product, made freely available this particular library generated from the academic-purposed FreePDK45 Physical Design Kit produced by NCSU[24] we mentioned above.

In the hope of making it possible to reproduce our work, we provide a copy of the NMOS_VTL and PMOS_VTL model cards in Appendix G and Appendix H, respectively. These model cards may also be downloaded from our GitHub account at Author GitHub Repository.

The technique we will use in the analysis of capacitance seen by a driver circuit at the

input to the CMOS NAND gate should be applicable to any process devices, so if you have model cards for very old process devices, you should be able to use the method we present, *mutatis mutandis*, i.e., adjusting the power supply voltage, signal rise/fall times and frequency, etc. as appropriate for the process size.

V. CAPACITANCE

Our measurement of CMOS logic gate input capacitance seen by a driving circuit will utilize observed dynamic current and voltage relations. A good point of departure is the discussion of capacitance by Steve C. Cripps in his textbook on radio frequency amplifiers for wireless communications[25]. We modify the electrostatics definition of capacitance, electric charge Q in Coulombs, V in volts, C in Farads:

$$C = \frac{Q}{V} \tag{1}$$

to a differential form wrt (with respect to) voltage,

$$C = \frac{dQ}{dV} \tag{2}$$

The instantaneous current, i.e., the rate of change of electric charge, in any capacitor, $i(t)$, is given by

$$i(t) = \frac{\partial q(t)}{\partial t} \tag{3}$$

Because $q = Cv$ (from Eq. 1), it would therefore seem reasonable to substitute Cv for q in Eq. 3 and write:

$$i(t) = \frac{\partial}{\partial t}\left(Cv\right) \tag{4}$$

If C is defined to be the differential form from Eq. 2, then C can be written as a function of the partial variation of charge wrt voltage:

$$C(v) = \frac{\partial q(v)}{\partial v} \tag{5}$$

And we may obtain

$$\begin{aligned} i(t) = \frac{\partial q}{\partial t} &= \underbrace{\frac{\partial q}{\partial v}}_{C(v)} \underbrace{\frac{\partial v}{\partial t}}_{dv/dt} \\ &= C(v)\frac{\partial v}{\partial t} \end{aligned} \tag{6}$$

So we could write, for example,

$$C(v) = i \,/\, \left(\frac{dv}{dt}\right) \tag{7}$$

and go about observing the quantities on the rhs (right-hand-side) of Eq. 7 to obtain the lhs (left-hand-side), i.e., the effective input capacitance as a function of the ratio of the current to the rate of change of the applied voltage wrt time.[8] See also Hayt's[22] use of Eq. 5 incidental to illustrating the use of Poisson's equation in characterizing the P-N junction (having obtained the potential across the junction, it appeared the solution could be used to find the junction capacitance).

VI. CBCM, THEORY

CBCM (Charge-Based Capacitance Measurement) is a method for measuring the ultra-small capacitances encountered in submicrometer interconnects in CMOS technology.[1] The authors of this article[9][10] realized that for a known voltage, we need only be able to measure charge Q to determine any capacitance of interest, relying on the equation of a linear capacitor $Q = CV$ (see Eq. 1).

By focusing on the difference in average current, $I_\Delta = I_{ref} - I_{test}$, drawn from separate power supplies V_{dd} by two identical CMOS process inverters (as would be measured by a DC ammeter interposed between the circuit and each V_{dd} supply) over a given time interval Δt where the test circuit drives the unknown capacitance C_{test}, the amount of charge $Q_{(C\,test)}$ is directly measured:

$$\begin{aligned}
I_\Delta &= I_{ref} - I_{test} && \text{(these are average } V_{dd} \text{ currents measured)} \\
Q_{(C\,test)} &= I_\Delta \Delta t && \text{(by Eq. 3)} \\
Q_{(C\,test)} &= C_{test} V_{dd} && \text{(by Eq. 1)} \\
I_\Delta \Delta t &= C_{test} V_{dd} && \text{(by transitive property of equality)} \\
C_{test} &= \frac{I_\Delta\, \Delta t}{V_{dd}} && (\Delta t \text{ is the period of the applied pulse train.})
\end{aligned} \tag{8}$$

[8] If you decide to use Δv and i(RMS) or i(avg), be sure to multiply the current by Δt of the interval used.

[9] One of the authors, Chenming Hu, was instrumental in the development of the BSIM3v3 and BSIM4v4.8.0 MOSFET SPICE models for circuit simulation and CMOS technology development.

[10] The authors apparently first published a description of CBCM in 1996, in collaboration with J. C. Chen and B. McGaughy.[26]

The schematic for the above scenario is Fig. 5. Notice that the NMOS transistors in the two inverters receive a delayed, slightly smaller pulse width (but at the same frequency) than the PMOS devices in order to eliminate CMOS transition current spikes[27] while the PUN (pullup network) and PDN (pulldown network) are normally briefly both on during a transition.

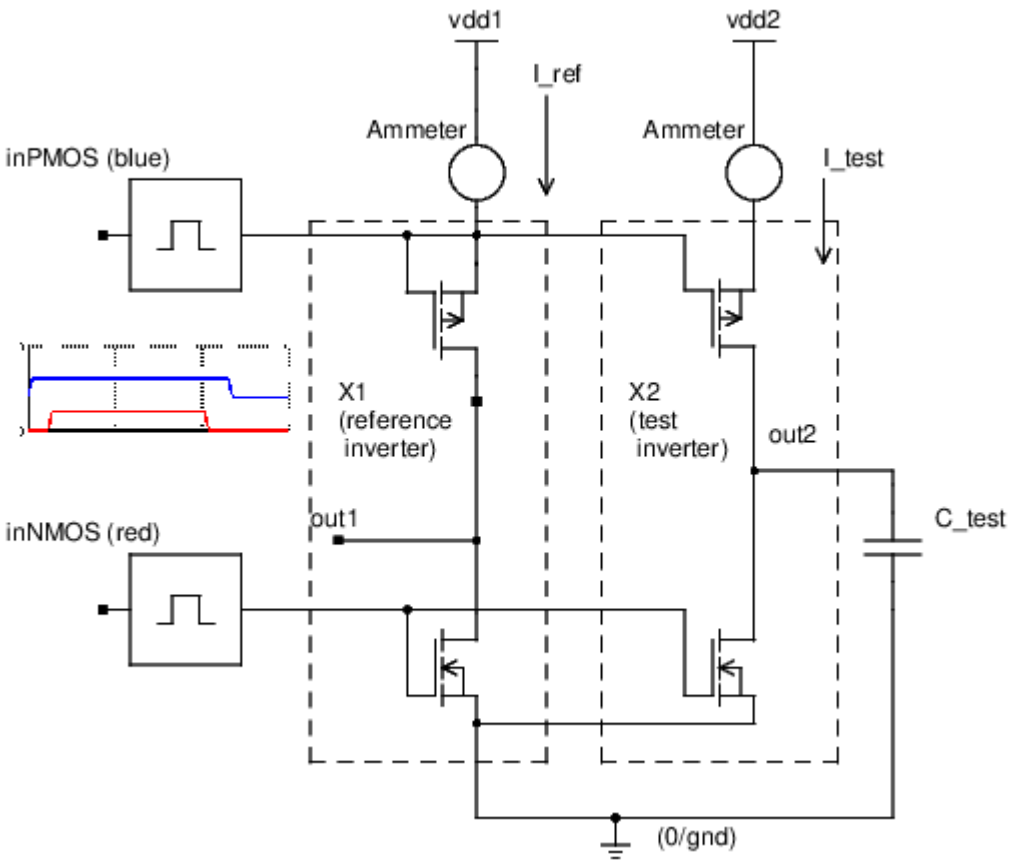

FIG. 5. The schematic for the two inverter CBCM (Charge-Based Capacitance Measurement) method is shown. See text for discussion.

When the PMOS transistors turn on (when the PMOS input signal inPMOS at logic HIGH V_{dd} returns to logic LOW gnd, where NMOS were already turned off with their input signal inNMOS already grounded, see pulse diagram on Fig. 5), they begin to charge intrinsic circuit (both test and reference) and load capacitances (for the test circuit alone) to V_{dd}. Given enough time, the PMOS transistors will have completely charged all capacitances and their current will return to zero before the next input transition. After the PMOS have turned off following their input signal going HIGH V_{dd} again, the NMOS transistors are

turned on when their input pulse goes HIGH, providing a discharge path to ground for the output capacitances seen by the inverters.

As we alluded to above, the objective of this drive configuration is to assure that all current being drawn and measured from each power supply is being used by the PMOS devices to charge load capacitance (intrinsic for the reference inverter and intrinsic plus load for the test inverter), excluding the usual CMOS $\sim$ 10% of total current switching transient short-circuit current[11] (which flows from V_{dd} to ground down through the open PMOS NMOS series path).

The shape of the PMOS device current waveform is not significant in this measurement technique. Only the total current delivered is of interest and its value can be determined with any dc ammeter, or in our case, averaging the SPICE (Ngspice-27 to be precise) current probe (the array of values generated by a simulation run, also known as the current vector in SPICE terminology, where "vector" here means a sequence of data accessible by one or more indices) on each circuit V_{dd} path.

We should note that the authors of the CBCM method measured the same target capacitance (on-chip interconnect capacitances) repeatedly while varying the frequency. They then plot the net current difference between loaded and unloaded inverters vs frequency (which they assume will be a straight line). The slope of that plot is equal to the product of the target capacitance and the supply voltage. The capacitance can then be extracted from that product. Alternately, the frequency could be kept constant and the supply voltage varied.[1] We did not consider this procedure to be necessary in our context, as we will specify a proper test frequency to apply to normal operation of the selected CMOS process and standard library (in Section VIII) and we do not expect the input capacitance of a CMOS gate to necessarily be a linear function of supply voltage or frequency (though we do find very little variation between 100 MHz and 1 GHz in Section XII).

You may measure a discrete capacitor on the output of the test inverter X2 in the circuit shown in Fig. 5) to verify proper performance of the circuit for example. To measure the input capacitance of a CMOS gate, replace C_test with a wire (connection) to the input of the circuit of interest. In our own context, that will eventually be the A1 input of the NAND2_X1, in1, referring to Fig. 1. If the NAND2_X1 circuit (or other CMOS circuit of

[11] "...when switching [CMOS] from one state to another, the input crosses the threshold region, causing the N-channel and the P-channel to turn on simultaneously, generating a current path between V_{CC} and GND."[27]

interest) is added to the circuit netlist file for simulation, a third power supply independent of the two used for the inverter circuits should be added.

VII. APPROPRIATE INPUT DRIVE IMPEDANCE

Many semiconductor vendors specify timing information based on a test setup using a $50\,\Omega$ output impedance pulse generator[28], the intent being to provide information using standard test equipment impedance, which should be low to approximate the voltage source character of a CMOS output[12]. In the context of our Ngspice simulations, we are driving the CBCM inverter stages constituting the measurement apparatus (reference inverter x1 and test inverter x2) with an ideal voltage source pulse generator (two actually, recalling the two phases driving PMOS and NMOS separately in CBCM, see Section VI).

However, the target capacitance to be measured is the capacitance on the *output* of the test inverter (x2) gate. Initially, this will be a discrete validation capacitor. When the method is established as valid, the discrete capacitor will be replaced with the input to the NAND2_X1. The MOSFETs in CMOS gates are voltage-controlled switches (see §3.6 [6], or §11.5.1 [4] for example) so it would be inappropriate to drive them with a current source if the intent is to measure gate input capacitance seen by another CMOS driving circuit.

VIII. TEST FREQUENCY AND RISE/FALL TIME CONSIDERATIONS

Recall from Section IV above that we are using a 45 nm CMOS process, i.e., the nominal corner NMOS_VTL and PMOS_VTL SPICE model cards (characterization under PTM[13]) from a 2011 build of FreePDK 45nm[16], and the INV_X1 and NAND2_X1 standard cells from NCSU FreePDK 45nm[11]. We define our goal to be measuring the input capacitance of a CMOS logic gate (NAND2_X1 specifically) seen by a driving circuit (INV_X1 in our context) using a reasonable approximation to the conditions normally seen in applications of the selected CMOS process.

As an initial hint for a suitable test frequency we find August 2008 Intel archive documents listing clock speeds in the range $1.2 - 2.4\,\text{GHz}$ for 45nm process microprocessors for Mobile

[12] A CMOS driver impedance is often approximated as $1\,\text{k}\Omega$, which is almost exactly the R_{ds} of the PMOS transistor in a LOW to HIGH transition of our NAND2_X1 gate estimated in Section XV.

PCs. That being said, Intel appears to have been using high-k metal gate silicon technology in 2008.[29] We are using a bulk CMOS model card, i.e., a traditional CMOS process without the high-k metal gate adjustments, so the Intel historical data is not exactly applicable.

A. Nangate cell library databook

A more relevant source in this regard is the Nangate stdcells-databook.pdf automatically generated when Nangate built the standard cell library we are using, NangateOpenCellLibrary.[13] The stdcells-databook.pdf provides expected values for a number of circuit parameters for each type of logic gate in the library. In our context, we are interested in INV_X1 and NAND2_X1. The library data includes values for several *corners*. Recall that a corner is "a characterization of the standard cell library and technology with specific assumptions about the process, temperature, and voltage (PVT)."[15] We consulted the values for the typical build, $V_{dd} = 1.10\,\mathrm{V}$, $T_j = 25.0^\circ\,\mathrm{C}$ (other corners include fast, slow, low temperature). The input capacitance given for the NAND2_X1 gate was $\sim 1.59\,\mathrm{fF}$, hence our choice of $1.5\,\mathrm{fF}$ (testcapacitance=1.5ff) as an initial run capacitance target in Section IX farther below.

B. Estimate operating frequency using rise time

We may make use of the familiar relation between rise time t_r and signal $-3\,\mathrm{dB}$ cutoff frequency, $f_{-3\,\mathrm{dB}} = 0.35/t_r$[1]. One of the Nangate stdcells-databook.pdf applicable (i.e., for our typical corner circuit NAND2_X1 loaded with $C_L = 59.3567\,\mathrm{fF}$) test input transition signal rise/fall times is $0.1985\,\mathrm{ns}$ (we note that they also give expected results at $1.2\,\mathrm{ps}$). $t_r = 0.1985\,\mathrm{ns}$ is then roughly equivalent to a clock speed of $1.8\,\mathrm{GHz}$:

$$\begin{aligned} f_{-3\,\mathrm{dB}}\,(\text{signal bandwidth}) &= 0.35/t_{rise} \\ &= 0.35/0.1985\,\mathrm{ns} \\ &= 1.8\,\mathrm{GHz} \end{aligned} \tag{9}$$

[13] The stdcells-databook.pdf was built Feb 17 15:07 2011. The circuit description for the two CMOS logic cells we used, which we preferred as omitting the many lines of integrated circuit parasitics in file stdcells-lpe.spi, was stdcells.cdl, from a 2010 December build with NGLibraryCreator. Christopher Torng assembled these files for an ASIC/FPGA flow generator.[11]

C. Estimate maximum switch rate from propagation delay

A maximum possible (if not recommended[14]) switching frequency estimate could be had by considering the minimum pulse width to be the sum of propagation delays for output rising and output falling transitions, $f_{max} = 1/(t_{pLH} + t_{pHL})$[6]. Our Nangate stdcells-databook.pdf suggests, for the above NAND2_X1 conditions (and considering specifically NAND2_X1 input A1 to output ZN) a $t_{pLH} = 0.25\,\text{ns}$ and $t_{pHL} = 0.21\,\text{ns}$. We obtain a maximum switch rate of 2.2 GHz:

$$\begin{aligned} f_{max} &= 1/\left(t_{pLH} + t_{pHL}\right) \\ &= 1/\left(0.25\,\text{ns} + 0.21\,\text{ns}\right) \\ &= 2.2\,\text{GHz} \end{aligned} \tag{10}$$

D. Slew rate and full power bandwidth f_{fp}

For NAND2_X1 input A1 to output ZN into 59 fF load with input transition 0.1985 ns, our Nangate stdcells-databook.pdf specifies output transition of 0.13 ns (falling) and 0.15 ns (rising). They define this transition as 30% to 70% of final voltage output rising and 70% to 30% falling. If we adapt a concept from analog design, i.e., the slew rate (SR), we can use the output transition data to obtain still another estimate of proper operating frequency for our purposes. We note that Texas Instruments also uses the term SR for "the average rate of change (i.e., V/ns) for a waveform that is changing from one defined logic level to another defined logic level."[28]

The slew rate (SR), the maximum rate of change of output voltage of a circuit with respect to time, $\text{SR} = dV_{out}/dt$ is usually measured between 10% and 90% of the final value of the output, though 20% to 80% is also used.[31] We will use 30% to 70%, that interval delimiting the relevant Nangate data for our CMOS NAND2_X1 circuit. We do note that we are using the transistors as saturated switches (large-signal), not as small-signal amplifiers so we must keep this in mind when applying analog approximations.

[14] Texas Instruments warns that the maximum signaling rate is not necessarily equal to the inverse of the propagation delay. The maximum data rate on buffers depends on propagation delay matching, input sensitivity, and output edge rates.[30]

With our maximum ouptut voltage $V_{dd} = 1.1\,\mathrm{V}$ (disregarding overshoot), $30\% = 0.33\,\mathrm{V}$ and $70\% = 0.77\,\mathrm{V}$. Our ΔV_{out} (which will serve in the following derivation as V_p, the usual peak amplitude found in the expression for sinusoidal output) is then $0.77 - 0.33 = 0.44\,\mathrm{V}$. We will let Δt [15] be the average of the Nangate falling and rising output transitions above, or $0.14\,\mathrm{ns}$.

With slew rate terms in hand, we may calculate the full power bandwidth, f_{fp}. In the analog world, f_{fp} is the highest frequency for which an undistorted sinusoidal output is obtainable at the maximum output voltage[16]. The maximum usable frequency output (i.e., above which slew rate limiting occurs) is directly proportional to the slew rate and inversely proportional to the output signal amplitude.[31] Accordingly, we relate the slew rate and full power bandwidth as follows in Eq. 11:

$$\begin{aligned} v_{out}(t) &= V_p \sin 2\pi f t \qquad (2\pi f = \omega) \\ \frac{dv_{out}(t)}{dt} &= 2\pi f V_p \cos 2\pi f t \\ \frac{dv_{out}(t)}{dt}|_{max} &\stackrel{\text{def}}{=} \mathrm{SR} = 2\pi f V_p \qquad \max \frac{dv_{out}(t)}{dt} \text{ of sine wave is at origin, where } \cos\omega t = 1 \\ f_{fp} &\stackrel{\text{def}}{=} \frac{\mathrm{SR}}{2\pi V_p} \qquad \text{full power bandwidth defined} \end{aligned} \tag{11}$$

Using this result, Eq. 11, and our effective slew rate of $\mathrm{SR} = dv/dt = 0.44\,\mathrm{V}/0.14\,\mathrm{ns}$, we obtain $f_{fp} = 1.137\,\mathrm{GHz}$:

$$\begin{aligned} \frac{\Delta V_{out}}{\Delta t}\,(\text{ effective SR }) &= \frac{0.44\,\mathrm{V}}{0.14\,\mathrm{ns}} = 3.143\,\mathrm{V/ns} \\ f_{fp} = \frac{\mathrm{SR}}{2\pi\Delta V} &= \frac{3.143\,\mathrm{V/ns}}{(2\pi)\,0.44\,\mathrm{V}} = \frac{3.143\ \times 10^9\ \mathrm{V\,s^{-1}}}{2.76460\,\mathrm{V}} = 1.137\,\mathrm{GHz} \end{aligned} \tag{12}$$

This estimate is consistent with the fact that the $-3\,\mathrm{dB}$ bandwidth (of an op amp) will usually be greater than the full power bandwidth because the output voltage swing is less.[31] Our $-3\,\mathrm{dB}$ bandwidth estimate in Section VIII B above was $1.8\,\mathrm{GHz}$, a little larger than the $f_{fp} = 1.137\,\mathrm{GHz}$.

[15] Pardon our playing fast and loose with the notion of infinitesimal $\mathrm{d}x$ and that of the finite difference Δx. Mathematically crude connections nevertheless often lead to an accurate description of phenomena, paraphrasing Wigner's amusing characterization of the irresponsibility of physicists in this regard.[32]

[16] This characterization is from Bob Zulinski's fine online text (pdf) *Introduction to Electronics* (an introduction to the subject with emphasis on amplifier analysis and design)[33].

E. Transit time estimate

There is a figure of merit often used to evaluate the frequency response of MOSFET transitors, i.e., f_T, the short-circuit current-gain cut-off frequency.[34]

Consider Figure 6:

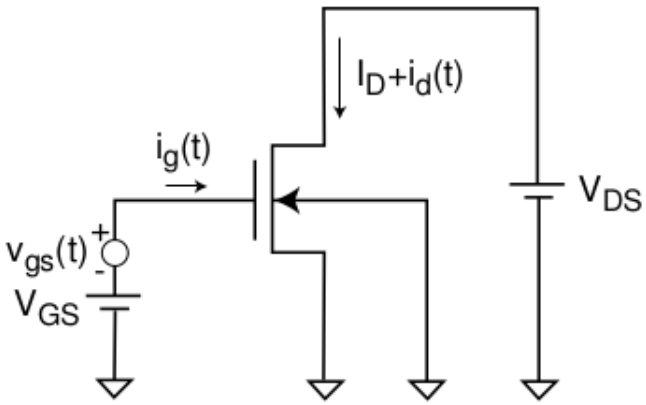

FIG. 6. To measure f_T, the MOSFET is saturation biased with output shorted from the small-signal viewpoint. $v_{gs}(t)$ is a small AC signal applied on top of the DC bias V_{GS} [34]

With the transistor in saturation, f_T is defined as the frequency at which the small-signal current gain, h_{21}, goes to unity:[34]

$$|h_{21}(f_T)| = \frac{i_d}{i_g}_{v_{ds}=0} = 1 \tag{13}$$

Though it is an interesting exercise, we will not discuss the two-port equivalent circuit for Figure 6 and its analysis using h-parameters. For our purposes, we simply wanted to make clear what we intended by f_T. What is of interest is that the analysis produces an equation using MOSFET semiconductor parameters Eq. 14:

$$f_T = \frac{1}{2\pi}\frac{3}{2}\frac{\mu_e\,(V_{GS} - V_T)}{L^2} \tag{14}$$

The inverse of f_T is then the delay time[34], τ_d: Eq. 15:

$$\tau_d = \frac{1}{2\pi f_T} = \frac{2}{3\mu_e}\frac{L^2}{V_{GS} - V_T} \tag{15}$$

We may consider the physical meaning of τ_d to be the average time for an electron to cross the channel from source to drain in a MOSFET, i.e., the transit time τ_t. That is, if electrons travel distance dy drifting at velocity v_e then $dt = dy/v_e$ and the channel transit time is

$$\tau_t = \int_0^L \frac{dy}{v_e(y)} \tag{16}$$

Identifying the terms required to calculate transit time τ_d with Eq. 15, L is the channel length. We have the electron mobility, μ_e in Eq. 15, applicable to our nmos BSIM4 model card. We may consider the mobility parameter μ_e to characterize the velocity that the carriers of interest (e.g., electrons, holes) acquire in response to the channel electric field $\mathcal{E}$ between source and drain. The BSIM4 manual[35] low field mobility parameter U0 (the BSIM4 parameter we select as surrogate for μ_e) defaults to $0.067\,\mathrm{m}^2/(\mathrm{V}\cdot\mathrm{s})$ (nmos) and $0.025\,\mathrm{m}^2/(\mathrm{V}\cdot\mathrm{s})$ (pmos). In the interest of brevity, we will consider the operation frequency implications that we develop for the case of the NMOS transistor to be applicable to the PMOS transistor in the NAND2_X1 circuit as well.

The model card for our nmos transistor (provided in Appendix G) adjusts that U0 value to $0.045\,\mathrm{m}^2/(\mathrm{V}\cdot\mathrm{s})$, so we will use that value in our calculations. For comparison, we find example data[34] presenting mobility for electrons in p-doped silicon (an NMOS transistor has p-type channel) at $3.4 \times 10^{18}\,\mathrm{cm}^{-3}$ doping level is $0.03\,\mathrm{m}^2/(\mathrm{V}\cdot\mathrm{s})$ (or $300\,\mathrm{cm}^2/(\mathrm{V}\cdot\mathrm{s})$). Our NMOS model card specifies the doping concentration parameter ndep as $3.4\times10^{18}\,\mathrm{cm}^{-3}$, so our mobility of $0.045\,\mathrm{m}^2/(\mathrm{V}\cdot\mathrm{s})$ passes a sanity check comparison with example data.

The threshold voltage parameter V_T in Eq. 15 is specified in our nmos model card (BSIM4 parameter vth0) as 0.322 V. The BSIM4 default for the NMOS device is 0.7 V, so we see that we are using a low-threshold model, as we discussed above in Section IV C. The channel length L is specified in our FreePDK45 45 nm NAND2_X1 standard cell circuit as L=0.05u, i.e., $L = 50\,\mathrm{nm}$ (discussed earlier in Section IV).

We select a V_{GS} value of 0.37 V, i.e., 0.05 V above the threshold voltage of 0.322 V. We wanted to turn on the transistor by exceeding threshold, but keep the $V_{DS} \approx (V_{GS} - V_T)$ close to zero, i.e., operating in weak inversion. If you increase V_{GS} farther above threshold, so-called overdrive, f_T (device speed) increases.[36] We wanted a baseline estimate here, so kept drive low.

Plugging the parameter values into Eq. 15 we calculate the transit time:

$$\begin{aligned}\tau_d &= \frac{2}{3(0.045\,\mathrm{m^2/(V\cdot s)}}\frac{(50\,\mathrm{nm})^2}{(0.37\,\mathrm{V}-0.322\,\mathrm{V})} \\ &= \frac{(2)(\mathrm{V\cdot s})}{(0.1350)(\mathrm{m^2})}\frac{(50\ \times 10^{-9}\,\mathrm{m})^2}{0.050(\mathrm{V})} \\ &= 0.741\,\mathrm{ps}\end{aligned} \tag{17}$$

We then have $f_T = 1/(2\pi\tau_d) = 1/(2\cdot\pi\cdot 0.741\,\mathrm{ps}) = 214\,\mathrm{GHz}$. That appears unrealistically high, until you remember that this is the unity gain figure. Any realistic application would want gain of 100 or more, which would give an operating frequency of $214.8\,\mathrm{GHz}/100 = 2.15\,\mathrm{GHz}$

F. Choose test frequency

The various estimates above suggest that we may use a conservative test frequency of 1 GHz for the specified parameters of our MOSFETs.

IX. CBCM: OBSERVATIONS OF CBCM ORIGINAL IMPLEMENTATION

The following Ngspice simulation traces (using Ngspice code in Appendix B) are from the unloaded reference inverter X1 Fig. 7):

Note that the PMOS and NMOS gates of the two inverters are turned on and off, gold inpmos and green innmos traces in Fig. 7), at different times as discussed in Section VI above. To measure the test capacitance, we are only interested in the load-charging event when the output of the inverters is driven HIGH (to V_{dd}). In Fig. 7 that occurs when inpmos falls past the threshold to turn on PMOS, pulling output to V_{dd}, e.g., just before $t = 0.7\,\mathrm{ns}$. We see the NMOS current, blue vss1#branch, and PMOS current, red vdd1#branch, of reference inverter X1 are independent currents without the usual CMOS switching transient where V_{dd} to ground current would normally register simultaneously. Remember that Ngspice assigns a negative sign to the positive conventional current flow from an independent voltage source, so the red vdd1#branch negative current spike in excess of $-50\,\mu\mathrm{A}$ (the two currents in the lowest axis of the figure units are μA; the voltage waves above have been scaled to fit on the

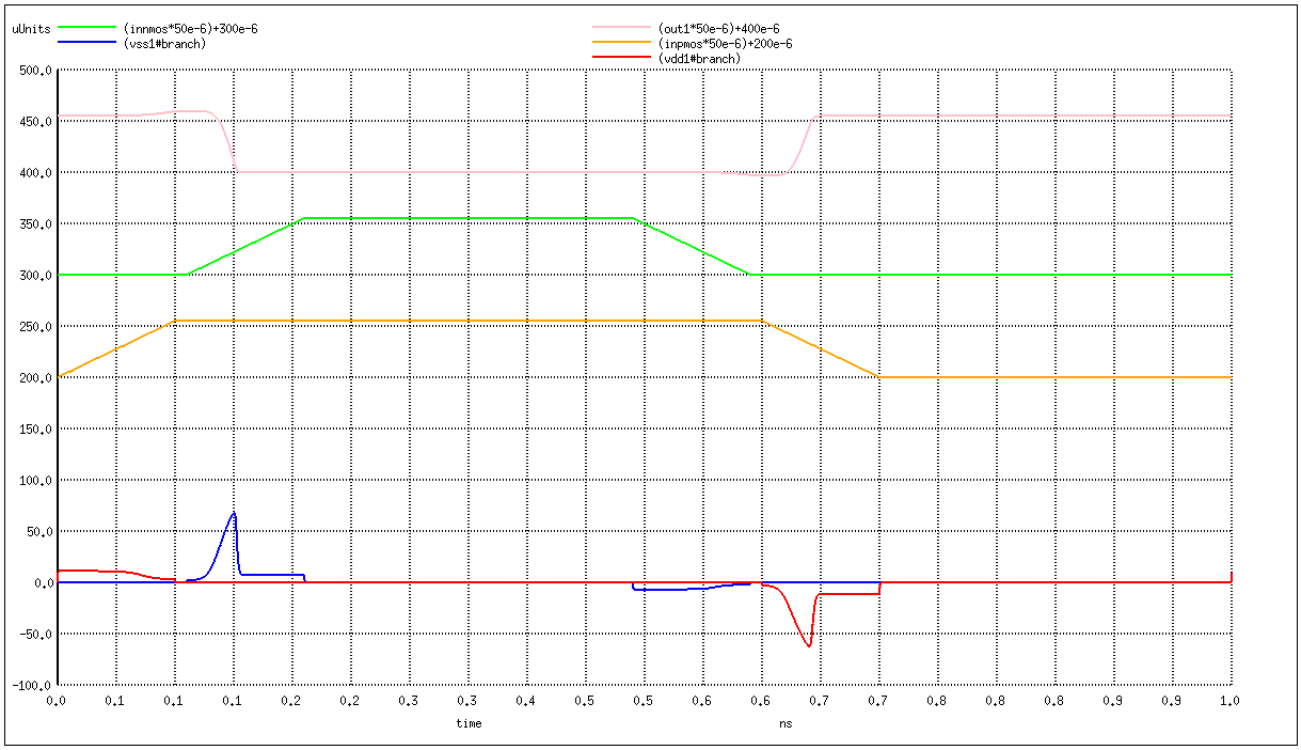

FIG. 7. Shown is a plot from a Ngspice simulation of the CBCM inverter apparatus (refer to schematic Fig. 5) at 1 GHz. The currents in the 1.5 fF loaded test inverter X2 and its output voltage wave are not included here. For measurement of capacitance, we are interested in the transition of inverter output (out1 light violet topmost trace) to HIGH, e.g., following inpmos (gold) beginning to drop, turning on the PMOS transistor. That results in a V_{dd} lead charging current spike as the PMOS transistor charges its intrinsic capacitance and any external load capacitance. The negative going pulse (vdd1#branch, red) between $t = 0.65$ ns and $t = 0.7$ ns is actually positive conventional current sourced from V_{dd} down through the PMOS transistor in Ngspice. Note that the figure is high resolution, so may be zoomed in your pdf reader to make labels more legible.

figure) when inpmos grounds, is *positive* charging current from the power supply through the PMOS pullup of the unloaded inverter X1.

The output voltage wave and power supply currents are almost identical for the 1.5 fF loaded test inverter X2, other than the noticeably larger capacitive charging pulse in excess of $-100\,\mu$A when inpmos falls Fig. 8

The Ngspice circuit file code (see Appendix B) measures 1.537202 fF for the load capacitance of 1.5 fF:

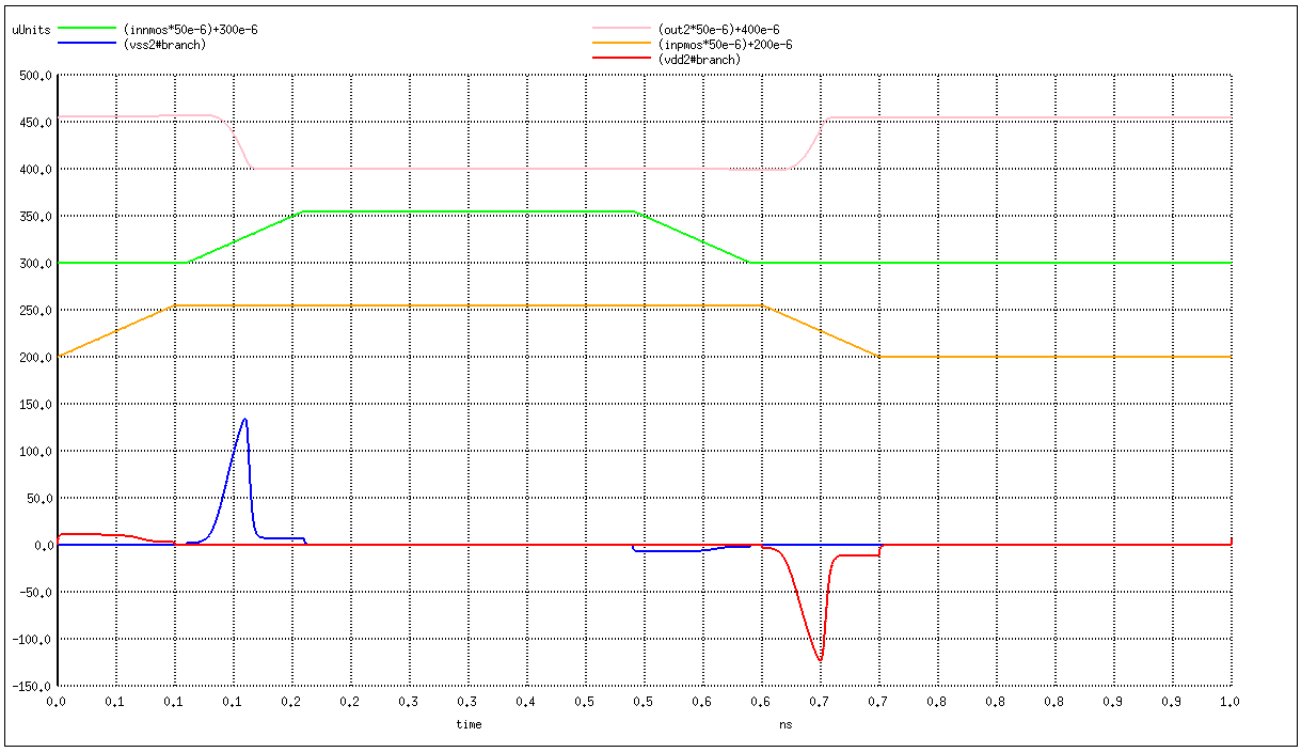

FIG. 8. Ngspice simulation run of CBCM (refer to schematic Fig. 5) at 1 GHz. NMOS discharges loaded X2 inverter capacitance with $\sim 150\,\mu$A positive excursion pulse at $\sim t = 0.16$ ns (blue trace vss2#branch). PMOS charges intrinsic and load capacitance between $\sim t = 0.65$ ns and $\sim t = 0.75$ ns (negative excursion red, vdd2#branch between $\sim -100\,\mu$A and $\sim -150\,\mu$A) as discussed in text. Note that the figure is high resolution, so may be zoomed in your pdf reader to make labels more legible.

```
No. of Data Rows : 100125
capmeasured = 1.537202e-15
Actual test capacitance on 2nd inverter was
1.5  fF
Test frequency was  1000  MHz
Input rise/fall time was  1E-10  s
Vdd supply voltage was
1.1  volts DC
```

Why are we +2.48% high on the measurement of a discrete capacitor load? The astute reader will have noticed the small red vdd2#branch positive trace in Fig. 8 occurring as

PMOS begins turn-off at $t = 0\,\text{ns}$. If our capacitive load charging current from V_{dd} via the PMOS pullup transistor of each inverter is shown as negative by Ngspice, then why are we seeing any positive PMOS excursions in the figure? We see it also in the unloaded X1 traces of Fig. 7. Because CBCM considers only the difference between the average V_{dd} currents of the reference and test loaded inverters, this should not be a problem unless they are not equal currents.[17]

X. CBCM: ANALYZE ORIGINAL CBCM ERROR

Let us examine more closely the odd opposite polarity vdd1#branch and vdd2#branch currents at PMOS turn-off.

With a 1.5 fF capacitive load on the test inverter X2 (refer to schematic Fig. 5) we observe the following return of current to the power supply (as we noted earlier, our SPICE simulator Ngspice-27 assigns current vectors from an independent voltage source a negative sign, so a positive trace indicates return to the supply rather than sourcing) in Fig. 9:

We obtained Fig. 9 by entering the following Ngspice command in the terminal window that remained open following the run of Ngspice code in Appendix B:

```
ngspice 1 -> plot vdd1#branch vdd2#branch inpmos*1e-5 xl 1n 1.2n yl 0 15e-6
```

Because we were relying on the difference in CMOS capacitive charging current between the reference inverter (unloaded) and the test inverter (loaded) to permit us to calculate the capacitance of the load, Fig. 9 is a problem. Not only is the mohawk current not directly associated with the PMOS load charging, but the loaded circuit returns *less* current here. We assumed any difference between otherwise identical circuit current should be *increased* load capacitor charging current.

When the average is taken of reference and test circuit PMOS V_{dd} currents, the test circuit average is reduced less (the average function uses signed arithmetic) than the reference average (or you could say the reference average was reduced more than the test circuit). Either way you view the result, the difference between the two current averages (refer to Eq. 8 discussion earlier) appears greater than it should if only actual load capacitor charging current was considered. This causes a slight increase in the value of target load capacitance calculated.

[17] However, they are *not* equal and this is a problem as we soon discover in Section X.

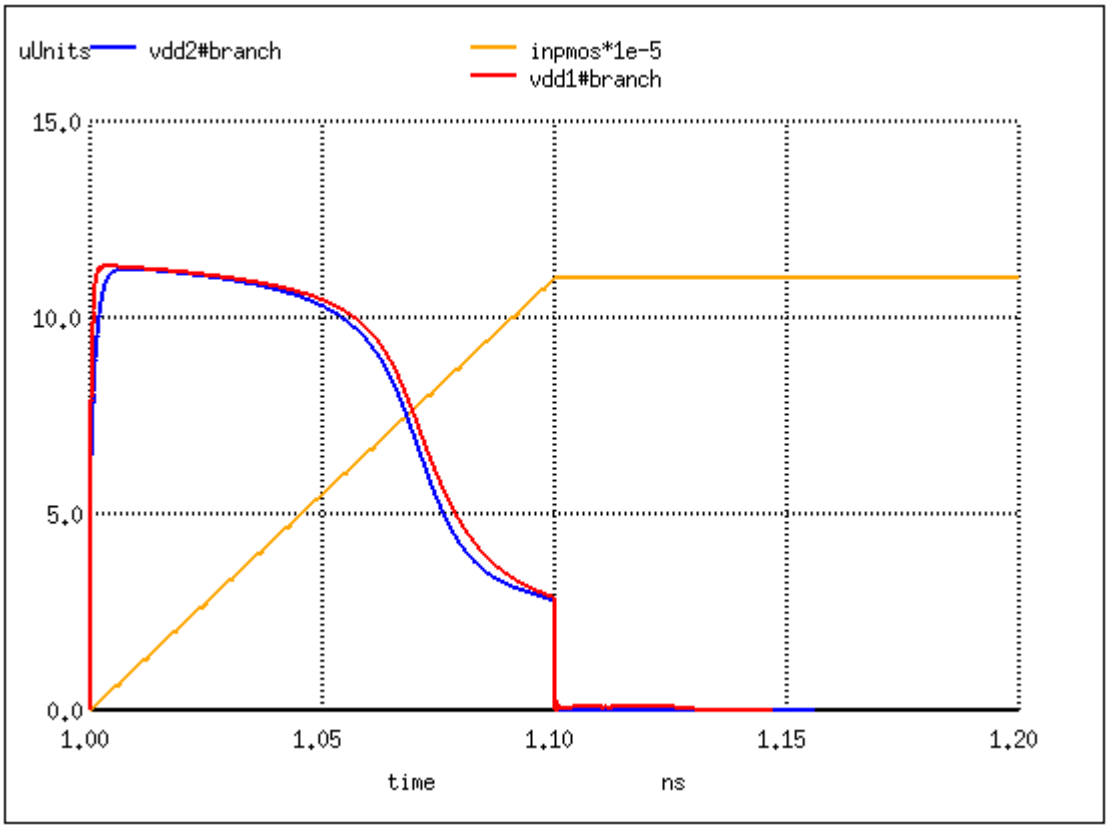

FIG. 9. In the Ngspice trace we observe what we dubbed the *mohawk current* returned through each inverter source lead to V_{dd} upon PMOS turn-off (we thought the red and blue V_{dd} current traces resembled a profile view of a Native American haircut of the Mohawk tribe of the Five Nations in pre-colonial North America). The gold inpmos ramp is the scaled gate voltage drive signal at test frequency 1 GHz, rise time of $t_r = 0.1$ ns . This mohawk current is referred to in the literature (less colorfully perhaps) as "parasitic charge injection" or "charge-dumping from overlap capacitances".[37]. The blue trace from the loaded inverter X2 is not identical to the unloaded reference inverter red trace X1 and this is a problem for CBCM. Units of current are μA.

A. 2002 Fan *et al* analysis

We mentioned the 2004 patent application by Jensen *et al*[37] in the caption for Fig. 9, but there was an earlier patent application directly concerned with the impact of the mohawk current on CBCM. For example, 2002 Fan *et al*[38] proposed a method "to limit measurement error due to the return of different size negative currents...so that accuracy of capacitance measurement improves" when using the CBCM method.

Fan *et al*[38] explains the problem in the following manner (we refer to our schematic and labels for clarity). The equivalent capacitance looking down the gate of the two PMOS transistors in Fig. 5 is different. In particular, the reference inverter X1 PMOS transistor sees only the device intrinsic capacitance gate to drain (PMOS drains connect to inverter output), while the test inverter X2 PMOS gate to drain path sees both the intrinsic capacitance and that of the load capacitor C_test. The resulting difference between negative current in X1 and X2 when PMOS gate signal returns to V_{dd} from ground[18] "often leads to an error in the measurement of capacitance" (C_test).

We would add that the additional capacitance on the test inverter X2 drain is seen in series with the intrinsic gate to drain capacitance, meaning that the resulting capacitance is lower ($C_{\text{series}} = (1/C_{\text{overlap}} + 1/C_{\text{L}})^{-1}$). When the PMOS transistor gate voltage changes on turn-off, the resulting discharge current of the effective gate-drain parasitic capacitance is then smaller by Eq. 6, i.e., the negative current is directly proportional to the effective capacitance seen.

B. An energy band explanation

A somewhat different analysis of the problem was presented in a 2000 paper[39]. The authors viewed the matter as the transistor switch turning off faster than the channel charge is able to redistribute, leaving residual charge in the channel at an effective energy band greater than the source and drain. In this condition the "forces caused by the electrical field and diffusion are not balanced and the source-channel and the drain-channel junctions can be considered to be forward biased." The remaining charge in the channel is then injected into the source and drain preferentially (over the substrate). They offer the following figure to illustrate their energy band explanation Fig. 10:

The focus of [39] was the development of an analytical model for charge injection in MOSFET switches generally, though their specific circuit was a sample-and-hold function with the MOSFET transistor as the switch between an amplifier buffering the input voltage to be sampled and a capacitor to be charged to the instantaneous voltage at the sample interval. They did present equations making use of channel charges transferred during the

[18] Jensen describes this as "when the gate Voltage of the IGFET is changed suddenly from Zero volts to a positive Voltage, positive charge is dumped at the Source and drains terminals of the IGFET by the action of the Overlap capacitors."[37] Negative charge is dumped for a positive to zero transition.

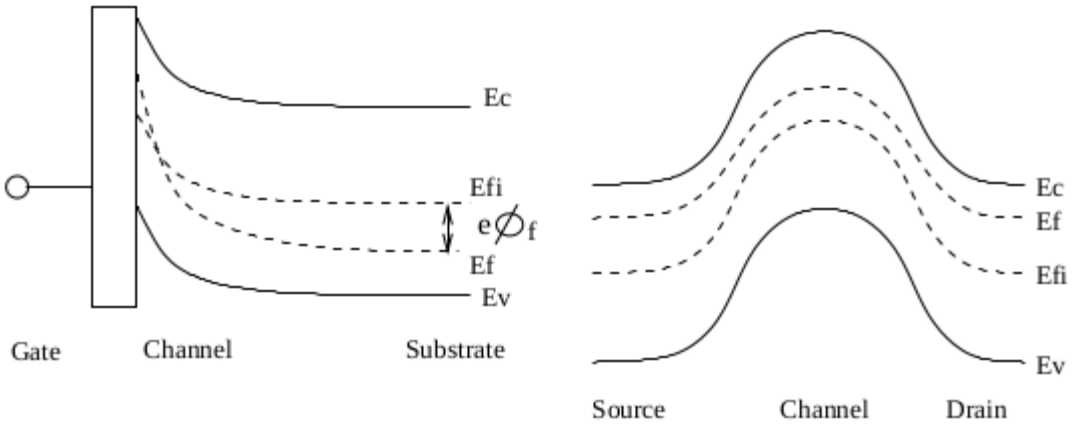

FIG. 10. Figure 3 from [39] illustrating non-equilibrium NMOS switch energy bands. Our comments (the authors declining to define terms assumed to be commonly known): On the left side of the figure we see that the Fermi level E_f (where fifty percent of the charge carriers can be found) crosses the intrinsic Fermi level of the semiconductor E_{fi} at the channel. The energy difference between the Fermi level E_f and the bottom of the conduction band E_c at the insulator-semiconductor interface becomes smaller than that between the Fermi level and the top of the valence band E_v (at 0 K valence band electrons occupy the highest energy levels available). On the right of the figure we see a depiction suggesting that residual charge in the channel is at a higher energy in relation to the source and drain levels and so is injected into those destinations.

turnoff. Unfortunately, most of the terms were not easily obtained or in fact required some of the unknowns which were the object of the measurement. We mentioned this work primarily because of the interesting energy band qualitative explanation offered for the turn-off charge injection.

C. 2002 Fan *et al* proposed solution

The 2002 Fan *et al* patent application mentioned above[38] proposed a method to correct the negative current injection when using the CBCM method. This involved basically adding a second PMOS transistor to each inverter of the CBCM circuit we gave in Fig. 5, in series with the existing PMOS device of the pullup portion of the circuit. They added a third gate drive phase also (remember we used two phases to turn the PMOS and NMOS devices on

and off separately).

The purpose of the Fan modifications was to assure the path to V_{dd} (and the monitoring ammeters) was open when the PMOS with drain connected to the load capacitance was turned off, i.e., the additional PMOS transistor was placed above the PMOS load PUN transistor and turned off before the driving transistor below it. This sounded promising, but did not improve the accuracy of our measurement when we implemented the circuit in Ngspice. Their Table 1 comparison of standard CBCM error to that achieved with their suggested correcting circuit was not very optimistic either (there were three rows in the table, with the third scrambled so unusable). For example, with test capacitor of 1 fF their fix improved the measurement error from 19.7% to 13.9%. We accordingly put that proposed fix aside (though we did appreciate their description of the problem).

D. 2004 Jensen *et al* proposed solution

2004 Jensen *et al*[37] proposed a solution to the overlap capacitance charge dumping into source and drain terminals on transition of gate potential which involved using a pair of parallel MOSFETs switched by a same phase gate signal and an identical type MOSFET preceding and following this parallel pair, each with shorted drain and source, these two switched by a gate signal of opposite phase.

For example, the NMOS transistor in our inverter would be replaced by two parallel NMOS and another NMOS before and after the pair (each of the latter shorted drain and source). The original signal on this NMOS of the inverter would be provided inverted to the two shorted NMOS transistors before and after the parallel pair. Basically this results in using the shorted transistors to dump opposite polarity charge from their gate to source and gate to drain overlap capacitances at the original source and drain of the now parallel pair inverter NMOS (or the PMOS, with PMOS replacing NMOS instances above, of any circuit where this fix was implemented) to cancel the injection current anomalies normally seen on switching of gate potential. See Fig. 11 for depiction of relevant MOSFET structures.

We did not test this proposal (it is simply too cumbersome for our purposes here) and no experimental validation data was offered in the patent application.

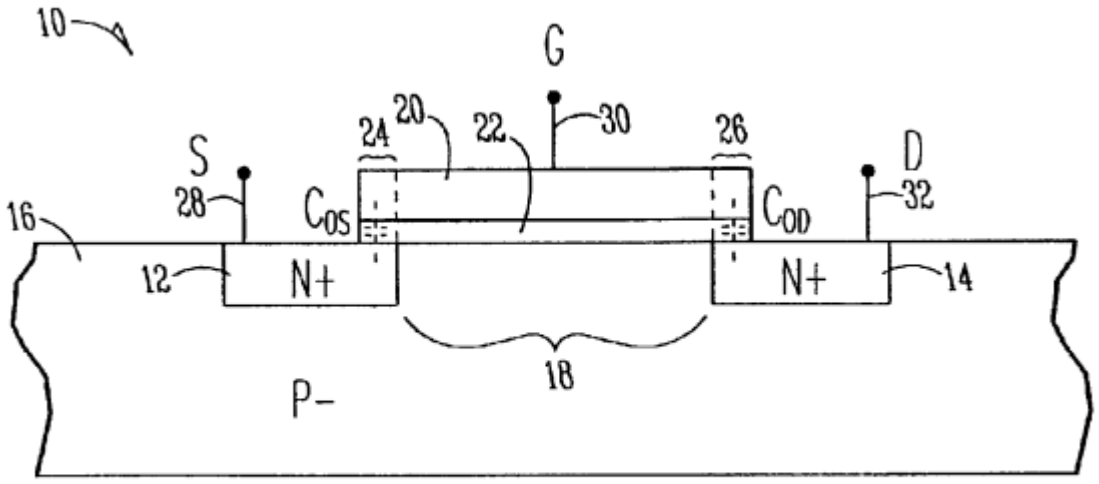

Fig. 1 (Prior Art)

FIG. 11. Figure 1 from 2004 Jensen *et al*[37] depicts relevant areas of an n-channel enhancement IGFET (a MOSFET being a form of insulated gate field effect transistor). The S (source) and D (drain) regions are heavily n^+-doped (i.e., electron donor impurities from Group 15 of the Periodic Table, e.g., P, As) within a lightly p-doped (electron acceptor impurities from Group 13, e.g., B, In) substrate. The region labeled 18 is a channel between source and drain. A positive G gate voltage $V_{GS} > V_{\text{thresh}}$ creates a minority carrier channel (electrons in this case, hence an effective "n-channel") of low impedance between S and D. The physical overlap of gate G with source and drain regions, labeled 24 and 26, produces parasitic overlap capacitances C_{OS} and C_{OD}. These dump charges onto their corresponding nodes upon rapid V_{GS} change.

XI. OUR SOLUTION TO THE CBCM CHARGE INJECTION ERROR

Having satisfied ourselves that the mohawk current (positive charge dump to source lead to V_{dd} on PMOS turnoff) problem was in the stars as it were and not within us[19], and having observed an increase in error using the Fan *et al*[38] suggested circuit revision, as discussed above in Section X C, we decided to simply strip the offending opposite polarity current from the V_{dd} current vectors vdd1#branch and vdd2#branch generated by Ngspice and use new clean copies vd1brclean and vd2brclean for the CBCM measurment. The implementation of

[19] "The fault, dear Brutus, is not in our stars, but in ourselves," Cassius speaking to Brutus in Scene II of *The Life and Death of Julius Caesar*, by William Shakespeare, 17th century England.

this software solution (the code to shave the mohawk as it were) was relatively trivial in Ngspice (do expect a few seconds processing delay when you run the circuit file, given the number of simulation steps is $\mathcal{O}(10^5)$ with more than 70 Ngspice vector arrays each of that size):

```
****
* BEGIN SHAVE THE MOHAWK
* strip the asymmetrical negative return
* currents on pmos turn-off

* get length of vectors to process in loop
let samplelen=length(vdd1#branch)

* create copy of current vectors vdd1#branch,
* vdd2#branch to clean of negative return currents
let vd1brclean = vdd1#branch
let vd2brclean = vdd2#branch

* loop through the new current vectors and
* strip any positive current
* remembering that ngspice views negative return current
* to power supply as positive

* init loop counter and vector index
let loop = 0

* while loop counter is less than the length of
*  the current vectors (begin with index 0)
* check for a positive current value and replace
* it with zero as not relevant to charging C

  while loop < samplelen
```

```
    if vd1brclean[loop] > 0
      let vd1brclean[loop] = 0
    end

    if vd2brclean[loop] > 0
      let vd2brclean[loop] = 0
    end

    let loop = loop + 1
  end
* END SHAVE MOHAWK
```

See, for example, Ngspice-27 circuit file 20211209CBCMinvOnlyStripMohawk.cir (code given in Appendix C.

A. Alternative fix if you prefer more manual work

If you isolate one of the PMOS capacitive load charging current spikes (when the output of the inverters goes HIGH, see Fig. 7 and Fig. 8) from each inverter and integrate its value (integrate the Ngspice current vector) from its beginning to end (rather than averaging the current over the entire simulation as in the original CBCM measurement method), you will eliminate the mohawk reverse polarity current problem, which occurs only during PMOS turnoff (input going HIGH).

Take the difference between the two integrated charging spikes from reference and test inverter PMOS currents in their respective leads to the two V_{dd} supplies (which is a difference in charge at this point, being the difference in the integrals of current over a time interval). Divide the difference in charge by the V_{dd} voltage and you obtain an accurate load capacitance result, similar to what we did in Equation 8 ($I_\Delta \, \Delta t \approx \int_{t_1}^{t_2} I(t) \, dt$ for an approximately linear ramp $I(t)$). The following is some sample Ngspice-27 code to accomplish this in a terminal session following a run of the simulation circuit file 20211209CBCMinvOnly.cir (see Appendix B) with the test inverter loaded by a discrete 1.5 fF capacitor rather than the NAND2_X1 input, using uncorrected CBCM:

```
No. of Data Rows : 100125
inv1rmscur          =  -1.017119e-06 from=  0.000000e+00 to=  5.000000e-09
inv2rmscur          =  -2.708041e-06 from=  0.000000e+00 to=  5.000000e-09
capmeasured = 1.537202e-15
Actual test capacitance on 2nd inverter was  1.5  fF
Test frequency was  1000  MHz
Input rise/fall time was  1E-10  s
Vdd supply voltage was  1.1  volts DC

ngspice 1 -> meas tran inv1Cur INTEG vdd1#branch from=600p to=707.143p
inv1Cur             =  -1.83109e-15 from=  6.00000e-10 to=  7.07143e-10

ngspice 1 -> meas tran inv1Cur INTEG vdd2#branch from=600p to=708.182p
inv1Cur             =  -3.49556e-15 from=  6.00000e-10 to=  7.08182e-10

ngspice 1 -> print (abs( 1.83109e-15 - 3.49556e-15  )/1.1)
(abs( 1.83109e-15 - 3.49556e-15 )/1.1) = 1.513155e-15
```

For the above interactive session after running the circuit file, we obtained the beginning and end (i.e., the integration limits) of the PMOS charging spike (the negative current spike in the following figure, though it is positive conventional current from V_{dd}) by examining the plot of input waveforms and the V_{dd} current for the particular inverter and mouse-clicking on a point slightly in advance of the end of the current spike[20] Fig. 12:

We see that the uncorrected CBCM measurement (1.537202 fF) of the 1.5 fF capacitor (loading the test inverter) was about 2.5% high due to the presence of the mohawk charge dumping current at PMOS turnoff. Integrating only the PMOS capacitive charging current spike produces a more accurate value (1.513155 fF), high by only 0.9%. This technique is then another means of correcting the CBCM method, though not as attractive because requires direct participation of the observer, i.e., would be difficult to implement in hardware, as is the intent of the original CBCM proposal.[1]

[20] Ngspice prints the x and y coordinates in the terminal window when you mouse-click in a plot window.

XII. REVISED CBCM RESULTS

We now make some measurements using the revised CBCM method.

A. Test calibration capacitor

With C_test equal to 1.5 fF (instead of the input load of the NAND2_X1 gate) we measure 1.51224 fF using the Ngspice-27 circuit file 20211209CBCMinvOnlyStripMohawk.cir (code given in Appendix C) implementing the charge-based capacitance measurement method discussed in in Section VI and corrected with the method discussed in Section XI:

```
capmeasured = 1.512244e-15
Actual test capacitance on 2nd inverter was  1.5  fF
Test frequency was  1000  MHz
Input rise/fall time was  1E-10  s
Vdd supply voltage was  1.1  volts DC
ngspice 1 -> print 1.512244/1.5
1.512244/1.5 = 1.008163e+00
```

Our error measuring this calibration capacitor at 1 GHz is then less than 1%.

B. 1 GHz test NAND2_X1 loaded 59 fF

Using circuit file 20211209CBCMmeasNANDStripMohawk.cir (code given in Appendix D), we now go to the original goal of measuring the input capacitance of the NAND2_X1 input A1 (labeled in1 in schematic 1):

```
capmeasured = 1.708691e-15
(equivalent NAND input capacitance being charged by 2nd inverter, Farads)
(Nangate typical build library suggests NAND2_X1 A1 input 1.599 fF)
Load capacitance on NAND output:   59  fF
Test frequency was  1000  MHz
Input rise/fall time was  1E-10  s
Vdd supply voltage was  1.1  volts DC
```

```
ngspice 1 -> print 1.708691/1.599
1.708691/1.599 = 1.068600e+00
```

We are within 6.8%[21] of the typical corner[22] Nangate stdcells-databook.pdf[11] suggested input A1 capacitance of 1.5990 fF for NAND2_X1. As is seen in the terminal response above from Ngspice, the NAND2_X1 output (labeled out in our schematic 1, label ZN in stdcells-databook.pdf) was loaded with a 59 fF capacitor during the run. 59.3567 fF was the upper load used to specify propagation delay and output transition time in the Nangate stdcells-databook.pdf. 59.3567 fF was also the maximum capacitance permitted on the NAND2_X1 output in the stdcells.lib file that accompanied our library distribution.[11] stdcells.lib was built by the Nangate NGLibraryCharacterizer with Spice engine Nanspice v2011 (and characterization corner typical as we have noted previously). Hence we selected 59 fF for our loaded run.

The INV_X1 circuits of our CBCM apparatus may drive up to 60 fF, consulting the same library references above, so driving the input of the NAND2_X1 should be relatively easy.

C. Run with 1 fF NAND2_X1 load

A quick test with the NAND2_X1 loaded with 1 fF instead of 59 fF[23] resulted in a change in the measured input capacitance:

```
capmeasured = 1.727344e-15
(equivalent NAND input capacitance being charged by 2nd inverter, Farads)
(Nangate typical build library suggests NAND2_X1 A1 input 1.599 fF)
Load capacitance on NAND output:   1  fF
Test frequency was  1000  MHz
Input rise/fall time was  1E-10  s
Vdd supply voltage was  1.1  volts DC
ngspice 1 -> print 1.727344/1.599
1.727344/1.599 = 1.080265e+00
```

[21] Compare with 8.67% error with mohawk injection current present running circuit file 20211209CBCMmeasNAND.cir (code given in Appendix E).

[22] See Section IV above for discussion of the term "corner."

[23] Modify the parameter NANDLoadCapacitance in circuit file 20211209CBCMmeasNANDStripMohawk.cir, code given in Appendix D.

We measure input capacitance of 1.727344 fF, about 8% off the suggested value (about a 1% change from the fully loaded measurement earlier). It is not unexpected that load capacitance of a CMOS gate is reflected to the input, i.e., modifies the input characteristic somewhat. The Nangate data did not specify what they expected those changes to be, but 1% change is not alarming to us.

D. Run with 100 MHz NAND2_X1 test frequency

A run with the NAND2_X1 loaded with 59 fF at lower test frequency 100 MHz[24] resulted in little change to the measured input capacitance:

```
capmeasured = 1.730522e-15
(equivalent NAND input capacitance being charged by 2nd inverter, Farads)
(Nangate typical build library suggests NAND2_X1 A1 input 1.599 fF)
Load capacitance on NAND output:   59  fF
Test frequency was  100  MHz
Input rise/fall time was  1E-09  s
Vdd supply voltage was  1.1  volts DC
ngspice 1 -> print 1.730522/1.599
1.730522/1.599 = 1.082253e+00
```

Testing at 100 MHz we measure 1.73 fF input capacitance, within 8.2% of the typical corner[25] Nangate stdcells-databook.pdf[11] suggested input A1 capacitance of 1.5990 fF for NAND2_X1.

E. Run with 2 GHz NAND2_X1 test frequency

A run with the NAND2_X1 loaded with 59 fF at higher test frequency 2 GHz (modify the parameter testfreq in circuit file 20211209CBCMmeasNANDStripMohawk.cir, code given in Appendix D) resulted in a small change downwards in the measured input capacitance:

```
capmeasured = 1.538518e-15
```

[24] Modify the parameter testfreq in circuit file 20211209CBCMmeasNANDStripMohawk.cir, code given in Appendix D.

[25] See Section IV above for discussion of the term "corner."

```
(injection current or mohawk stripped prior)
(equivalent NAND input capacitance being charged
 by 2nd inverter, Farads)
(Nangate typical build library suggests
NAND2_X1 A1 input is 1.599 fF)
Load capacitance on NAND output:
59  fF
Test frequency was  2000  MHz
Input rise/fall time was  5E-11  s
Vdd supply voltage was
1.1  volts DC
```

We measure 1.53851 fF, about 4% lower the Nangate suggested value of 1.599 fF. We note that at test frequency 2 GHz the output of the NAND2_X1 gate is beginning to slew quite a bit, i.e., it is just reaching $\sim V_{dd}$ when the next input change arrives. See Section VIII for a detailed discussion of operating frequency limits.

XIII. MANUAL METHOD

For those on a budget, i.e., unable to afford the CBCM pair of inverters, you can simply drive the NAND2_X1 input in1 with a SPICE pulse generator and calculate the effective input capacitance using the charge delivered to the gate (see Eq. 2 in Section V).

As we said in Section VII, MOSFETs in CMOS gates are voltage-controlled switches (see §3.6 [6], or §11.5.1 [4] for example) so it would be inappropriate to drive them with a current source if the intent is to measure gate input capacitance seen by another CMOS driving circuit. Accordingly, we use roughly the output R_{ds} of a CMOS circuit for our standard library 45 nm process, 1 kΩ [26], as the source impedance of our SPICE pulse driver, i.e., insert a 1 kΩ resistor in series with the pulse generator +-lead in series with the connection to the NAND2_X1 input in1 . Example Ngspice code lines:

```
RTOTARGETC PulseInput in1 1000
```

and

[26] We estimated about that value in Section XV for the NAND2_X1 effective PMOS R_{ds}.

```
VIN1 PulseInput 0 PULSE(0 {DCsupplyVoltage}
+ 0 {TriseOrFall} {TriseOrFall} {PulseWidth} {testperiod})
```

where in1 is the A1 input of the NAND2_X1 gate in Schematic 1.

For an Ngspice run of the circuit file in Appendix F, we obtain terminal result:

```
charge transferred to NAND input: (Coulombs)
incurlh             =  1.39093e-15 from=  0.00000e+00 to=  1.00000e-10
voltage change on NAND input: (Volts)
maxin1              =  1.088130e+00 at=  1.000000e-10

capacitance of NAND input measured, fF:
nandincap = 1.278276e+00

(Nangate typical build library suggests
NAND2_X1 A1 input is 1.599 fF)

Load capacitance on NAND output:
59  fF

Test frequency was  1000  MHz
Input rise/fall time was  1E-10  s
pulse source output impedance: (ohms)
@rtotargetc[resistance] = 1.000000e+03
Vdd supply voltage was
1.1  volts DC
```

We measure 1.278276 fF above, 20% lower than Nangate suggested 1.599 fF. The CBCM results appear more accurate (see test results in Section XII), but the CBCM method requires more circuit code.

XIV. PROPAGATION DELAY

Nangate stdcells-databook.pdf measures propagation delay from 50% input falling/rising to 50% output falling/rising. For our typical corner build and operating conditions, e.g., $V_{dd} = 1.1\,\mathrm{V}$ and $25^\circ\,\mathrm{C}$, the propagation delay of input A1 (our in1) to output ZN (our out) rising with output loaded with 59 fF is suggested to be 250 ps for an input transition (70% to 30% falling) 198.5 ps. Let us measure that in our circuit using circuit file 20211209CBCM-measNANDStripMohawk.cir, code given in Appendix D.

The relevant waveforms are Fig. 13

Our inverter input drive signal at 1 GHz has a rise/fall time of 100 ps (by design) as seen above in the Ngspice terminal report. However, the outputs of the two inverters rise and fall at a much faster rate, partly because CMOS inverters have a large gain close to the switching voltage (the voltage transfer characteristic is very steep at threshold[6]) and partly because of our two-phase drive which turns on the PMOS and NMOS at separate times. By the time the innmos signal rises on the inverter inputs to turn on the NMOS pull down network, the inpmos input has already turned off the PMOS transistor, i.e., the PMOS transistor has already passed through the initial overshoot operating region (reversed operating mode, probably synonymous with our overlap capacitance charge dump event) with its gate exceeding HIGH threshold, as well as the next region where it eventually turns off completely[40]. Refer to Fig. 7 above for phase relationship of innmos and inpmos.

The result is that the output of the test inverter X2, which will be the in1 input to the NAND2_X1 gate (refer to schematic 1 of the NAND_X1 circuit and schematic 5 of the CBCM two-inverter circuit, where C_test on inverter X2 out2 will be replaced by a line to in1 input of the NAND2_X1 gate), will present a 70% to 30% falling transition of only 10 ps to the NAND2_X1 gate input. We measured that in Ngspice (typing in the following commands in the interactive Ngspice session following the initial run of the circuit file), referring to Fig. 13, first finding the high point of the input signal before it falls (where we used the slight overshoot increase over V_{dd})

```
ngspice 1 -> meas tran maxinvolt MAX_AT in1  from=0.1n to=0.15n
maxinvolt            =  1.206750e-10 with=  1.152230e+00
```

Then we obtained the time interval from when in1 declined to $(0.7)(1.15223\,\mathrm{V}) = 0.80656\,\mathrm{V}$ to $(0.3)(1.15223\,\mathrm{V}) = 0.34567\,\mathrm{V}$:

```
meas tran tdiff TRIG in1 VAL=8.0656e-01 FALL=1 TARG in1 VAL=3.4567e-01 FALL=1
tdiff                = 1.00896e-11 targ= 1.568469e-10 trig= 1.468379e-10
```

(We omitted the prompt in the meas command line above because the line was exceeding available column width.) tdiff of 10.00896 ps is then the NAND2_X1 input falling transition time using the Nangate criteria. The closest input transition selection for Nangate propagation delay estimate is 1.2 ps, for which the 59 fF-loaded NAND2_X1 output rising propagation delay is 150 ps. We measure $t_{pLH} = 64.745$ ps from input fall to 50% point to output rise to 50% point.

```
meas tran tdiff TRIG in1 VAL=5.7612e-01 FALL=1 TARG out VAL=5.4984e-01 RISE=1
tdiff                = 6.474545e-11 targ= 2.170982e-10 trig= 1.523527e-10
```

Our observations of simulated performance of the CMOS standard circuits appears to be consistent with the library general specifications (somewhat faster in general).

XV. OUTPUT RESISTANCE OF NAND2_X1

We now estimate the NAND2_X1 output resistance on the output transition t_{pLH} (output LOW to HIGH). We assume the reader can apply the methods presented to the alternative case, output HIGH to LOW transition t_{pHL}.

A. Method using exponential characterization

We treat the output voltage waveform of the switch as an exponential with a time constant defined by R_S and C_{eq}[41](citing [42])

$$V_o = V\left[1 - \exp\left(\frac{-t}{R_S C_{eq}}\right)\right] \tag{18}$$

where R_S is the output impedance of the gate and C_{eg} the equivalent capacitive load on the output. For our purposes, $C_{eg} = C_L = 59$ fF, which should dominate any intrinsic capacitance of the PMOS or NMOS transistors.

Considering the equivalent output resistance $R_S = R_{DS}$ of the PMOS transistor in the output transition from LOW to HIGH, our time interval will be the rise time t_r which we define here (departing from the Nangate definition earlier) as the time required to rise from 0.1 to 0.9 of the final output voltage at the NAND2_X1 output. We solve for t_r using Eq. 18:

$$
\begin{aligned}
\frac{V_o}{V} = 0.9 &= \left[\exp\left(\frac{-t_{0.9V}}{R_S C_{eq}}\right)\right] \\
\ln(0.9) = -0.1053 &= \left(\frac{-t_{0.9V}}{R_S C_{eq}}\right) \qquad \Longrightarrow 0.1053 R_S C_{eq} = t_{0.9V} \\
\frac{V_o}{V} = 0.1 &= \left[\exp\left(\frac{-t_{0.1V}}{R_S C_{eq}}\right)\right] \\
\ln(0.1) = -2.3026 &= \left(\frac{-t_{0.1V}}{R_S C_{eq}}\right) \qquad \Longrightarrow 2.3026 R_S C_{eq} = t_{0.1V} \\
t_r &= R_S C_{eq}\,(t_{0.1V} - t_{0.9V}) = 2.19722(R_S C_{eq}) \approx 2.2(R_S C_{eq})
\end{aligned}
\tag{19}
$$

Knowing $C_{eq} = C_L$, if we measure the times at which the 0.1 and 0.9 voltage points on the rising output transition occur to obtain t_r, we then obtain an estimate of the PMOS drain source resistance

$$
R_S = R_{DS} = t_r/(2.2C_L) \tag{20}
$$

using the result of Eq. 19.

Refer to the waveforms in Fig. 13 for the 1 GHz run of NAND2_X1 with $C_L = 59$ fF. We run the circuit file 20211209CBCMmeasNANDStripMohawk.cir (see Appendix D) in Ngspice and interactively enter the following commands in the open terminal window. First we obtain the final voltage of our output rising signal, selecting an interval to measure on Fig. 13 where out appears to be leveling out, e.g., $t = 0.3$ to $t = 0.4$ ns:

```
ngspice 1 -> meas tran maxinvolt MAX_AT out  from=0.300n to=0.400n
maxinvolt            =  3.999750e-10 with=  1.094780e+00
```

It appears our final voltage on out rising will be 1.09478 V. We then calculate the 0.1 and 0.9 fractions of that voltage, making use of Ngspice interactive calculator capability:

```
ngspice 1 -> print 1.094780e+00*0.1
1.094780e+00*0.1 = 1.094780e-01
ngspice 1 -> print 1.094780e+00*0.9
1.094780e+00*0.9 = 9.853020e-01
```

We now have the two output voltage trigger points, so will enter the Ngspice command to measure the time difference between the output out at those two points:

```
meas tran tdiff TRIG out VAL=1.094780e-01 RISE=1 TARG out VAL=9.853020e-01 RISE=1
tdiff                =  1.231943e-10 targ=  2.910126e-10 trig=  1.678183e-10
```

We obtain $t_r = 123.1943\,\text{ps}$ for the tdiff. Now we apply Eq. 20 using our $C_L = 59\,\text{fF}$ value:

```
ngspice 1 -> print 1.231943e-10/(2.2*59e-15)
1.231943e-10/(2.2*59e-15) = 9.491086e+02
```

We obtain an estimated average $R_{ds} = 949\,\Omega$ for the single M3 PMOS transistor in saturated pullup mode, i.e., charging the output capacitance C_L.

We should note that in this run (as in the other measurements of the in1 input capacitance earlier) the second input in2 of the NAND2_X1 gate was tied to V_{dd} via a $10\,\text{k}\Omega$ resistor (see line declaring circuit element RX32 in the circuit file Appendix D). Recall that a NAND2 gate output is LOW/ground if both inputs are HIGH/V_{dd} (and output HIGH for all other combinations). The output goes HIGH then when in1 falls to ground, turning on only the M3 PMOS transistor of the circuit (refer to Schematic 1). We have therefore measured the effective drain-source resistance of a single PMOS of the pair in NAND2_X1. If both inputs had dropped to LOW, we would have expected to measure the effective parallel resistance of both PMOS transistors, M3 and M4 as the output climbed to HIGH/V_{dd}, charging through both PMOS in ON state.

B. Method using integration of $V_{DS}(t)$ and $I_D(t)$

As a check on this resistance result, it is possible to obtain an equivalent drain resistance R_{eg} by taking the average of the integral of the ratio of the functions $V_{DS}(t)$ and $I_D(t)$ over the time interval of interest, e.g., the time the MOSFET transistor is ON and charging a

load capacitance.[6]:

$$\begin{aligned} R_{eq} &= \text{average}_{t=t_1 \ldots t_2} \left[R_{on}(t) \right] \\ &= \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} R_{on}(t) \, \mathrm{dt} \\ &= \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} \frac{V_{DS}(t)}{I_D(t)} \, \mathrm{dt} \end{aligned} \tag{21}$$

Note that, in general[27]

$$\int_n^m \frac{f(x)}{g(x)} \, dx \neq \frac{\int_n^m f(x) \, dx}{\int_n^m g(x) \, dx} \tag{22}$$

However, we have the discrete time data for the two functions in the form of the Ngspice vectors[28] for $V_{DS}(t)$ and $I_D(t)$ generated by the BSIM4v4[3] simulation.

We mentioned BSIM4 briefly in Section III above. The Berkeley Short-channel IGFET Model is a compact model. A compact model is basically a set of equations expressing MOSFET currents and charges (or, depending on your viewpoint, capacitances[29]) as functions of terminal voltages. The set of equations in this case is more than 20 pages and there are more than 100 model parameters. BSIM4 "enables circuit designers to accurately simulate CMOS circuits by including gate tunneling, quantum effect, and RF effects."[44] We select the BSIM4.8.1 option in our Ngspice-27 model cards.

We will consider the quotient of the two internal BSIM4 data vectors for $V_{DS}(t)$ and $I_D(t)$ of the specific transistor M3 (refer to Schematic 1)

```
@m.x3.m3[vds]/@m.x3.m3[id]
```

to be a new function (a vector of quantities) consisting of the element-by-element (at time step resolution of the simulation) quotient of the complex BSIM4v4 modeling functions for drain-source voltage and drain current of the MOSFET over time. We may then integrate

[27] Gradshteyn and Ryzhik[43] §3-4, *Definite Integrals of Elementary Functions* contained no general formulas for rational functions, although some special cases in §3.11 were covered.

[28] We remind the reader that the term "vector" here means a sequence of data accessible by one or more indices.

[29] $C_{ij} = \delta_{ij}(\partial Q_i / \partial dV_j)$, where $i, j \in \{g, b, s, d\}$, i.e., gate, bulk, source and drain terminals and δ_{ij} is the Kronecker delta (equals (-1) if index $i \neq j$, $(+1)$ otherwise). We look at the capacitance modeling in BSIM4 in more detail in Appendix A.

that new function (represented as a vector of quotients) and divide the result by the δt of the limits of integration to obtain the average R_{eq}, as set out in Eq. 21.

Accordingly, we added the following Ngspice lines to the control code within the circuit code file 20211209CBCMmeasNANDStripMohawk.cir (see Appendix D):

```
* save the NAND2 PMOS M3 pullup transistor BSIM4 vectors id, and vds
 save all @m.x3.m3[id] @m.x3.m3[vds]
```

to obtain access to the BSIM4v4 internal id drain current of PMOS transistor M3 and its vds drain-source voltage. After running the circuit file in Ngspice, with Ngspice session window still open, we enter the following interactive commands to create the new quotient vector, integrate it, and divide by the time interval of integration (for closer comparison to the result above in Section XV A, we use the same time interval delineated by $t_{0.1V} = 167.8\,\text{ps}$ and $t_{0.9V} = 291\,\text{ps}$):

```
ngspice 1 -> let vdsByIdVector =  @m.x3.m3[vds]/@m.x3.m3[id]
ngspice 1 -> meas tran m3IDinteg INTEG vdsByIdVector from=1.678183e-10 to=2.910126e-10
m3IDinteg           =  1.28845e-07 from=  1.67818e-10 to=  2.91013e-10

ngspice 1 -> set deltaT = (2.910126e-10-1.678183e-10)
ngspice 1 -> print m3IDinteg/$deltaT
m3idinteg/(2.910126e-10-1.678183e-10) = 1.045866e+03
```

That $R_{DS} = 1.045\,\text{k}\Omega$ calculated using the averaged integral of the ratio of the $V_{DS}(t)$ and $I_D(t)$ function vectors compares well with the result obtained above in Section XV A using an exponential characterization, estimated average $R_{ds} = 949\,\Omega$, about 10% difference.

XVI. CONCLUSION

We have taken you along on our own exploration of the subject of input capacitance in CMOS logic circuits (and, as is our habit, of anything interesting that came up along the way) and hope you benefit from the account. We have used dynamic measurement in circuit operating context, which is somewhat of a departure from the usual techniques, benefiting from the CBCM approach[1] of Dennis Sylvester and Chenming Hu. CBCM is a useful tool to have available, if for no other reason than to validate the results of simpler approaches.

Appendix A: Non-linear capacitance in the MOSFET

In our simulations, we relied on the BSIM4 MOSFET simulation model to handle the nonlinear variation of MOSFET capacitance associated with the gate, bulk, source and drain terminals as a function of charge transport and voltage. We will attempt here to give a brief description of the relevant portions of the BSIM4 model. First, view the following depictions[30] of the relevant capacitances. Figure 14 presents a cross section of a MOSFET with relevant capacitances.[45]

We find it necessary to maintain a conceptual picture of the MOSFET[31], because almost every writer changes the symbolic term labels as well as the way the models are presented, e.g., the overlap capacitance between gate and source C_{os} in Figure 14 is C_{GSO} (contained in composite term C_{GS}) in the following Figure 15[6]:

In Figure 15 we find that the entity representing capacitance seen at a MOSFET gate G with respect to source S, C_{GS}, is here a composite mixing the intrinsic channel charge relation between gate and source, G_{GCS}, and the extrinsic overlap capacitance C_{GSO} (we grant that it is useful to keep in mind that both components contribute if one is able to recognize the various disguises as it were of the players).

In an equivalent circuit for a MOSFET transient model Figure 16, the junction capacitances (C_{js} and C_{jd} in Figure 14) may be accompanied by diodes also. This is because they arise at the incidental junction (which occurs along the area at the bottom and perimeter of source and drain regions) between n and p semiconductor regions, e.g., the n^+-doped source region and the p-type bulk in which it rests in an NMOS transistor.

1. Intrinsic core capacitance model

Now, with the dotted-line-boxed intrinsic MOSFET region of Figure 14 in mind, we zero in on the four-terminal transcapacitance model of the intrinsic capacitance region, illustrated in simplified form in Figure 17. BSIM4 models the intrinsic capacitance of the four-terminal MOSFET (g gate, s source, b bulk, d drain) using charge assigned to each of those internal

[30] The reader may want to look again at the CMOS figures and discussion in Section IV A and Section IV B above for physical context.

[31] That is, maintain a picture of the bulk MOSFET technology which is our focus. As we mentioned in Section IV though, as of 2021 Intel is already shipping 10 nm FinFET processors, where gates may contact both top and sides of the channel. In other words, technology is changing rapidly. That being said, the principles we discuss remain generally applicable.

(intrinsic) terminals, i.e., the gate charge Q_g, the bulk charge Q_b, the source charge Q_s and the drain charge Q_d. Charge conservation is assured by using these terminal charges rather than terminal voltages as state variables (see §7.2 *Methodology for Intrinsic Capacitance Modeling* [35])

The gate charge Q_g is comprised of mirror charges (see Eq. A1 below re "mirror charges") from the channel charge (Q_{inv}), accumulation charge (Q_{acc}) and substrate depletion charge (Q_{sub}). (see §7.2.1 [35]).

Referring to §7.2.1 [35], accumulation charge Q_{acc} and the substrate charge Q_{sub} are associated with the substrate (what we have generally referred to as the p-type body or bulk semiconductor in our NMOS transistor). The channel charge Q_{inv} derives from the source and drain terminals[35]:

$$\begin{cases} Q_g & = -\left(Q_{sub} + Q_{inv} + Q_{acc}\right) \qquad \text{lhs is "mirrored" on rhs} \\ Q_b & = Q_{acc} + Q_{sub} \qquad \text{depletion charge is included here} \\ Q_{inv} & = Q_s + Q_d \qquad \text{the mobile charge is partitioned between source and drain} \end{cases} \tag{A1}$$

They (BSIM4) divide the substrate charge into a charge at zero source-drain bias and a non-uniform charge in the presence of drain bias, but we are not concerned with that level of detail here. The charge along the channel is integrated to obtain the total gate charge Q_g.

Now that we have an idea of the identity of the four intrinsic MOSFET terminal charges, consider that a two-terminal capacitor has charges Q_1 and Q_2 with $Q_1 = -Q_2$, i.e., the sum of the charges is zero and the charge is a function of the voltage difference between the two terminals $V_{12} = V_1 - V_2$. The small-signal characteristic of such a capacitor is completely described by $C = dQ_1/dV_{12}$.[47] (a form we used in Eq. 2, in Section V above).

With a four-terminal capacitor, the charges on the four terminals must sum to zero (that is, charge conservation requires $Q_1+Q_2+Q_3+Q_4 = 0$) and though these charges must depend on voltage differences between the terminals, they are otherwise arbitrary functions.[47][48]

One may consider the four charges separately as functions of the four terminal voltages, $Q_1(V_1, V_2, V_3, V_4) \ldots Q_4(V_1, V_2, V_3, V_4)$. The partial derivatives form a four-by-four matrix, $\partial Q_i/\partial V_j |\, i = 1, \ldots 4, j = 1, \ldots 4$ (which we present explicitly in Eq. A4 below). In the

present context, the indices will be, rather than numbers, g, d, s, b, signifying gate, drain, source and bulk MOSFET intrinsic terminals.

Such a matrix has a direct interpretation in terms of AC measurements. If an AC voltage signal is applied to terminal j with the other terminals AC grounded, and AC current into terminal i is measured, the current is $i2\pi f \cdot \partial Q_i/\partial V_j$ (i is $\sqrt{-1}$).[47][48]

Charge conservation (equivalent to observing Kirchoff's current law[32]) requires that each column sum to zero and for the matrix to be reference independent (charges can only depend on voltage differences) each row must sum to zero.[45] In general, the matrix is not symmetrical: $\partial Q_i/\partial V_j$ need not equal $\partial Q_j/\partial V_i$. For example, $C_{gd} = \partial Q_g/\partial V_d$ represents the flow of current from the gate in response to a change in drain voltage (Miller feedback). On the other hand, $C_{dg} = \partial Q_d/\partial V_g$ represents a capacitive current flowing from the drain in response to a change in gate voltage (Miller feedthrough).[48][47]

The terminal input capacitances (the four terminals CGG, CSS, CDD, CBB in Figure 17) are the diagonal matrix entries, i.e., where $i = j$:

$$C_{ii} = \partial Q_i/\partial V_i \quad i = j \tag{A2}$$

and the transcapacitances are the negative of off-diagonal entries

$$C_{ij} = -\partial Q_i/\partial V_j \quad i \neq j \tag{A3}$$

Arranging those terms in the four-by-four matrix we mentioned earlier, we see:

$$M = \begin{bmatrix} C_{gg} & -C_{gd} & -C_{gs} & -C_{gb} \\ -C_{dg} & C_{dd} & -C_{ds} & -C_{db} \\ -C_{sg} & -C_{sd} & C_{ss} & -C_{sb} \\ -C_{bg} & -C_{bd} & -C_{bs} & C_{bb} \end{bmatrix} \tag{A4}$$

where the rows are all partial derivatives of charge Q_i with respect to the column index j voltage V_j. For example $-C_{gd}$, the second position in row one, is $C_{ij} = -dQ_g/dV_d$, the partial derivative of gate charge Q_g with respect to drain terminal voltage V_d (which we mentioned above is also known as Miller feedback). The minus signs indicate the result of explicitly applying the Kronecker delta δ_{ij} to each entry of the matrix, i.e., the elements

[32] As Richard Feynman put it, "conservation of charge requires that any charge which leaves one circuit element immediately enters some other circuit element...we require that the algebraic sum of the currents which enter any given junction must be zero."[49]

with equal indices ($i = j$) receive a positive sign while all others ($i \neq j$) receive a minus sign.

The complete MOSFET intrinsic large-signal equivalent circuit based on this transcapacitance matrix is then Figure 18:

2. Model with extrinsic and intrinsic components

The complete MOSFET large-signal equivalent circuit is depicted in Figure 19. In that figure, the intrinsic model of Figure 18 is replaced (mostly) by a dotted-line area labeled "core intrinsic." This is approximately what is occurring in the BSIM4 model. The cited paper[46] provides another figure which is more specific in how the matrix transcapacitances are incorporated in the equivalent circuit containing both intrinsic and extrinsic components, but that includes ten more equations relating the sixteen elements of the matrix in Eq. A4 above (and introducing several new composite terms). We do not believe this is going to be helpful for the reader in obtaining a general idea of how the BSIM4 models MOSFETs.

3. BSIM4 RF high-speed settings

We should note that we are running the BSIM4 simulations (refer to the manual for BSIM4 thoughout our discussion [35]) with the high-speed RF (radio frequency) models enabled, i.e., with a charge-deficit non-quasi-static (NQS) model and substrate resistance network model (looking approximately like the resistors in Figure 19).

We set trnqsMod=1 [33] to turn on the charge-deficit NQS model for transient simulation. Other simulation models often ignore the finite time required for charge to build up along the MOSFET channel, i.e., quasi-static models assume an instantaneous charging of the inversion layer, adversely affecting accuracy of high-speed simulations.[45] BSIM4 NQS uses an Elmore[50] equivalent circuit to model this delay time[34]. An internal node $Q_{def}(t)$ tracks the deficit/surplus channel charge necessary to reach equilibrium, i.e., Q_{def} will decay exponentially into the channel with NQS relaxation time τ (based on the Elmore RC characterization).[45]

[33] We are talking about settings made in the model cards, see Appendix G and Appendix H.

[34] The Elmore delay is equivalent to the first-order time constant of the network[6], i.e., the lowest frequency pole of the original RC circuit characterizing the network is retained.[45]

We set rbodyMod=1 to turn on a five-resistance substrate network to model high frequency coupling through the substrate. We set rgateMod = 1, which generates a bias-independent internal electrode gate resistance (see §9.2 [35]).

BSIM4 also applies a charge-thickness model (CTM) to correct overestimates of intrinsic capacitance. Without CTM, this discrepancy is more pronounced in thinner gate oxide devices because it is assumed that the inversion and accumulation charge are located immediately at the interface of semiconductor and oxide of gate, whereas numerical quantum simulations show significant variation of the charge distribution with depth into the bulk semiconductor region below the oxide interface. CTM introduces a capacitance in series with the oxide capacitance and an analytical DC charge thickness model based on numerical solutions of the applicable Schrödinger, Poisson and Fermi-Dirac equations (§7.3 [35]).

4. A look under the hood

We will caution at the outset that in analogy to having the thermodynamic equations for a Ferrari engine, the practical use of the equations is in the operation of the car, not in attempting to relate internal abstracted details to the, e.g., acceleration of the car (unless you are a Ferrari design engineer). So too should one consider the internal BSIM4 variables that are available in some SPICE implementations, e.g., our Ngspice-27 software, i.e., the point of having a model in software is to avoid having to try to do complicated and tedious manual calculations.

That being said, BSIM4v4 makes it possible to save all sixteen transcapacitance values (see Chapter 31 of the Ngspice-27 Manual, Model and Device Parameters[18]), i.e., the vectors containing the instantaneous values of the elements of the matrix in Eq. A4 during each time step of a simulation, as we did with the MOSFET vds and id internal vectors above in Section XV B.

The values along the diagonal of the matrix in Eq. A4 are the terminal capacitance values (from the partial derivatives discussed above in Section A 1) for the gate, drain, source and body, so might be expected to produce positive values correlated directly with capacitance seen at those terminals. It is easy to remember their Ngspice BSIM4 internal parameter labels because being on the diagonal, the two indexes will be equal, e.g., the capacitance of the gate terminal is cgg, capacitance of the drain terminal is cdd.

The off-diagonal matrix elements, e.g., cgs and csg, on the other hand, are transcapacitances and are typically negative and not recognizable as capacitance as one might expect to observe at a terminal (see the pairs of capacitors surrounding the core terminal capacitances in Figure 18). As we noted earlier, [46] provides a circuit in their Fig. 6 and their equations (8a) through (10e) that suggest one way of interpreting the transcapacitances in terms of more or less recognizable equivalent capacitances (composed of combinations of transcapacitances) associated with four current sources (if you want to dive into that subject in more depth).[35]

a. Graph BSIM4 internal capacitances

In any case, we may save the intrinsic core capacitances and the relatively constant body-drain and body-source diode capacitances of the PMOS transistors of our CMOS INV_X1 inverters with the following Ngspice code (see Ngpice manual [18] §31.6.9.1):

```
save all @m.x2.mp1[cgg] @m.x1.mp1[cgg]
+ @m.x2.mp1[cdd] @m.x1.mp1[cdd]
+ @m.x2.mp1[css] @m.x1.mp1[css]
+ @m.x2.mp1[cbb] @m.x1.mp1[cbb]
+ @m.x2.mp1[capbd] @m.x1.mp1[capbd]
+ @m.x2.mp1[capbs] @m.x1.mp1[capbs]
```

As in Section IX, we execute an Ngspice circuit file (with the code above inserted in the control section) with two CMOS inverters, gate signals at 1 GHz, load a calibration capacitor of 1.5 fF on the test inverter x2 (refer to schematic Fig. 5). We will look briefly at the effective capacitance seen at the PMOS gate during the PMOS turn-off event we examined in Section X). We will plot the capacitance values first. To put the observations in context, we repeat the PMOS gate waveform that turns off the transistor at $t = 1$ ns through $t = 1.1$ ns Figure 20:

The Ngspice code to plot that figure (either interactive in terminal or inserted in circuit file control section) is:

[35] It is annoying that BSIM4 is ostensibly open source, yet the details of the capacitance model are only available in a book for purchase.[51] Much of what we presented in Section A 1 was from sources other than the BSIM4 manual.

```
plot inpmos xl 0.95n 1.15n
```

Next we plot the instantaneous gate, source and drain capacitance values associated with x1 and x2 inverters in the simulation interval around PMOS turnoff at $t = 1$ ns Figure 21:

The Ngspice code to plot that figure (either interactive in terminal or inserted in circuit file control section) is:

```
plot @m.x2.mp1[cgg] @m.x1.mp1[cgg]
+ @m.x2.mp1[cdd] @m.x1.mp1[cdd]
+ @m.x2.mp1[css] @m.x1.mp1[css]
+ xl 0.95n 1.15n
```

We now plot the cbb body terminal capacitance from Eq. A4, as well as the extrinsic source-body and drain-body junction diode capacitances, capbs and capbd Figure 22. For context, capbs and capbd are shown as junction capacitances C_{js} and C_{jd} in Figure 14.

With the following Ngspice code we summed the average of these BSIM4 internal capacitances for the unloaded x1 inverter PMOS transistor over our PMOS turnoff interval:

```
echo "sum of average Ward-Dutton capacitances along diagonal "
echo "and source-body and drain-body junction diode capacitances "
 echo "from beginnng of PMOS turnoff to end of turnoff x1 inv"
 echo ""
 meas tran capavgcgg AVG @m.x1.mp1[cgg]  from=1n to=1.1n
 meas tran capavgcdd AVG @m.x1.mp1[cdd]  from=1n to=1.1n
 meas tran capavgcss AVG @m.x1.mp1[css]  from=1n to=1.1n
 meas tran capavgcbb AVG @m.x1.mp1[cbb]  from=1n to=1.1n
 meas tran cavgcapbd AVG @m.x1.mp1[capbd]  from=1n to=1.1n
 meas tran cavgcapbs AVG @m.x1.mp1[capbs]  from=1n to=1.1n

 echo "sum of average capacitances during turnoff (F):"
 print capavgcgg + capavgcdd + capavgcss + capavgcbb + cavgcapbd + cavgcapbs
```

We obtain terminal output:

```
sum of average Ward-Dutton capacitances along diagonal
```

```
and source-body and drain-body junction diode capacitances
from beginnng of PMOS turnoff to end of turnoff x1 inv

capavgcgg           =  2.876370e-16 from=  1.000000e-09 to=  1.100005e-09
capavgcdd           =  2.316269e-16 from=  1.000000e-09 to=  1.100005e-09
capavgcss           =  5.021677e-17 from=  1.000000e-09 to=  1.100005e-09
capavgcbb           =  7.716284e-17 from=  1.000000e-09 to=  1.100005e-09
cavgcapbd           =  3.166634e-16 from=  1.000000e-09 to=  1.100005e-09
cavgcapbs           =  5.040000e-16 from=  1.000000e-09 to=  1.100005e-09
sum of average capacitances during turnoff (F):
capavgcgg + capavgcdd + capavgcss + capavgcbb + cavgcapbd + cavgcapbs
 = 1.467307e-15
```

The sum of the averaged capacitances in turnoff above is 1.467 fF, These variables, of course, are not intended to be summed in this way, e.g., capbd is usually considered to be a self-load on the *output* of a gate (see §5.4.1 [6], for example). Also, the input capacitance of the inverter would include the parallel input capacitance of the paired NMOS transistor gate (the pull-down transistor in the inverter circuit).

It does appear though that a subset of the applicable capacitances of the PMOS transistor shown added with those of the paired NMOS (not measured) would be consistent with the suggested total gate input capacitance of suggested of this standard inverter typical of 1.7 fF in the stdcells-databook.pdf[36]

b. Graph BSIM4 internal charge variables

It is perhaps also instructive to look at the charge variables in order to cut through some of the abstraction in Section A 1. We may add the following commands to the control section of one of our circuit files:

```
save all @m.x1.mp1[qg] @m.x1.mp1[qs] @m.x1.mp1[qd]
+  @m.x1.mp1[qinv]  @m.x1.mp1[qb]
+ @m.x2.mp1[qg] @m.x2.mp1[qs] @m.x2.mp1[qd]
```

[36] We briefly discussed the stdcells-databook.pdf in Section IV C and Section VIII A.

```
+  @m.x2.mp1[qinv]  @m.x2.mp1[qb]
```

From Section A 1 and Eq. A1 you will recognize the charge variables. We insert some additional commands in the control section to obtain the charge before turnoff and following (for inverter x1, which is indistinguishable at this scale from x2), compare the gate charge Q_g lost with that of the sum of Q_b and $Q_s + Q_d$ and plot the saved charges. A portion of the terminal result is copied here:

```
qq gate positive charge loss at turnoff:
(qgval1 - qgval2) = 3.123873e-16
approx. equals qb + qs + qd loss negative charge
( (qbval1 - qbval2) + qsval1 + qdval1 ) = -3.12398e-16
```

We see that the sum of Q_b and $Q_s + Q_d$ lost during PMOS turnoff in x1 from $t = 0.99\,\text{ns}$ to $t = 1.1\,\text{ns}$ was $-312.398\,\text{aC}$ (atto Coulomb) and the loss of positive gate charge Q_g over that interval was almost identical, $312.3873\,\text{aC}$, consistent with Eq. A1. This is observed in the plot of these variables in Figure 23:

We did plot the difference of these charge variables for x1 and x2, looking for obvious change in the BSIM4 MOSFET charge behavior between unloaded and loaded inverters to account for the mohawk injection current we examined in Section X, but found only some small variation less than $0.5\,\text{aC}$ that occurred briefly at the beginning and end of the turnoff period. It seemed unlikely this would produce the $\sim 51\,\text{aC}$ excess charge dumped by x1, a value we obtained by integrating the drain and source currents of both inverters over the relevant PMOS turnoff interval, then taking the difference in resulting charge.

Appendix B: Original CBCM measure discrete cap Ngspice-27 circuit code

The following is the Ngspice-27 circuit file named 20211209CBCMinvOnly.cir. It implements the original CBCM method (without correction for mohawk injection current) measuring a discrete load capacitor.

```
CHARGE BASED CAPACITANCE MEAS OF DISCRETE C WITH INVERTER PAIR
* test a discrete capacitor with CBCM method
* two inverter CBCM charge-based capacitance measurement setup
```

```
* the 2nd inverter is the test vehicle which charges load C

* see
* Analytical Modeling and Characterization
* of Deep-Submicrometer Interconnect
* by Dennis Sylvester and Chenming Hu, PROCEEDINGS OF THE IEEE,
* VOL. 89, NO. 5, MAY 2001

* using Ngspice-27 Creation Date: Tue Dec 26 17:10:20 UTC 2017
* using BSIM4 level=54 mosfet models from  process file
* /FreePDK45/ncsu_basekit/models/hspice
* /tran_models/models_nom/NMOS_VTL.inc
* in which we updated the version statement from 4.0 to 4.8
* and enabled RF high speed support

* the W and L dimensions of INV_X1
* come from NCSU FreePDK 45nm
* used by Christopher Torng 2019 for
* a 45nm ASIC design kit for mflowgen
* we use ngspice 25 degree C default circuit temperature,
* and vdd 1.10v per
* NangateOpenCellLibrary_typical
* Build Date: Thursday Feb 17 15:07 2011 library

********************************
* PARAMETERS for command lines,
* CSPARAMETERS for lines within control section or echos/prints
* refer to ngspice-27 manual for syntax

* Simulation time setup
* desired pulse train frequency in Hz
.param testfreq=1000e6
```

```
* printable as MHz
.csparam testfreqPrintable={testfreq/1e6}
* the period is then 1/frequency in seconds
.param testperiod='1/testfreq'
.csparam testperiodctl={testperiod}
*.csparam testperiodSmallerForPlotctl={testperiod*0.7}
.csparam testperiodSmallerForPlotctl={testperiod*1.1}
* how many cycles at specified frequency should be analyzed
.param testcycles='5'
* length of the simulation in seconds will be
* number of cycles times period
.param ttime='testperiod*testcycles'
.csparam ttimectl={ttime}
* how many simulation steps desired:
.param numberOfStepsInSim=100000
* resolution is length of simulation divided by steps
.param transstepsize='ttime/numberOfStepsInSim'

* power supply DC voltage, using
* Nangate typical corner Vdd=1.10 v, see
* NangateOpenCellLibrary_typical Build Date: Thursday Feb 17 15:07 2011
.param DCsupplyVoltage=1.10
* want to space multiple voltage plots clearly,
* so get ceiling of DCsupplyVoltage
.param PlotMultiVoltsSpacer=(DCsupplyVoltage+.9)
* need a csparam form to use in the control section
.csparam PlotMultiVoltsSpacerForCTL={PlotMultiVoltsSpacer}
* similarly, need DCsupplyVoltage param that works in control section
.csparam DCsupplyVoltageForCTL={DCsupplyVoltage}
* for y limit if plot n traces together, say n=4
.param numOfVoltTraces=4
.csparam DCsupplyVoltageForYLIM={PlotMultiVoltsSpacer*numOfVoltTraces}
```

```
* two-phase pulse train input stimulus parameters
* note rise/fall based on 20% of PMOS pulse width
.param RiseFallPercent=0.20
.param HalfPeriod='(testperiod/2)'
*.param PulseWpmos='(HalfPeriod)-(HalfPeriod*2*RiseFallPercent)'
.param PulseWpmos='(HalfPeriod)'
.param TdelayBeginNMOS='(HalfPeriod*RiseFallPercent*1.1)'
*.param PulseWnmos='(HalfPeriod)-(HalfPeriod*4*RiseFallPercent)'
.param PulseWnmos='(HalfPeriod)-(HalfPeriod*2.2*RiseFallPercent)'
.param TriseOrFall='(PulseWpmos*RiseFallPercent)'
.csparam TriseOrFallPrintable={TriseOrFall*1}

* test capacitance to be measured (if make changes below)
.param TestCapacitance=1.5fF
.csparam TestCapacitancePrintable={TestCapacitance*1e15}

************************************
* BSIM4 level=54 mosfet models from
* 2006 45nm_bulk.pm process file
* which we updated the version statement from 4.0 to 4.8

.include modelcard.nmos
.include modelcard.pmos

* use NMOS_VTL for the model name nmos in instantiation
* use PMOS_VTL for the model name pmos in instantiation
************************************

* MANDATORY SEPARATE POWER SUPPLIES for each inverter

* inverter 1 power supply:
```

```
vdd1 dd1 0 {DCsupplyVoltage}
vss1 ss1 0 dc 0
ve1  sub1  0 dc 0
vpe1 well1 0 {DCsupplyVoltage}

* inverter 2 power supply:
vdd2 dd2 0 {DCsupplyVoltage}
vss2 ss2 0 dc 0
ve2  sub2  0 dc 0
vpe2 well2 0 {DCsupplyVoltage}

*****************************************
* INVERTER SUBCIRCUIT definition

* Reminder: nd ng ns nb is the usual MOSFET lead order

* drive the PMOS gate separately from the NMOS
* one input each of PUN and PDN transistors of the inverter

.subckt inverter dd ss sub well inp inn out

mn1  out inn  ss  sub  NMOS_VTL W=0.415000U L=0.050000U

mp1  out inp  dd  well  PMOS_VTL W=0.630000U L=0.050000U

* used W and L dimensions of INV_X1 from NCSU FreePDK 45nm
* see Christopher Torng 2019 45nm ASIC design kit for mflowgen

.ends inverter
*****************************************

*****************************************
```

```
* INSTANTIATE 2 INVERTERS

* make 2 inverter instances using the subckt inverter above

* instantiate reference inverter (no load) inverter 1:
* dd ss sub well in out
X1 dd1  ss1  sub1  well1 inPMOS inNMOS out1 inverter

* instantiate test inverter 2 (load with target capacitance):
* dd ss sub well in out
X2 dd2  ss2  sub2  well2 inPMOS inNMOS out2 inverter

*****************************************

*****************************************
* MEASUREMENT CONNECTIONS

* measure test capacitor of specified value TestCapacitance
* driven by test inverter X2 out2

* hang a test cap off the output of the 2nd inverter
CTEST2 out2 0 {TestCapacitance}

*****************************************
* CONTROL SECTION

.control

* syntax note: initial plus sign is extension of previous line
* to keep columns less than equal 66 for printing

* can save NMOS PMOS transistor internal model vectors
```

```
* if desired, see ngspice-27 manual chap 31,
* section 31.6.9 BSIM4
* for example:
* save all  @m.x1.mp1[id] @m.x2.mp1[id]
*+@m.x1.mp1[ibs] @m.x2.mp1[ibs] @m.x1.mp1[ibd] @m.x2.mp1[ibd]
*+ @m.x1.mp1[isub] @m.x2.mp1[isub] @m.x1.mp1[igs] @m.x2.mp1[igs]

* run transient analysis defined outside control section
 run
* make plot white background instead of default black
 set color0 = white ; plot window -background color
 set color1 = black ; plot window -grid and text color
* thinner grid and plot lines?
 set xbrushwidth=0.5

* show "the old in out" (Clockwork Orange, Malcolm McDowell line)
plot inPMOS
+ inNMOS+$&PlotMultiVoltsSpacerForCTL
+ out1+($&PlotMultiVoltsSpacerForCTL*2)
+ out2+($&PlotMultiVoltsSpacerForCTL*3)
+ xl 0 $&testperiodSmallerForPlotctl
+ yl 0 $&DCsupplyVoltageForYLIM
+ title "CBCM measure calibration capacitor"

* plot inverter x1 in, out and currents
* this is not general code like the above,
* so you may need to manually adjust if change
* run parameters
plot vdd1#branch vss1#branch
+ inpmos*(50e-6)+200e-6 innmos*(50e-6)+300e-6
+ out1*(50e-6)+400e-6 xl 0 $&testperiodctl
+ title "CBCM apparatus run x1 ref inverter"
```

```
* plot inverter x2 in, out and currents
* this is not general code so you may need
* to manually adjust if change run parameters
plot vdd2#branch vss2#branch
+ inpmos*(50e-6)+200e-6 innmos*(50e-6)+300e-6
+ out2*(50e-6)+400e-6 xl 0 $&testperiodctl
+ title "CBCM apparatus run x2 test inverter (loaded)"

* measure avg vdd current in each inverter
* from t=0 to end of simulation
 meas tran inv1rmsCur AVG vdd1#branch from=0 to=$&ttimectl
 meas tran inv2rmsCur AVG vdd2#branch from=0 to=$&ttimectl

* use difference of two average inverter currents
* in vdd leads to measure capacitance load on one;
* formally,
* C=(input_period*abs(avg_inv1_current-avg_inv2_current)/Vdd
 let capmeasured= $&testperiodctl * abs(inv2rmsCur-inv1rmsCur)/
+               $&DCsupplyVoltageForCTL
 print capmeasured

 echo "Actual test capacitance on 2nd inverter was "
 echo $&TestCapacitancePrintable " fF"

 echo "Test frequency was " $&testfreqPrintable " MHz"
 echo "Input rise/fall time was " $&TriseOrFallPrintable " s"
 echo "Vdd supply voltage was "
 echo $&DCsupplyVoltageForCTL " volts DC"

.endc
* END CONTROL SECTION
```

```
*******************************************

*******************************************
* INVERTER PULSE INPUTS

* pulse both inverters' PMOS gates with VDD voltage

VIN1 inPMOS 0 PULSE(0 {DCsupplyVoltage}
+ 0 {TriseOrFall} {TriseOrFall} {PulseWpmos} {testperiod})

* pulse both inverters' NMOS input with VDD, but--
* want NMOS off while PMOS turns on or off, so gets
* delayed and smaller pulse width PulseWnmos

VIN2 inNMOS 0 PULSE(0 {DCsupplyVoltage} {TdelayBeginNMOS}
+ {TriseOrFall} {TriseOrFall} {PulseWnmos}
+ {testperiod})

*****************************************
* transient analysis
.tran 'transstepsize' 'ttime'

*****************************************
.END
```

Appendix C: Revised CBCM measure discrete cap Ngspice-27 circuit code

The following is the Ngspice-27 circuit file named 20211209CBCMinvOnlyStripMohawk.cir. It implements the CBCM method with correction for mohawk injection current (as described in Section XI) measuring a discrete load capacitor.

```
REVISED CBCM MEAS DISCRETE C WITH INVERTER PAIR

* test a discrete capacitor with CBCM method
* two inverter CBCM charge-based capacitance measurement setup
* the 2nd inverter is the test vehicle which charges load C

* see
* Analytical Modeling and Characterization
* of Deep-Submicrometer Interconnect
* by Dennis Sylvester and Chenming Hu, PROCEEDINGS OF THE IEEE,
* VOL. 89, NO. 5, MAY 2001

* In this version, we implement Ngspice control structures
* to strip the negative current on pmos turn-off (our mohawk)
* otherwise known in the literature as negative return current
* This mohawk introduced error ~7% in the original CBCM method

* using Ngspice-27 Creation Date: Tue Dec 26 17:10:20 UTC 2017
* using BSIM4 level=54 mosfet models from  process file
* /FreePDK45/ncsu_basekit/models/hspice
* /tran_models/models_nom/NMOS_VTL.inc
* in which we updated the version statement from 4.0 to 4.8
* and enabled RF high speed support

* the W and L dimensions of INV_X1
* come from NCSU FreePDK 45nm
* used by Christopher Torng 2019 for
* a 45nm ASIC design kit for mflowgen
* we use ngspice 25 degree C default circuit temperature,
* and vdd 1.10v per
* NangateOpenCellLibrary_typical
* Build Date: Thursday Feb 17 15:07 2011 library
```

```
*********************************
* PARAMETERS for command lines,
* CSPARAMETERS for lines within control section or echos/prints
* refer to ngspice-27 manual for syntax

* Simulation time setup
* desired pulse train frequency in Hz
.param testfreq=1000e6
* printable as MHz
.csparam testfreqPrintable={testfreq/1e6}
* the period is then 1/frequency in seconds
.param testperiod='1/testfreq'
.csparam testperiodctl={testperiod}
*.csparam testperiodSmallerForPlotctl={testperiod*0.7}
.csparam testperiodSmallerForPlotctl={testperiod*1.1}
* how many cycles at specified frequency should be analyzed
.param testcycles='5'
* length of the simulation in seconds will be
* number of cycles times period
.param ttime='testperiod*testcycles'
.csparam ttimectl={ttime}
* how many simulation steps desired:
.param numberOfStepsInSim=100000
* resolution is length of simulation divided by steps
.param transstepsize='ttime/numberOfStepsInSim'

* power supply DC voltage, using
* Nangate typical corner Vdd=1.10 v, see
* NangateOpenCellLibrary_typical Build Date: Thursday Feb 17 15:07 2011
.param DCsupplyVoltage=1.10
```

```
* want to space multiple voltage plots clearly,
* so get ceiling of DCsupplyVoltage
.param PlotMultiVoltsSpacer=(DCsupplyVoltage+.9)
* need a csparam form to use in the control section
.csparam PlotMultiVoltsSpacerForCTL={PlotMultiVoltsSpacer}
* similarly, need DCsupplyVoltage param that works in control section
.csparam DCsupplyVoltageForCTL={DCsupplyVoltage}
* for y limit if plot n traces together, say n=4
.param numOfVoltTraces=4
.csparam DCsupplyVoltageForYLIM={PlotMultiVoltsSpacer*numOfVoltTraces}

* two-phase pulse train input stimulus parameters
* note rise/fall based on 20% of PMOS pulse width
.param RiseFallPercent=0.20
.param HalfPeriod='(testperiod/2)'
*.param PulseWpmos='(HalfPeriod)-(HalfPeriod*2*RiseFallPercent)'
.param PulseWpmos='(HalfPeriod)'
.param TdelayBeginNMOS='(HalfPeriod*RiseFallPercent*1.1)'
*.param PulseWnmos='(HalfPeriod)-(HalfPeriod*4*RiseFallPercent)'
.param PulseWnmos='(HalfPeriod)-(HalfPeriod*2.2*RiseFallPercent)'
.param TriseOrFall='(PulseWpmos*RiseFallPercent)'
.csparam TriseOrFallPrintable={TriseOrFall*1}

* test capacitance to be measured (if make changes below)
.param TestCapacitance=1.5fF
.csparam TestCapacitancePrintable={TestCapacitance*1e15}

************************************
* BSIM4 level=54 mosfet models from
* 2006 45nm_bulk.pm process file
* which we updated the version statement from 4.0 to 4.8
```

```
.include modelcard.nmos
.include modelcard.pmos

* use NMOS_VTL for the model name nmos in instantiation
* use PMOS_VTL for the model name pmos in instantiation
************************************
* MANDATORY SEPARATE POWER SUPPLIES for all circuits

* inverter 1 power supply:
vdd1 dd1 0 {DCsupplyVoltage}
vss1 ss1 0 dc 0
ve1  sub1  0 dc 0
vpe1 well1 0 {DCsupplyVoltage}

* inverter 2 power supply:
vdd2 dd2 0 {DCsupplyVoltage}
vss2 ss2 0 dc 0
ve2  sub2  0 dc 0
vpe2 well2 0 {DCsupplyVoltage}

*****************************************
* INVERTER SUBCIRCUIT definition

* Reminder: nd ng ns nb is the usual MOSFET lead order

* drive the PMOS gate separately from the NMOS
* one input each of PUN and PDN transistors of the inverter

.subckt inverter dd ss sub well inp inn out

mn1  out inn  ss  sub  NMOS_VTL W=0.415000U L=0.050000U
```

```
mp1  out inp  dd  well  PMOS_VTL W=0.630000U L=0.050000U

* used W and L dimensions of INV_X1 from NCSU FreePDK 45nm
* see Christopher Torng 2019 45nm ASIC design kit for mflowgen

.ends inverter

*********************************************

*********************************************
* INSTANTIATE 2 INVERTERS

* make 2 inverter instances using the subckt inverter above

* instantiate reference inverter (no load) inverter 1:
* dd ss sub well in out
X1 dd1  ss1  sub1  well1 inPMOS inNMOS out1 inverter

* instantiate test inverter 2 (load with target capacitance):
* dd ss sub well in out
X2 dd2  ss2  sub2  well2 inPMOS inNMOS out2 inverter

*********************************************

*********************************************
* MEASUREMENT CONNECTIONS

* measure test capacitor of specified value TestCapacitance
* driven by test inverter X2 out2

* hang a test cap off the output of the 2nd inverter
CTEST2 out2 0 {TestCapacitance}
```

```
*****************************************
* CONTROL SECTION

.control

* syntax note: initial plus sign is extension of previous line
* to keep columns less than equal 66 for printing

* can save NMOS PMOS transistor internal model vectors
* if desired, see ngspice-27 manual chap 31,
* section 31.6.9 BSIM4
* for example:
* save all  @m.x1.mp1[id] @m.x2.mp1[id]
*+@m.x1.mp1[ibs] @m.x2.mp1[ibs] @m.x1.mp1[ibd] @m.x2.mp1[ibd]
*+ @m.x1.mp1[isub] @m.x2.mp1[isub] @m.x1.mp1[igs] @m.x2.mp1[igs]

* run transient analysis defined outside control section
 run
* make plot white background instead of default black
 set color0 = white ; plot window -background color
 set color1 = black ; plot window -grid and text color
* thinner grid and plot lines?
 set xbrushwidth=0.5

* show "the old in out" (Clockwork Orange, Malcolm McDowell line)
plot inPMOS
+ inNMOS+$&PlotMultiVoltsSpacerForCTL
+ out1+($&PlotMultiVoltsSpacerForCTL*2)
+ out2+($&PlotMultiVoltsSpacerForCTL*3)
+ xl 0 $&testperiodSmallerForPlotctl
+ yl 0 $&DCsupplyVoltageForYLIM
```

```
+ title "CBCM meas. calibration cap, mohawk stripped"

****
* BEGIN SHAVE THE MOHAWK
* strip the asymmetrical negative return
*currents on pmos turn-off

* get length of vectors to process in loop
let samplelen=length(vdd1#branch)

* create copy of current vectors vdd1#branch,
* vdd2#branch to clean of negative return currents
let vd1brclean = vdd1#branch
let vd2brclean = vdd2#branch

* loop through the new current vectors and
* strip any positive current
* remembering that ngspice views negative return current
* to power supply as positive

* init loop counter and vector index
let loop = 0

* while loop counter is less than the length of
*  the current vectors (begin with index 0)
* check for a positive current value and replace
* it with zero as not relevant to charging C

  while loop < samplelen
    if vd1brclean[loop] > 0
      let vd1brclean[loop] = 0
    end
```

```
    if vd2brclean[loop] > 0
      let vd2brclean[loop] = 0
    end

    let loop = loop + 1
  end
* END SHAVE MOHAWK
****

* measure avg vdd current in each inverter
* from t=0 to end of simulation
* using the clean current vectors we made above:

 meas tran inv1rmsCur AVG vd1brclean from=0 to=$&ttimectl
 meas tran inv2rmsCur AVG vd2brclean from=0 to=$&ttimectl

* use difference of two average inverter currents
* in vdd leads to measure capacitance load on one;
* formally,
* C=(input_period*abs(avg_inv1_current-avg_inv2_current)/Vdd
 let capmeasured= $&testperiodctl * abs(inv2rmsCur-inv1rmsCur)/
+               $&DCsupplyVoltageForCTL
 print capmeasured

 echo "(injection current or mohawk stripped prior)"

 echo "Actual test capacitance on 2nd inverter was "
 echo $&TestCapacitancePrintable " fF"

 echo "Test frequency was " $&testfreqPrintable " MHz"
 echo "Input rise/fall time was " $&TriseOrFallPrintable " s"
```

```
 echo "Vdd supply voltage was "
 echo $&DCsupplyVoltageForCTL " volts DC"

.endc
*******************************************

*******************************************
* INVERTER PULSE INPUTS

* pulse both inverters' PMOS gates with VDD voltage

VIN1 inPMOS 0 PULSE(0 {DCsupplyVoltage}
+ 0 {TriseOrFall} {TriseOrFall} {PulseWpmos} {testperiod})

* pulse both inverters' NMOS input with VDD, but--
* want NMOS off while PMOS turns on or off, so gets
* delayed and smaller pulse width PulseWnmos

VIN2 inNMOS 0 PULSE(0 {DCsupplyVoltage} {TdelayBeginNMOS}
+ {TriseOrFall} {TriseOrFall} {PulseWnmos}
+ {testperiod})

*****************************************
* transient analysis
.tran 'transstepsize' 'ttime'

*****************************************
.END
```

Appendix D: Revised CBCM measure NAND cap Ngspice-27 circuit code

The following is the Ngspice-27 circuit file named 20211209CBCMmeasNANDStripMohawk.cir. It implements the CBCM method with correction for mohawk injection current (as described in Section XI) measuring the input A1 (labeled in1 in schematic 1) of the NAND2_X1 circuit.

```
CHARGE BASED CAPACITANCE MEAS OF NAND INPUT CAP
* measure input capacitance of NAND2_X1 using CBCM method
* two inverter CBCM charge-based capacitance measurement setup
* the 2nd inverter is the test vehicle which charges load
* capacitance, i.e., the input of the NAND2_X1 gate

* see
* Analytical Modeling and Characterization
* of Deep-Submicrometer Interconnect
* by Dennis Sylvester and Chenming Hu, PROCEEDINGS OF THE IEEE,
* VOL. 89, NO. 5, MAY 2001

* In this version, we implement Ngspice control structures
* to strip the negative current on pmos turn-off (our mohawk)
* otherwise known in the literature as negative return current
* This mohawk introduced error 8.6% in the original CBCM method

* using Ngspice-27 Creation Date: Tue Dec 26 17:10:20 UTC 2017
* using BSIM4 level=54 mosfet models from  process file
* /FreePDK45/ncsu_basekit/models/hspice
* /tran_models/models_nom/NMOS_VTL.inc
* in which we updated the version statement from 4.0 to 4.8
* and enabled RF high speed support

* the W and L dimensions of INV_X1 and NAND2_X1
* come from NCSU FreePDK 45nm
```

```
* used by Christopher Torng 2019 for
* a 45nm ASIC design kit for mflowgen
* we use ngspice 25 degree C default circuit temperature,
* and vdd 1.10v per
* NangateOpenCellLibrary_typical
* Build Date: Thursday Feb 17 15:07 2011 library

********************************
* PARAMETERS for command lines,
* CSPARAMETERS for lines within control section or echos/prints
* refer to ngspice-27 manual for syntax

* Simulation time setup
* desired pulse train frequency in Hz
.param testfreq=1000e6
* printable as MHz
.csparam testfreqPrintable={testfreq/1e6}
* the period is then 1/frequency in seconds
.param testperiod='1/testfreq'
.csparam testperiodctl={testperiod}
*.csparam testperiodSmallerForPlotctl={testperiod*0.7}
.csparam testperiodSmallerForPlotctl={testperiod*1.1}
* how many cycles at specified frequency should be analyzed
.param testcycles='5'
* length of the simulation in seconds will be
* number of cycles times period
.param ttime='testperiod*testcycles'
.csparam ttimectl={ttime}
* how many simulation steps desired:
.param numberOfStepsInSim=100000
* resolution is length of simulation divided by steps
.param transstepsize='ttime/numberOfStepsInSim'
```

```
* power supply DC voltage, using
* Nangate typical corner Vdd=1.10 v, see
* NangateOpenCellLibrary_typical Build Date:
* Thursday Feb 17 15:07 2011
.param DCsupplyVoltage=1.10
* want to space multiple voltage plots clearly,
* so get ceiling of DCsupplyVoltage
.param PlotMultiVoltsSpacer=(DCsupplyVoltage+.9)
* need a csparam form to use in the control section
.csparam PlotMultiVoltsSpacerForCTL={PlotMultiVoltsSpacer}
* similarly, need DCsupplyVoltage param that works in control section
.csparam DCsupplyVoltageForCTL={DCsupplyVoltage}
* for y limit if plot n traces together, say n=5
.param numOfVoltTraces=5
.csparam DCsupplyVoltageForYLIM={PlotMultiVoltsSpacer*numOfVoltTraces}

* two-phase pulse train input stimulus parameters
* note rise/fall based on 20% of PMOS pulse width
.param RiseFallPercent=0.20
.param HalfPeriod='(testperiod/2)'
.param PulseWpmos='(HalfPeriod)'
.param TdelayBeginNMOS='(HalfPeriod*RiseFallPercent*1.1)'
.param PulseWnmos='(HalfPeriod)-(HalfPeriod*2.2*RiseFallPercent)'
.param TriseOrFall='(PulseWpmos*RiseFallPercent)'
.csparam TriseOrFallPrintable={TriseOrFall*1}

* load capacitance for NAND output
.param NANDLoadCapacitance=59fF
.csparam NANDLoadCapacitancePrintable={NANDLoadCapacitance*1e15}

************************************
```

```
* BSIM4 level=54 mosfet models from
* 2006 45nm_bulk.pm process file
* which we updated the version statement from 4.0 to 4.8

.include modelcard.nmos
.include modelcard.pmos

* use NMOS_VTL for the model name nmos in instantiation
* use PMOS_VTL for the model name pmos in instantiation
************************************
* MANDATORY SEPARATE POWER SUPPLIES for all circuits

* inverter 1 power supply:
vdd1 dd1 0 {DCsupplyVoltage}
vss1 ss1 0 dc 0
ve1  sub1  0 dc 0
vpe1 well1 0 {DCsupplyVoltage}

* inverter 2 power supply:
vdd2 dd2 0 {DCsupplyVoltage}
vss2 ss2 0 dc 0
ve2  sub2  0 dc 0
vpe2 well2 0 {DCsupplyVoltage}

* target C NAND gate power supply:
vdd3 vddnand 0 {DCsupplyVoltage}
* the NAND subckt simply uses global gnd node 0

*****************************************
* INVERTER SUBCIRCUIT definition

* Reminder: nd ng ns nb is the usual MOSFET lead order
```

```
* drive the PMOS gate separately from the NMOS
* one input each of PUN and PDN transistors of the inverter

.subckt inverter dd ss sub well inp inn out

mn1  out inn  ss  sub  NMOS_VTL W=0.415000U L=0.050000U

mp1  out inp  dd  well  PMOS_VTL W=0.630000U L=0.050000U

* used W and L dimensions of INV_X1 from NCSU FreePDK 45nm
* see Christopher Torng 2019 45nm ASIC design kit for mflowgen

.ends inverter

*********************************************

*********************************************
* INSTANTIATE 2 INVERTERS


* make 2 inverter instances using the subckt inverter above

* instantiate reference inverter (no load) inverter 1:
* dd ss sub well in out
X1 dd1  ss1  sub1  well1 inPMOS inNMOS out1 inverter

* instantiate test inverter 2 (load with target capacitance):
* dd ss sub well in out
X2 dd2  ss2  sub2  well2 inPMOS inNMOS out2 inverter

*****************************************
```

```
* CREATE the NAND gate whose input gate A will be
* target capacitance loading test inverter X2 of
* CBCM inverter pair

*** SUBCIRCUIT DEFINITION
* global gnd 0 not included in parameter list
.SUBCKT NAND in1 in2 out VDD
*
* PMOS parallel PUN pair
*  nd  ng  ns  nb  model
M3 out in1 vdd vdd PMOS_VTL W=0.630000U L=0.050000U
M4 out in2 vdd vdd PMOS_VTL W=0.630000U L=0.050000U

* NMOS series PDN pair
*   nd   ng  ns    nb model
M1 net.1 in2 0     0  NMOS_VTL W=0.415000U L=0.050000U
M2 out   in1 net.1 0  NMOS_VTL W=0.415000U L=0.050000U
*
.ENDS NAND
* above we use the W and L dimensions of NAND2_X1
* from NCSU FreePDK 45nm
* from Christopher Torng 2019 45nm ASIC design kit for mflowgen
*

* INSTANTIATE a NAND gate X3 using the SUBCKT:
*  in1 in2  out  VDD
X3 in1 in2  out  vddnand NAND

*****************************************

*****************************************
```

```
* MEASUREMENT CONNECTIONS

* measure effective capacitance of the in1 input to NAND gate
* when driven by the test vehicle inverter X2 out2

* provide low resistance path from test inverter X2 out2
* to NAND input 1 in1:
RTOTARGETC out2 in1 0.01

* tie unused NAND input in2 of NAND to vddnand,
* then a LOW to HIGH transition on inverter input
* will present LOW on in1 of NAND, causing NAND out
* to transition from LOW to HIGH, charging its load C
RX32 in2 vddnand 10k

* LOAD the NAND X3 output:
CLOAD out 0 {NANDLoadCapacitance}

*****************************************
* CONTROL SECTION

.control

* syntax note: initial plus sign is extension of previous line
* to keep columns less than equal 66 for printing

* make the PMOS M3 transistor internal BSIM4
* vectors for drain current, id, and vds
* available
save all @m.x3.m3[id] @m.x3.m3[vds]

* run transient analysis defined outside the control section
```

```
 run
* make the plot white background instead of default black
 set color0 = white ; plot window -background color
 set color1 = black ; plot window -grid and text color
* thinner grid and plot lines?
 set xbrushwidth=0.5

* show "the old in out" (Clockwork Orange, Malcolm McDowell line)
plot inPMOS inNMOS+$&PlotMultiVoltsSpacerForCTL
+ out1+($&PlotMultiVoltsSpacerForCTL*2)
+ out2+($&PlotMultiVoltsSpacerForCTL*3)
+ out+($&PlotMultiVoltsSpacerForCTL*4)
+ yl 0 $&DCsupplyVoltageForYLIM
+ xl 0 $&testperiodSmallerForPlotctl
+ title "CBCM measure NAND input C (strip inj current)"

* plot NAND output and in1 NAND input separately
* to examine, e.g., prop delay
plot in1 out xl 0 $&testperiodctl

****
* BEGIN SHAVE THE MOHAWK
* strip the asymmetrical negative return
* currents on pmos turn-off

* get length of vectors to process in loop
let samplelen=length(vdd1#branch)

* create copy of current vectors vdd1#branch,
* vdd2#branch to clean of negative return currents
let vd1brclean = vdd1#branch
let vd2brclean = vdd2#branch
```

```
* loop through the new current vectors and
* strip any positive current
* remembering that ngspice views negative return current
* to power supply as positive

* init loop counter and vector index
let loop = 0

* while loop counter is less than the length of
*  the current vectors (begin with index 0)
* check for a positive current value and replace
* it with zero as not relevant to charging C

  while loop < samplelen
    if vd1brclean[loop] > 0
      let vd1brclean[loop] = 0
    end

    if vd2brclean[loop] > 0
      let vd2brclean[loop] = 0
    end

    let loop = loop + 1
  end
* END SHAVE MOHAWK

****
* measure avg vdd current in each inverter
* from t=0 to end of simulation
* using the clean current vectors we made above:
```

```
 meas tran inv1rmsCur AVG vd1brclean from=0 to=$&ttimectl
 meas tran inv2rmsCur AVG vd2brclean from=0 to=$&ttimectl

* use difference of two average inverter currents
* in vdd leads to measure capacitance load on one;
* formally,
* C=(input_period*abs(avg_inv1_current-avg_inv2_current)/Vdd
 let capmeasured= $&testperiodctl * abs(inv2rmsCur-inv1rmsCur)/
+               $&DCsupplyVoltageForCTL
 print capmeasured

 echo "(injection current or mohawk stripped prior)"

 echo "(equivalent NAND input capacitance being charged"
 echo " by 2nd inverter, Farads)"
 echo "(Nangate typical build library suggests"
 echo "NAND2_X1 A1 input is 1.599 fF)"

 echo "Load capacitance on NAND output:  "
 echo $&NANDLoadCapacitancePrintable " fF"

 echo "Test frequency was " $&testfreqPrintable " MHz"
 echo "Input rise/fall time was " $&TriseOrFallPrintable " s"
 echo "Vdd supply voltage was "
 echo $&DCsupplyVoltageForCTL " volts DC"

.endc
*******************************************

*******************************************
* INVERTER PULSE INPUTS
```

```
* pulse both inverters' PMOS gates with VDD voltage

VIN1 inPMOS 0 PULSE(0 {DCsupplyVoltage}
+ 0 {TriseOrFall} {TriseOrFall} {PulseWpmos} {testperiod})

* pulse both inverters' NMOS input with VDD, but--
* want NMOS off while PMOS turns on or off, so gets
* delayed and smaller pulse width PulseWnmos

VIN2 inNMOS 0 PULSE(0 {DCsupplyVoltage} {TdelayBeginNMOS}
+ {TriseOrFall} {TriseOrFall} {PulseWnmos}
+ {testperiod})

*****************************************
* transient analysis
.tran 'transstepsize' 'ttime'

*****************************************
.END
```

Appendix E: Uncorrected CBCM measure NAND cap Ngspice-27 circuit code

The following is the Ngspice-27 circuit file named 20211209CBCMmeasNAND.cir. It implements the CBCM method without correction for mohawk injection current, measuring the input A1 (labeled in1 in schematic 1) of the NAND2_X1 circuit.

```
CHARGE BASED CAPACITANCE MEAS OF NAND INPUT CAP
* measure input capacitance of NAND2_X1 using CBCM method
* two inverter CBCM charge-based capacitance measurement setup
* the 2nd inverter is the test vehicle which charges load
* capacitance, i.e., the input of the NAND2_X1 gate
```

```
* see
* Analytical Modeling and Characterization
* of Deep-Submicrometer Interconnect
* by Dennis Sylvester and Chenming Hu, PROCEEDINGS OF THE IEEE,
* VOL. 89, NO. 5, MAY 2001

* using Ngspice-27 Creation Date: Tue Dec 26 17:10:20 UTC 2017
* using BSIM4 level=54 mosfet models from  process file
* /FreePDK45/ncsu_basekit/models/hspice
* /tran_models/models_nom/NMOS_VTL.inc
* in which we updated the version statement from 4.0 to 4.8
* and enabled RF high speed support

* the W and L dimensions of INV_X1 and NAND2_X1
* come from NCSU FreePDK 45nm
* used by Christopher Torng 2019 for
* a 45nm ASIC design kit for mflowgen
* we use ngspice 25 degree C default circuit temperature,
* and vdd 1.10v per
* NangateOpenCellLibrary_typical
* Build Date: Thursday Feb 17 15:07 2011 library

********************************
* PARAMETERS for command lines,
* CSPARAMETERS for lines within control section or echos/prints
* refer to ngspice-27 manual for syntax

* Simulation time setup
* desired pulse train frequency in Hz
.param testfreq=1000e6
* printable as MHz
.csparam testfreqPrintable={testfreq/1e6}
```

```
* the period is then 1/frequency in seconds
.param testperiod='1/testfreq'
.csparam testperiodctl={testperiod}
*.csparam testperiodSmallerForPlotctl={testperiod*0.7}
.csparam testperiodSmallerForPlotctl={testperiod*1.1}
* how many cycles at specified frequency should be analyzed
.param testcycles='5'
* length of the simulation in seconds will be
* number of cycles times period
.param ttime='testperiod*testcycles'
.csparam ttimectl={ttime}
* how many simulation steps desired:
.param numberOfStepsInSim=100000
* resolution is length of simulation divided by steps
.param transstepsize='ttime/numberOfStepsInSim'

* power supply DC voltage, using
* Nangate typical corner Vdd=1.10 v, see
* NangateOpenCellLibrary_typical Build Date:
* Thursday Feb 17 15:07 2011
.param DCsupplyVoltage=1.10
* want to space multiple voltage plots clearly,
* so get ceiling of DCsupplyVoltage
.param PlotMultiVoltsSpacer=(DCsupplyVoltage+.9)
* need a csparam form to use in the control section
.csparam PlotMultiVoltsSpacerForCTL={PlotMultiVoltsSpacer}
* similarly, need DCsupplyVoltage param that works in control section
.csparam DCsupplyVoltageForCTL={DCsupplyVoltage}
* for y limit if plot n traces together, say n=5
.param numOfVoltTraces=5
.csparam DCsupplyVoltageForYLIM={PlotMultiVoltsSpacer*numOfVoltTraces}
```

```
* two-phase pulse train input stimulus parameters
* note rise/fall based on 20% of PMOS pulse width
.param RiseFallPercent=0.20
.param HalfPeriod='(testperiod/2)'
*.param PulseWpmos='(HalfPeriod)-(HalfPeriod*2*RiseFallPercent)'
.param PulseWpmos='(HalfPeriod)'
.param TdelayBeginNMOS='(HalfPeriod*RiseFallPercent*1.1)'
*.param PulseWnmos='(HalfPeriod)-(HalfPeriod*4*RiseFallPercent)'
.param PulseWnmos='(HalfPeriod)-(HalfPeriod*2.2*RiseFallPercent)'
.param TriseOrFall='(PulseWpmos*RiseFallPercent)'
.csparam TriseOrFallPrintable={TriseOrFall*1}

* load capacitance for NAND output
.param NANDLoadCapacitance=59fF
.csparam NANDLoadCapacitancePrintable={NANDLoadCapacitance*1e15}

************************************
* BSIM4 level=54 mosfet models from
* 2006 45nm_bulk.pm process file
* which we updated the version statement from 4.0 to 4.8

.include modelcard.nmos
.include modelcard.pmos

* use NMOS_VTL for the model name nmos in instantiation
* use PMOS_VTL for the model name pmos in instantiation
************************************
* MANDATORY SEPARATE POWER SUPPLIES for all circuits

* inverter 1 power supply:
vdd1 dd1 0 {DCsupplyVoltage}
vss1 ss1 0 dc 0
```

```
ve1  sub1  0 dc 0
vpe1 well1 0 {DCsupplyVoltage}

* inverter 2 power supply:
vdd2 dd2 0 {DCsupplyVoltage}
vss2 ss2 0 dc 0
ve2  sub2  0 dc 0
vpe2 well2 0 {DCsupplyVoltage}

* target C NAND gate power supply:
vdd3 vddnand 0 {DCsupplyVoltage}
* the NAND subckt simply uses global gnd node 0

*****************************************
* INVERTER SUBCIRCUIT definition

* Reminder: nd ng ns nb is the usual MOSFET lead order

* drive the PMOS gate separately from the NMOS
* one input each of PUN and PDN transistors of the inverter

.subckt inverter dd ss sub well inp inn out

mn1  out inn  ss  sub  NMOS_VTL W=0.415000U L=0.050000U

mp1  out inp  dd  well  PMOS_VTL W=0.630000U L=0.050000U

* used W and L dimensions of INV_X1 from NCSU FreePDK 45nm
* see Christopher Torng 2019 45nm ASIC design kit for mflowgen

.ends inverter
*****************************************
```

```
*******************************************
* INSTANTIATE 2 INVERTERS

* make 2 inverter instances using the subckt inverter above

* instantiate reference inverter (no load) inverter 1:
* dd ss sub well in out
X1 dd1  ss1  sub1  well1 inPMOS inNMOS out1 inverter

* instantiate test inverter 2 (load with target capacitance):
* dd ss sub well in out
X2 dd2  ss2  sub2  well2 inPMOS inNMOS out2 inverter

*******************************************

* CREATE the NAND gate whose input gate A will be
* target capacitance loading test inverter X2 of
* CBCM inverter pair

*** SUBCIRCUIT DEFINITION
* global gnd 0 not included in parameter list
.SUBCKT NAND in1 in2 out VDD
*
* PMOS parallel PUN pair
*  nd  ng  ns  nb  model
M3 out in1 vdd vdd PMOS_VTL W=0.630000U L=0.050000U
M4 out in2 vdd vdd PMOS_VTL W=0.630000U L=0.050000U

* NMOS series PDN pair
*   nd   ng  ns    nb model
M1 net.1 in2 0     0  NMOS_VTL W=0.415000U L=0.050000U
```

```
M2 out   in1 net.1 0  NMOS_VTL W=0.415000U L=0.050000U
*
.ENDS NAND
* above we use the W and L dimensions of NAND2_X1
* from NCSU FreePDK 45nm
* from Christopher Torng 2019 45nm ASIC design kit for mflowgen
*

* INSTANTIATE a NAND gate X3 using the SUBCKT:
*  in1 in2  out  VDD
X3 in1 in2  out  vddnand NAND

*****************************************

*****************************************
* MEASUREMENT CONNECTIONS

* measure effective capacitance of the in1 input to NAND gate
* when driven by the test vehicle inverter X2 out2


* provide low resistance path from test inverter X2 out2
* to NAND input 1 in1:
RTOTARGETC out2 in1 0.01

* tie unused NAND input in2 of NAND to vddnand,
* then a LOW to HIGH transition on inverter input
* will present LOW on in1 of NAND, causing NAND out
* to transition from LOW to HIGH, charging its load C
RX32 in2 vddnand 10k

* LOAD the NAND X3 output:
```

```
CLOAD out 0 {NANDLoadCapacitance}

*****************************************
* CONTROL SECTION

.control

* syntax note: initial plus sign is extension of previous line
* to keep columns less than equal 66 for printing

* run transient analysis defined outside the control section
 run
* make the plot white background instead of default black
 set color0 = white ; plot window -background color
 set color1 = black ; plot window -grid and text color
* thinner grid and plot lines?
 set xbrushwidth=0.5

* show "the old in out" (Clockwork Orange, Malcolm McDowell line)
plot inPMOS inNMOS+$&PlotMultiVoltsSpacerForCTL
+ out1+($&PlotMultiVoltsSpacerForCTL*2)
+ out2+($&PlotMultiVoltsSpacerForCTL*3)
+ out+($&PlotMultiVoltsSpacerForCTL*4)
+ yl 0 $&DCsupplyVoltageForYLIM xl 0 $&testperiodSmallerForPlotctl
+ title "CBCM measure NAND input C"

* plot NAND output and in1 NAND input separately
* to examine, e.g., prop delay
plot in1 out xl 0 $&testperiodctl

* measure avg vdd current in each inverter
* from t=0 to end of simulation
```

```
 meas tran inv1rmsCur AVG vdd1#branch from=0 to=$&ttimectl
 meas tran inv2rmsCur AVG vdd2#branch from=0 to=$&ttimectl

* use difference of two average inverter currents
* in vdd leads to measure capacitance load on one;
* formally,
* C=(input_period*abs(avg_inv1_current-avg_inv2_current)/Vdd
 let capmeasured= $&testperiodctl * abs(inv2rmsCur-inv1rmsCur)/
+               $&DCsupplyVoltageForCTL
 print capmeasured

 echo "(equivalent NAND input capacitance being charged"
 echo " by 2nd inverter, Farads)"
 echo "(Nangate typical build library suggests"
 echo "NAND2_X1 A1 input is 1.599 fF)"

 echo "Load capacitance on NAND output:  "
 echo $&NANDLoadCapacitancePrintable " fF"

 echo "Test frequency was " $&testfreqPrintable " MHz"
 echo "Input rise/fall time was " $&TriseOrFallPrintable " s"
 echo "Vdd supply voltage was "
 echo $&DCsupplyVoltageForCTL " volts DC"

.endc
*********************************************

*********************************************
* INVERTER PULSE INPUTS

* pulse both inverters' PMOS gates with VDD voltage
```

```
VIN1 inPMOS 0 PULSE(0 {DCsupplyVoltage}
+ 0 {TriseOrFall} {TriseOrFall} {PulseWpmos} {testperiod})

* pulse both inverters' NMOS input with VDD, but--
* want NMOS off while PMOS turns on or off, so gets
* delayed and smaller pulse width PulseWnmos

VIN2 inNMOS 0 PULSE(0 {DCsupplyVoltage} {TdelayBeginNMOS}
+ {TriseOrFall} {TriseOrFall} {PulseWnmos}
+ {testperiod})

*****************************************
* transient analysis
.tran 'transstepsize' 'ttime'

*****************************************
.END
```

Appendix F: Manually measure input capacitance, circuit file

The following is the Ngspice-27 circuit file named 20211211simpleCapMeasNANDin.cir. It measures the effective capacitance of input A1 (labeled in1 in schematic 1) of the NAND2_X1 circuit using the charge delivered to the gate (see Eq. 2 in Section V)

```
SIMPLE CAPACITANCE MEAS OF NAND INPUT CAP

* a less sophisticated simple input capacitance
* driving gate with voltage pulse
* and calculating charge delivered

* meas capacitance of NAND2_X1 gate input 1
```

```
* using Ngspice-27 Creation Date: Tue Dec 26 17:10:20 UTC 2017
* using BSIM4 level=54 mosfet models from  process file
* /FreePDK45/ncsu_basekit/models/hspice
* /tran_models/models_nom/NMOS_VTL.inc
* in which we updated the version statement from 4.0 to 4.8
* and enabled RF high speed support

* the W and L dimensions of INV_X1 and NAND2_X1
* come from NCSU FreePDK 45nm
* used by Christopher Torng 2019 for
* a 45nm ASIC design kit for mflowgen
* we use ngspice 25 degree C default circuit temperature,
* and vdd 1.10v per
* NangateOpenCellLibrary_typical
* Build Date: Thursday Feb 17 15:07 2011 library

********************************
* PARAMETERS for command lines,
* CSPARAMETERS for lines within control section or echos/prints
* refer to ngspice-27 manual for syntax

* Simulation time setup
* desired pulse train frequency in Hz
.param testfreq=1000e6
* printable as MHz
.csparam testfreqPrintable={testfreq/1e6}
* the period is then 1/frequency in seconds
.param testperiod='1/testfreq'
.csparam testperiodctl={testperiod}
.csparam testperiodctl2={testperiod*2}
*.csparam testperiodSmallerForPlotctl={testperiod*0.7}
```

```
.csparam testperiodSmallerForPlotctl={testperiod*1.1}
* how many cycles at specified frequency should be analyzed
.param testcycles='5'
* length of the simulation in seconds will be
* number of cycles times period
.param ttime='testperiod*testcycles'
.csparam ttimectl={ttime}
* how many simulation steps desired:
.param numberOfStepsInSim=100000
* resolution is length of simulation divided by steps
.param transstepsize='ttime/numberOfStepsInSim'

* power supply DC voltage, using
* Nangate typical corner Vdd=1.10 v, see
* NangateOpenCellLibrary_typical Build Date:
* Thursday Feb 17 15:07 2011
.param DCsupplyVoltage=1.10
* want to space multiple voltage plots clearly,
* so get ceiling of DCsupplyVoltage
.param PlotMultiVoltsSpacer=(DCsupplyVoltage+.9)
* need a csparam form to use in the control section
.csparam PlotMultiVoltsSpacerForCTL={PlotMultiVoltsSpacer}
* similarly, need DCsupplyVoltage param that works in control section
.csparam DCsupplyVoltageForCTL={DCsupplyVoltage}
* for y limit if plot n traces together, say n=5
.param numOfVoltTraces=5
.csparam DCsupplyVoltageForYLIM={PlotMultiVoltsSpacer*numOfVoltTraces}

* pulse train input stimulus parameters

* note rise/fall based on 20% pulse width
.param RiseFallPercent=0.20
```

```
.param HalfPeriod='(testperiod/2)'
.param PulseWidth='(HalfPeriod)'

.param TriseOrFall='(PulseWidth*RiseFallPercent)'
.csparam TriseOrFallPrintable={TriseOrFall*1}

* load capacitance for NAND output
.param NANDLoadCapacitance=59fF
.csparam NANDLoadCapacitancePrintable={NANDLoadCapacitance*1e15}

************************************
* BSIM4 level=54 mosfet models from
* 2006 45nm_bulk.pm process file
* which we updated the version statement from 4.0 to 4.8

.include modelcard.nmos
.include modelcard.pmos

* use NMOS_VTL for the model name nmos in instantiation
* use PMOS_VTL for the model name pmos in instantiation
************************************
* POWER SUPPLY

* NAND gate power supply:
vdd3 vddnand 0 {DCsupplyVoltage}
* the NAND subckt simply uses global gnd node 0

*******************************************

* CREATE the NAND gate whose input gate A will be
* target capacitance
```

```
*** SUBCIRCUIT DEFINITION
* global gnd 0 not included in parameter list
.SUBCKT NAND in1 in2 out VDD
*
* PMOS parallel PUN pair
*  nd  ng  ns  nb  model
M3 out in1 vdd vdd PMOS_VTL W=0.630000U L=0.050000U
M4 out in2 vdd vdd PMOS_VTL W=0.630000U L=0.050000U

* NMOS series PDN pair
*   nd   ng  ns    nb model
M1 net.1 in2 0     0  NMOS_VTL W=0.415000U L=0.050000U
M2 out   in1 net.1 0  NMOS_VTL W=0.415000U L=0.050000U
*
.ENDS NAND
* above we use the W and L dimensions of NAND2_X1
* from NCSU FreePDK 45nm
* from Christopher Torng 2019 45nm ASIC design kit for mflowgen
*

* INSTANTIATE a NAND gate X3 using the SUBCKT:
*  in1 in2  out  VDD
X3 in1 in2  out  vddnand NAND

*****************************************

*****************************************
* MEASUREMENT CONNECTIONS

* measure effective capacitance of the in1 input to NAND gate
* when driven voltage pulse train
```

```
* provide low resistance path
* to NAND input 1 in1 from pulse
* source, use approx. CMOS output R
* as voltage pulse source resistance
RTOTARGETC PulseInput in1 1000

* tie unused NAND input in2 of NAND to vddnand
RX32 in2 vddnand 10k

* LOAD the NAND X3 output:
CLOAD out 0 {NANDLoadCapacitance}

******************************************
* CONTROL SECTION

.control

* syntax note: initial plus sign is extension of previous line
* to keep columns less than equal 66 for printing

* run transient analysis defined outside the control section
 run
* make the plot white background instead of default black
 set color0 = white ; plot window -background color
 set color1 = black ; plot window -grid and text color
* thinner grid and plot lines?
 set xbrushwidth=0.5

* plot 2 cycles NAND output and in1 NAND input
plot in1 out xl 0 $&testperiodctl2
+ title "NAND input (in1, in2 at VDD) and output, 2 cycles"
```

```
* measure input capacitance of the NAND:
 let rcurrent = ( (pulseinput - in1) / @rtotargetc[resistance] )
 echo "charge transferred to NAND input: (Coulombs)"
 meas tran incurlh INTEG rcurrent from=0 to=$&TriseOrFallPrintable
 echo "voltage change on NAND input: (Volts)"
 meas tran maxin1 MAX in1 from=0 to=$&TriseOrFallPrintable
 echo ""
 echo "capacitance of NAND input measured, fF:"
 let NANDincap = (incurlh / maxin1)*1e15
 print NANDincap
 echo ""

 settype current rcurrent
 plot rcurrent xl 0 $&TriseOrFallPrintable
+ title "NAND in1 current L to H"
 plot in1 xl 0 $&TriseOrFallPrintable
+ title "NAND in1 gate voltage L to H"

****
 echo "(Nangate typical build library suggests"
 echo "NAND2_X1 A1 input is 1.599 fF)"

 echo ""
 echo "Load capacitance on NAND output:  "
 echo $&NANDLoadCapacitancePrintable " fF"

 echo ""
 echo "Test frequency was " $&testfreqPrintable " MHz"
 echo "Input rise/fall time was " $&TriseOrFallPrintable " s"
 echo "pulse source output impedance: (ohms)"
 print @RTOTARGETC[resistance]
 echo "Vdd supply voltage was "
```

```
 echo $&DCsupplyVoltageForCTL " volts DC"

.endc
*******************************************

*******************************************
* PULSE DRIVE of NAND input 1

VIN1 PulseInput 0 PULSE(0 {DCsupplyVoltage}
+ 0 {TriseOrFall} {TriseOrFall} {PulseWidth} {testperiod})

*****************************************
* transient analysis
.tran 'transstepsize' 'ttime'

*****************************************
.END
```

Appendix G: BSIM4 NMOS model card

The following is the BSIM4 NMOS model card used in the Ngspice circuits:

```
* Customized PTM 45 NMOS nom
* gdb: changed BSIM4 model to ngspice-27 latest version 4.8
* original file:
* /baiken-files/FreePDK45/ncsu_basekit/models/
* hspice/tran_models/models_nom/NMOS_VTL.inc
* also turn on RF high speed support:
* rgatemod=1 is not IIR, so leave on while turn on trnqsmod
* change trnqsmod=0 to trnqsmod=1; leave rbodymod= 1 alone,
* since it is appropriate * for high-speed sim
* and not incompatible with transient NQS
```

```
.model NMOS_VTL nmos level = 54

* parameters related to the technology node
+tnom = 27 epsrox = 3.9
+eta0 = 0.006 nfactor = 2.1 wint = 5e-09
+cgso = 1.1e-10 cgdo = 1.1e-10 xl = -2e-08

* parameters customized by the user
+toxe = 1.14e-09 toxp = 1e-09 toxm = 1.14e-09 toxref = 1.14e-09
+dtox = 0.14e-09 lint = 3.75e-09
+vth0 = 0.322 k1 = 0.4 u0 = 0.045 vsat = 148000
+rdsw = 155 ndep = 3.4e+18 xj = 1.98e-08

+version = 4.8 binunit = 1 paramchk= 1 mobmod = 0
+capmod = 2 igcmod = 1 igbmod = 1 geomod = 1
+diomod = 1 rdsmod = 0 rbodymod= 1 rgatemod= 1
+permod = 1 acnqsmod= 0 trnqsmod= 1

+ll = 0 wl = 0 lln = 1 wln = 1
+lw = 0 ww = 0 lwn = 1 wwn = 1
+lwl = 0 wwl = 0 xpart = 0

+k2 = 0 k3 = 0
+k3b = 0 w0 = 2.5e-006 dvt0 = 1 dvt1 = 2
+dvt2 = 0 dvt0w = 0 dvt1w = 0 dvt2w = 0
+dsub = 0.1 minv = 0.05 voffl = 0 dvtp0 = 1e-010
+dvtp1 = 0.1 lpe0 = 0 lpeb = 0
+ngate = 3e+20 nsd = 2e+020 phin = 0
+cdsc = 0 cdscb = 0 cdscd = 0 cit = 0
+voff = -0.13 etab = 0
+vfb = -0.55 ua = 6e-010 ub = 1.2e-018
```

```
+uc = 0 a0 = 1 ags = 0
+a1 = 0 a2 = 1 b0 = 0 b1 = 0
+keta = 0.04 dwg = 0 dwb = 0 pclm = 0.02
+pdiblc1 = 0.001 pdiblc2 = 0.001 pdiblcb = -0.005 drout = 0.5
+pvag = 1e-020 delta = 0.01 pscbe1 = 8.14e+008 pscbe2 = 1e-007
+fprout = 0.2 pdits = 0.08 pditsd = 0.23 pditsl = 2300000
+rsh = 5 rsw = 80 rdw = 80
+rdswmin = 0 rdwmin = 0 rswmin = 0 prwg = 0
+prwb = 0 wr = 1 alpha0 = 0.074 alpha1 = 0.005
+beta0 = 30 agidl = 0.0002 bgidl = 2.1e+009 cgidl = 0.0002
+egidl = 0.8 aigbacc = 0.012 bigbacc = 0.0028 cigbacc = 0.002
+nigbacc = 1 aigbinv = 0.014 bigbinv = 0.004 cigbinv = 0.004
+eigbinv = 1.1 nigbinv = 3 aigc = 0.02 bigc = 0.0027
+cigc = 0.002 aigsd = 0.02 bigsd = 0.0027 cigsd = 0.002
+nigc = 1 poxedge = 1 pigcd = 1 ntox = 1
+xrcrg1 = 12 xrcrg2 = 5

+cgbo = 2.56e-011 cgdl = 2.653e-010
+cgsl = 2.653e-010 ckappas = 0.03 ckappad = 0.03 acde = 1
+moin = 15 noff = 0.9 voffcv = 0.02

+kt1 = -0.11 kt1l = 0 kt2 = 0.022 ute = -1.5
+ua1 = 4.31e-009 ub1 = 7.61e-018 uc1 = -5.6e-011 prt = 0
+at = 33000

+fnoimod = 1 tnoimod = 0

+jss = 0.0001 jsws = 1e-011 jswgs = 1e-010 njs = 1
+ijthsfwd= 0.01 ijthsrev= 0.001 bvs = 10 xjbvs = 1
+jsd = 0.0001 jswd = 1e-011 jswgd = 1e-010 njd = 1
+ijthdfwd= 0.01 ijthdrev= 0.001 bvd = 10 xjbvd = 1
+pbs = 1 cjs = 0.0005 mjs = 0.5 pbsws = 1
```

```
+cjsws = 5e-010 mjsws = 0.33 pbswgs = 1 cjswgs = 3e-010
+mjswgs = 0.33 pbd = 1 cjd = 0.0005 mjd = 0.5
+pbswd = 1 cjswd = 5e-010 mjswd = 0.33 pbswgd = 1
+cjswgd = 5e-010 mjswgd = 0.33 tpb = 0.005 tcj = 0.001
+tpbsw = 0.005 tcjsw = 0.001 tpbswg = 0.005 tcjswg = 0.001
+xtis = 3 xtid = 3

+dmcg = 0 dmci = 0 dmdg = 0 dmcgt = 0
+dwj = 0 xgw = 0 xgl = 0

+rshg = 0.4 gbmin = 1e-010 rbpb = 5 rbpd = 15
+rbps = 15 rbdb = 15 rbsb = 15 ngcon = 1
```

Appendix H: BSIM4 PMOS model card

The following is the BSIM4 PMOS model card used in the Ngspice circuits:

```
* Customized PTM 45 PMOS
* gdb: changed BSIM4 model to ngspice-27 latest version 4.8
* original file:
* /baiken-files/FreePDK45/ncsu_basekit/models/
* hspice/tran_models/models_nom/PMOS_VTL.inc
* also turn on RF high speed support:
* rgatemod=1 is not IIR, so leave on while turn on trnqsmod
* change trnqsmod=0 to trnqsmod=1; leave rbodymod= 1 alone,
* since it is appropriate * for high-speed sim
* and not incompatible with transient NQS

.model PMOS_VTL pmos level = 54

* parameter customized by user
+vth0 = -0.3021 toxref = 1.26e-009 vsat = 69000
```

```
+toxe = 1.26e-009 toxp = 1.0e-009 toxm = 1.26e-009
+dtox = 2.6e-010 epsrox = 3.9 wint = 5e-009 lint = 3.75e-009

+version = 4.8 binunit = 1 paramchk= 1 mobmod = 0
+capmod = 2 igcmod = 1 igbmod = 1 geomod = 1
+diomod = 1 rdsmod = 0 rbodymod= 1 rgatemod= 1
+permod = 1 acnqsmod= 0 trnqsmod= 1

+tnom = 27

+ll = 0 wl = 0 lln = 1 wln = 1
+lw = 0 ww = 0 lwn = 1 wwn = 1
+lwl = 0 wwl = 0 xpart = 0 toxref = 1.3e-009
+xl  = -20e-9

+k1 = 0.4 k2 = -0.01 k3 = 0
+k3b = 0 w0 = 2.5e-006 dvt0 = 1 dvt1 = 2
+dvt2 = -0.032 dvt0w = 0 dvt1w = 0 dvt2w = 0
+dsub = 0.1 minv = 0.05 voffl = 0 dvtp0 = 1e-011
+dvtp1 = 0.05 lpe0 = 0 lpeb = 0 xj = 1.98e-008
+ngate = 2e+020 ndep = 2.44e+018 nsd = 2e+020 phin = 0
+cdsc = 0 cdscb = 0 cdscd = 0 cit = 0
+voff = -0.126 nfactor = 2.22 eta0 = 0.0055 etab = 0
+vfb = 0.55 u0 = 0.02 ua = 2e-009 ub = 5e-019
+uc = 0 a0 = 1 ags = 1e-020
+a1 = 0 a2 = 1 b0 = 0 b1 = 0
+keta = -0.047 dwg = 0 dwb = 0 pclm = 0.12
+pdiblc1 = 0.001 pdiblc2 = 0.001 pdiblcb = 3.4e-008 drout = 0.56
+pvag = 1e-020 delta = 0.01 pscbe1 = 8.14e+008 pscbe2 = 9.58e-007
+fprout = 0.2 pdits = 0.08 pditsd = 0.23 pditsl = 2300000
+rsh = 5 rdsw = 155 rsw = 75 rdw = 75
+rdswmin = 0 rdwmin = 0 rswmin = 0 prwg = 0
```

```
+prwb = 0 wr = 1 alpha0 = 0.074 alpha1 = 0.005
+beta0 = 30 agidl = 0.0002 bgidl = 2.1e+009 cgidl = 0.0002
+egidl = 0.8 aigbacc = 0.012 bigbacc = 0.0028 cigbacc = 0.002
+nigbacc = 1 aigbinv = 0.014 bigbinv = 0.004 cigbinv = 0.004
+eigbinv = 1.1 nigbinv = 3 aigc = 0.010687 bigc = 0.0012607
+cigc = 0.0008 aigsd = 0.010687 bigsd = 0.0012607 cigsd = 0.0008
+nigc = 1 poxedge = 1 pigcd = 1 ntox = 1
+xrcrg1 = 12 xrcrg2 = 5

+cgso = 1.1e-010 cgdo = 1.1e-010 cgbo = 2.56e-011 cgdl = 2.653e-010
+cgsl = 2.653e-010 ckappas = 0.03 ckappad = 0.03 acde = 1
+moin = 15 noff = 0.9 voffcv = 0.02

+kt1 = -0.11 kt1l = 0 kt2 = 0.022 ute = -1.5
+ua1 = 4.31e-009 ub1 = 7.61e-018 uc1 = -5.6e-011 prt = 0
+at = 33000

+fnoimod = 1 tnoimod = 0

+jss = 0.0001 jsws = 1e-011 jswgs = 1e-010 njs = 1
+ijthsfwd= 0.01 ijthsrev= 0.001 bvs = 10 xjbvs = 1
+jsd = 0.0001 jswd = 1e-011 jswgd = 1e-010 njd = 1
+ijthdfwd= 0.01 ijthdrev= 0.001 bvd = 10 xjbvd = 1
+pbs = 1 cjs = 0.0005 mjs = 0.5 pbsws = 1
+cjsws = 5e-010 mjsws = 0.33 pbswgs = 1 cjswgs = 3e-010
+mjswgs = 0.33 pbd = 1 cjd = 0.0005 mjd = 0.5
+pbswd = 1 cjswd = 5e-010 mjswd = 0.33 pbswgd = 1
+cjswgd = 5e-010 mjswgd = 0.33 tpb = 0.005 tcj = 0.001
+tpbsw = 0.005 tcjsw = 0.001 tpbswg = 0.005 tcjswg = 0.001
+xtis = 3 xtid = 3

+dmcg = 0 dmci = 0 dmdg = 0 dmcgt = 0
```

```
+dwj = 0 xgw = 0 xgl = 0

+rshg = 0.4 gbmin = 1e-010 rbpb = 5 rbpd = 15
+rbps = 15 rbdb = 15 rbsb = 15 ngcon = 1
```

Appendix I: Some recommended texts for MOSFET/CMOS theory

If you want to refresh yourself on the theory and operation of MOSFETs, Steck's textbook covering fundamental analog and digital electronics (covers general semiconductor junctions in Diode and FET/MOSFET characteristics in the analog section and CMOS specifically in the digital section) is freely available at[4]. B. Van Zeghbroeck at University of Colorado has a nice set of lectures on MOS field-effect transistors up as web pages.[52] The textbook by Ytterdal, Cheng and Fjeldly[45] has an excellent treatment of MOSFETs and modeling, in particular Chapter 1 *MOSFET Device Physics and Operation.* If you want to get a better understanding of semiconductor physics of the MOSFET at the graduate level (excellent presentation of energy bands), but with about as many diagrams as equations (happily), Professor Jesús del Alamo has his *Integrated Microelectronic Devices*[34] lecture notes freely available online at MIT.

For an integrated circuits view, the1999 Berkeley text, "Digital Integrated Circuits A Design Perspective," is very good. Portions of the first edition are downloadable at Berkeley.[6] The updated second edition can be purchased (Prentice Hall, 2006). For a readable, but highly technical at times, account of the development of the MOSFET by one of the participants from 1955 through 1986 (the date the paper was submitted), see[5].

For coverage of CMOS layout, see Kim's online lecture slides[7] at the School of Electrical Engineering and Computer Science, Washington State University or Andrew Mason's online lectures at Michigan State University College of Engineering.[21] For a detailed tutorial using Synopsys and Cadence automatic synthesis of integrated circuit (ASIC) tools, the online files of the Cornell University course ECE 5745 are recommended.[15]

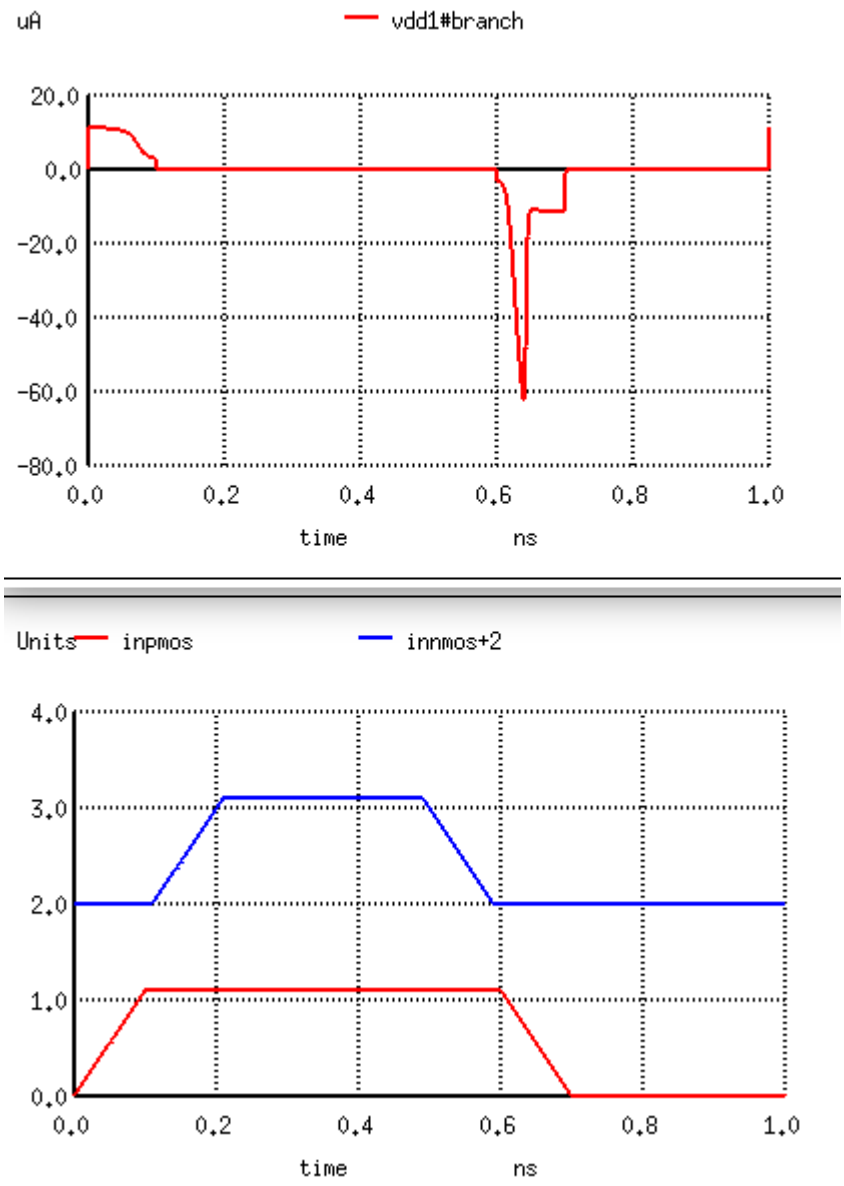

FIG. 12. The Ngspice V_{dd} charging current for the reference inverter (no load, i.e., simply intrinsic capacitance charged) in a CBCM pair is shown in upper plot (negative-going spike at $t = 0.6\,\text{ns}$, units μA. The lower part of the figure shows the input voltage waveforms to the PMOS and NMOS transistors of the inverters. The PMOS begins to turn on when inpmos (red) begins to drop at $t = 0.6\,\text{ns}$ (after the NMOS transistor has already turned off, blue trace inmos).

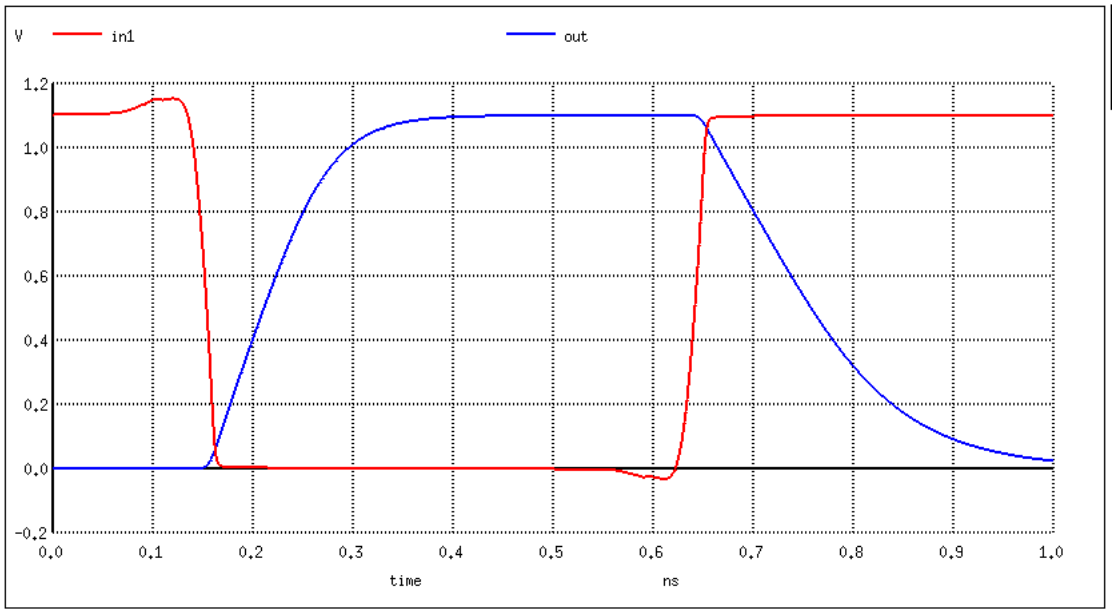

FIG. 13. Propagation of NAND2_X1 input in1 falling to output out rising, charging $C_L = 59\,\text{fF}$. Test frequency 1 GHz.

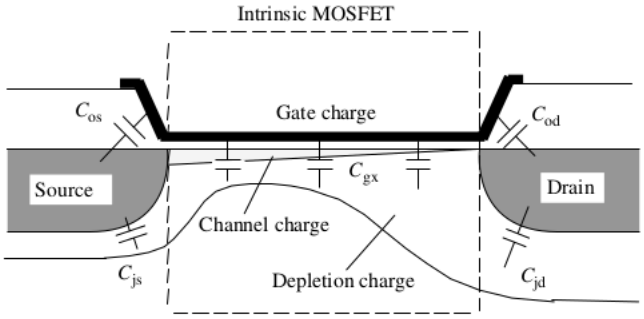

Figure 1.15 Intrinsic and parasitic capacitive elements of the MOSFET. Reproduced from Fjeldly T. A., Ytterdal T., and Shur M. (1998) *Introduction to Device Modeling and Circuit Simulation*, John Wiley & Sons, New York

FIG. 14. The MOSFET "parasitic capacitances," include overlap capacitances between the gate electrode and the highly doped source and drain regions (C_{os} and C_{od}), junction capacitances C_{js} and C_{jd} between substrate and source and drain regions. These are classified within the *extrinsic* part of the four-terminal MOSFET model.[46] The *intrinsic* portion of models (within the dotted line box in the figure) contains the channel mobile inversion charge and the depletion charge.

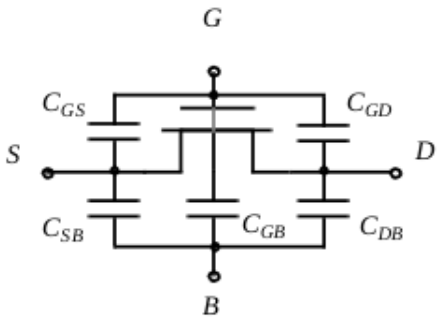

Figure 3.33 MOSFET capacitance model.

$$C_{GS} = C_{GCS} + C_{GSO};\ C_{GD} = C_{GCD} + C_{GDO};\ C_{GB} = C_{GCB}$$
$$C_{SB} = C_{Sdiff};\ C_{DB} = C_{Ddiff} \tag{3.47}$$

FIG. 15. A depiction of a MOSFET capacitance model with definition of composite terms. C_{SB} and C_{DB} denote the capacitive components contributed by the reverse-biased source-body and drain-body *pn*-junctions. Because these component are also referred to commonly as diffusion capacitance (though the doping may have been implanted by ion beam rather than diffusion), they may be labeled C_{Sdiff} and C_{Ddiff} at times ([6]).

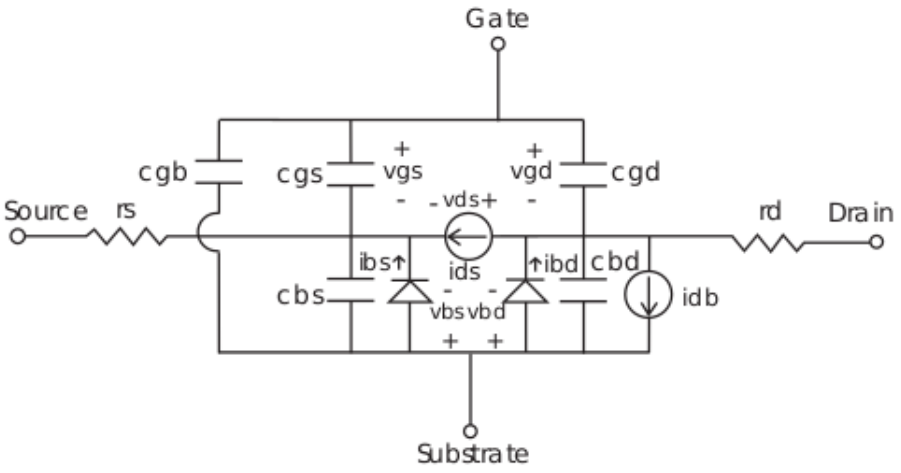

FIG. 16. Shown is an equivalent circuit for MOSFET transient analysis.[47]

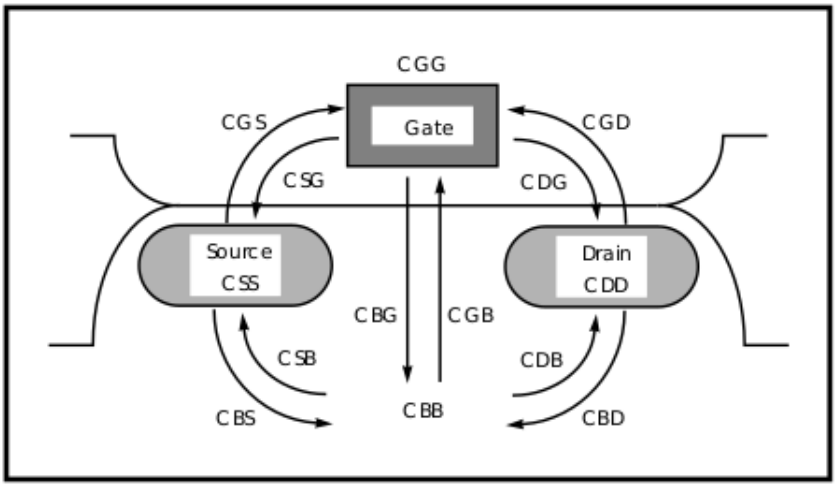

FIG. 17. MOSFET intrinsic four-terminal transcapacitances.[47]

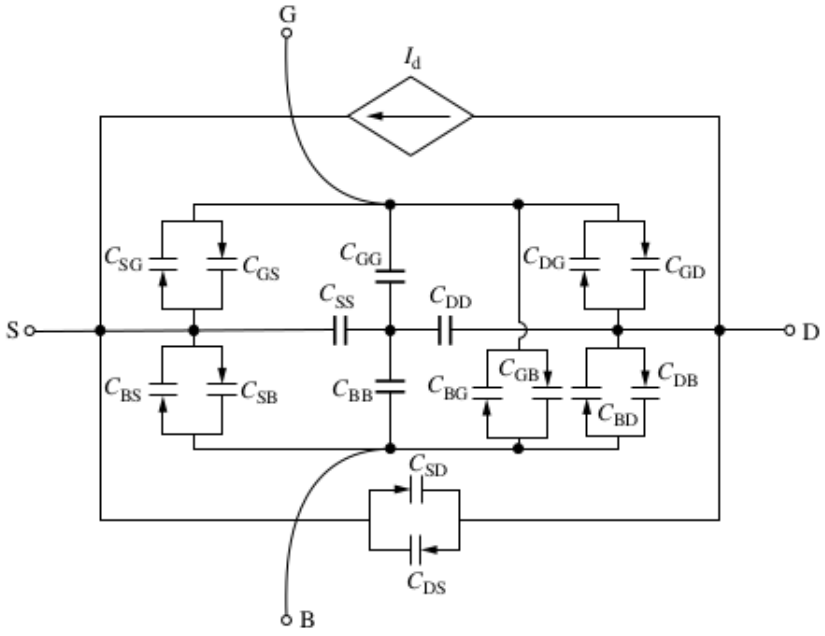

FIG. 18. MOSFET intrinsic large-signal equivalent circuit based on the Ward–Dutton charge-based nonreciprocal capacitances is shown.[45]

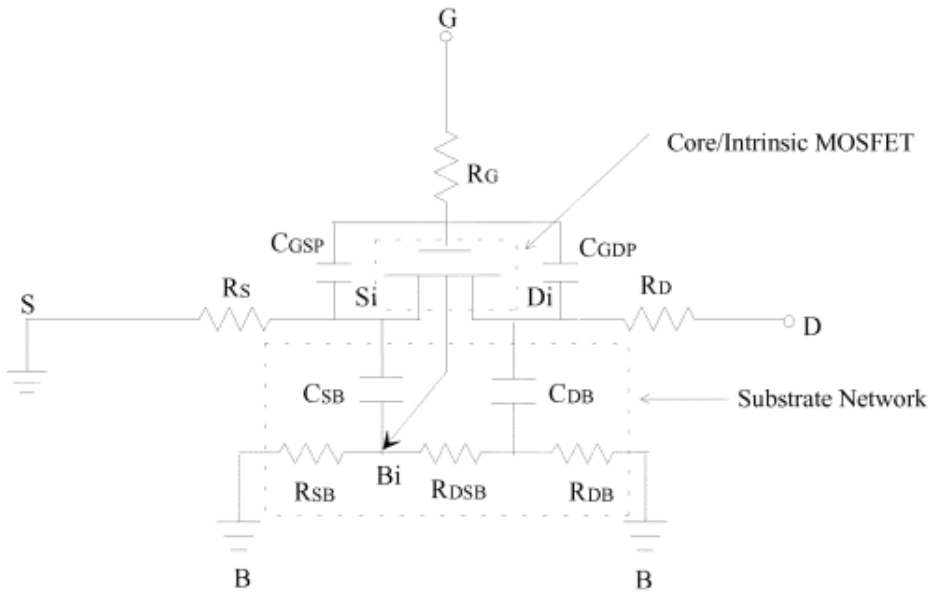

FIG. 19. MOSFET large-signal equivalent circuit.[46]

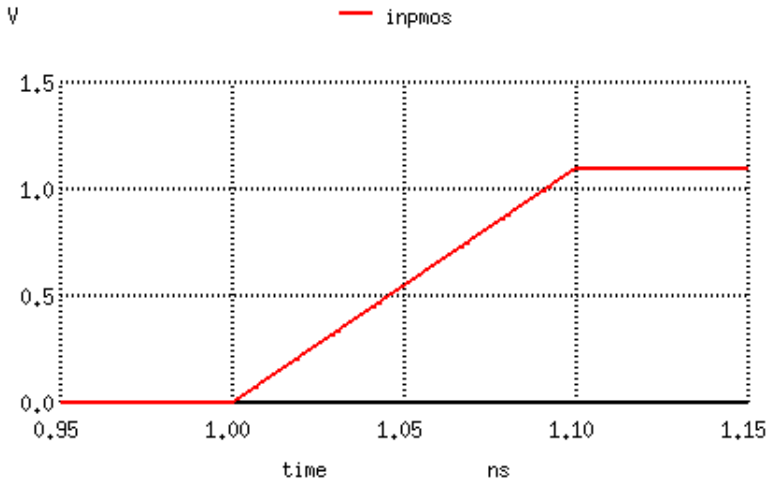

FIG. 20. PMOS gate voltage rises at $t = 1\,\text{ns}$ during the simulation, beginning turnoff of the transistor.

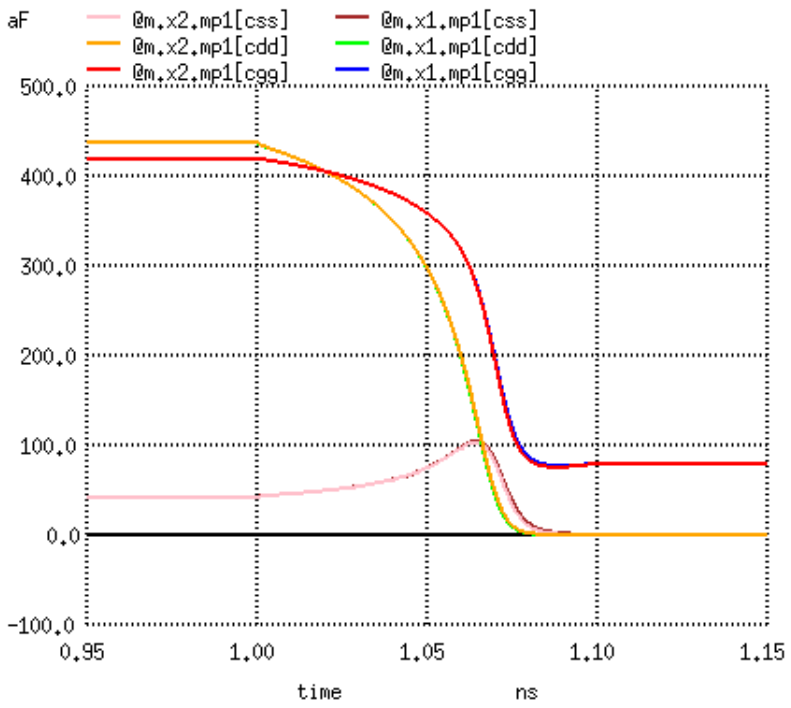

FIG. 21. PMOS gate voltage rises at $t = 1\,\text{ns}$ during the simulation, beginning turnoff of the transistor. The figure plots (for both inverter x1 and inverter x2, the latter loaded with 1.5 fF) the internal intrinsic core capacitance variables css (capacitance at internal source terminal), cdd (capacitance at internal drain terminal), and cgg (capacitance at internal gate terminal) (refer to Figure 17 and Figure 18 for depictions of these terminals). The values plotted are the partial derivatives $\partial Q_i/\partial V_j | \, i = j \in \{g, s, d\}$ along the diagonal of the Ward-Dutton matrix in Eq. A4 at the simulation time steps noted along the x-axis. The vertical scale is aF ($10^{-18}\,\text{F}$). The two inverters display almost identical capacitances here, so only one trace color may be visible if superimposed over the equivalent from the paired inverter.

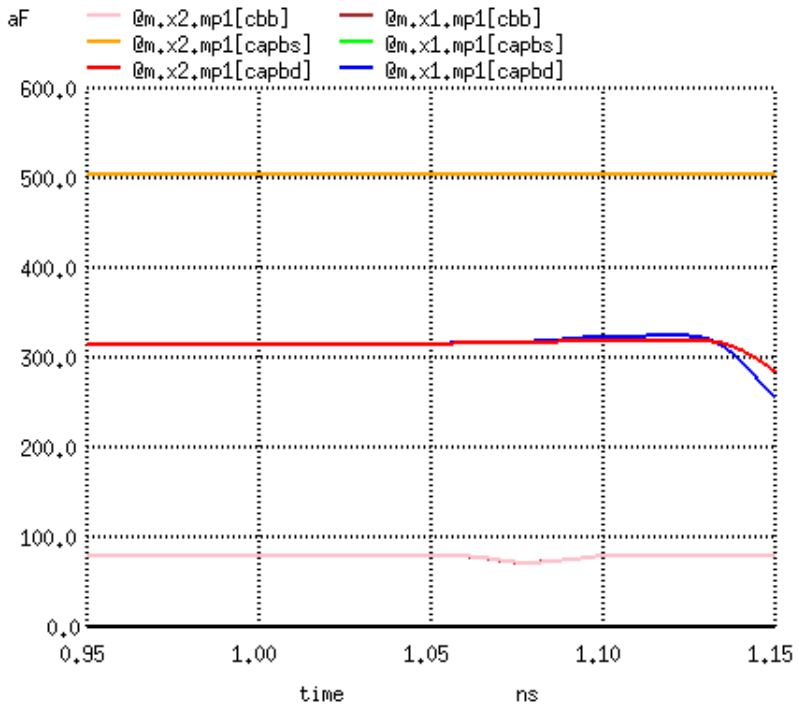

FIG. 22. PMOS gate voltage rises at $t = 1\,\text{ns}$ during the simulation, beginning turnoff of the transistor. The figure plots the internal intrinsic core capacitance variable cbb (capacitance at internal body terminal, superimposed faint and darker violet lowest trace in the plot below 100 aF, with slight change between 1.05 and 1.10 ns) and the extrinsic source-body and drain-body junction diode capacitances, capbs and capbd. Though largely constant, capbd begins to drop some time after PMOS turnoff completed at $t = 1.10\,\text{ns}$ and we see a divergence between loaded inverter x2 in red, delayed drop, and unloaded inverter x1 in blue, immediate drop.

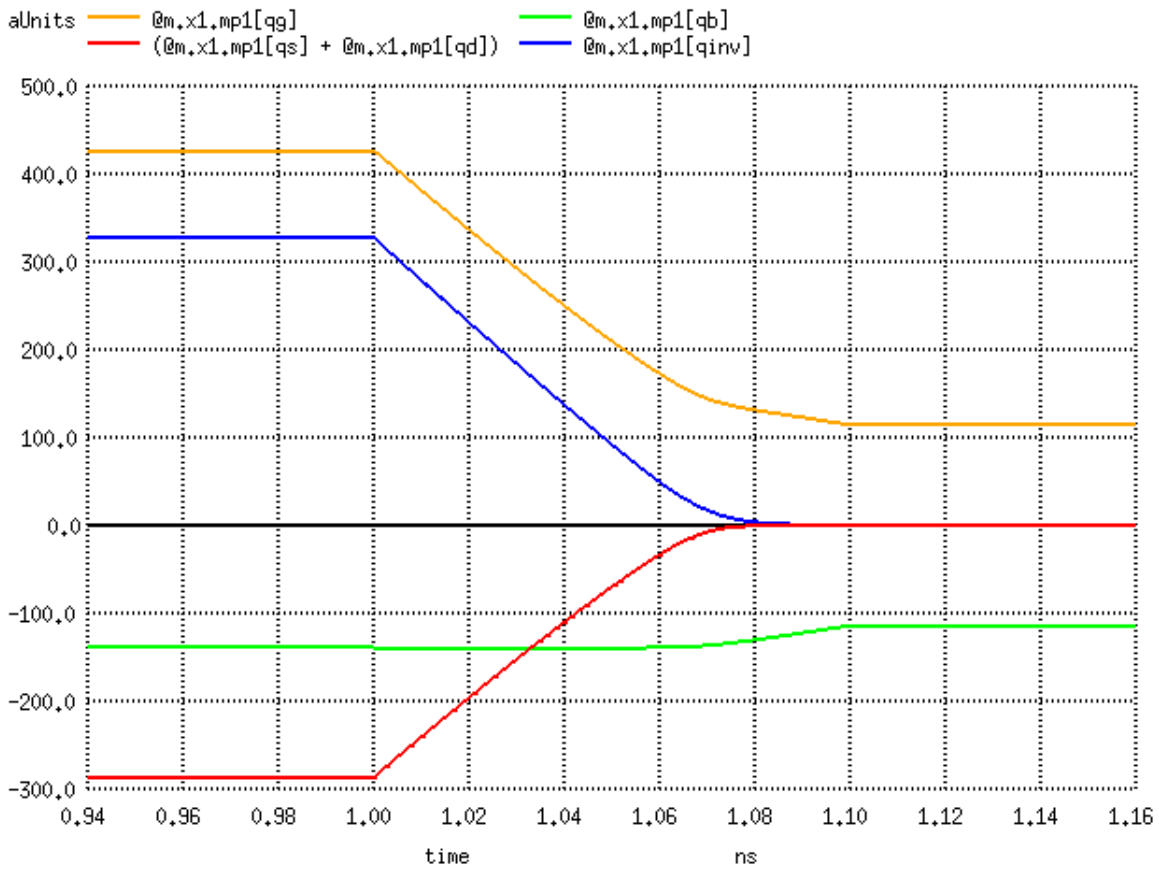

FIG. 23. BSIM4 internal charge variable behavior at PMOS turnoff $t = 1\,\text{ns}$ to $t = 1.1\,\text{ns}$ for the x1 inverter is shown. Notice that the channel charge qinv (blue trace) reflects the sum of drain qd and qs source channel charge (red trace) and that the gate charge qg (gold) can be seen to decrease as the sum of charge lost from qd, qs and qb.

[1] Dennis Sylvester and Chenming Hu, "Analytical Modeling and Characterization of Deep-Submicrometer Interconnect," PROCEEDINGS OF THE IEEE **89** (2001).

[2] Darsen Lu, "Compact Models for Future Generation CMOS," (2011), dissertation for PhD by Darsen Lu, Electrical Engineering and Computer Sciences, University of California at Berkeley, Technical Report No. UCB/EECS-2011-69. Chapter 3.

[3] "BSIM research group," Department of Electrical Engineering and Computer Sciences at the University of California, Berkeley, MOSFET SPICE models for circuit simulation and CMOS technology development.

[4] Daniel A. Steck, "Analog and Digital Electronics," (2021), Author is a physicist and professor at University of Oregon who wrote this textbook covering the fundamentals of analog and digital electronics for the senior physics undergraduate student. It is freely avalable in pdf at the link given.

[5] Chi-Tang Sah, "Evolution of the MOS Transistor–From Conception to VLSI," Proceedings of the IEEE **76** (1988), IEEE invited Sah, who was hired by Shockley when he left Bell Labs to form his own transistor manufacturing company in 1955 at Palo Alto, along with Noyce and Moore (who later started Intel), among others, to write about the historical development of the MOSFET up to 1986.

[6] "Digital integrated circuits a design perspective," (1999), draft of the first edition available in class notes for 2001 Berkeley course by Professor Borivoje Nikolic (one of the authors of the book). Intended for use in a senior/graduate level digital circuit design class, but also as a reference for professional engineers, the second edition is available for purchase, Prentice Hall, 2006.

[7] Dae Hyun Kim , "Physical Design of CMOS Integrated Circuits," 2017 EE434 lecture slides, School of Electrical Engineering and Computer Science, Washington State University, eecs.wsu.edu.

[8] Monzural Islam Dewan and Dae Hyun Kim, "NP-Separate: A New VLSI Design Methodology for Area, Power, and Performance Optimization," IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS **39** (2020).

[9] Paul Acorn, "Intel Process Roadmap Through 2025: Renamed Process Nodes, Angstrom Era

Begins," Tom's Hardware story discussing Intel CEO July 2021 announcement of roadmap to 2025 for the company.

[10] "New generation of PTM for bulk CMOS," 2006 Predictive Technology Model BSIM4 SPICE file for 45nm bulk CMOS.

[11] "45nm ASIC design kit for mflowgen, a modular ASIC/FPGA flow generator," (2019), We obtained portions of the FreePDK45 and the NanGate Open Cell Library files that were assembled into an ASIC design kit for mflowgen by Christopher Torng in 2019.

[12] "FreePDK45 Manual," Manual introduces the NCSU TechLib FreePDK45 technology library intended to work with the 45nm BSIM4 Predictive Technology Model from Arizona State University.

[13] "Predictive Technology Model (PTM) website (Arizona State University.)," PTM provides accurate, customizable, and predictive model files for future transistor and interconnect technologies. These predictive model files are compatible with standard circuit simulators, such as SPICE, and scalable with a wide range of process variations. With PTM, competitive circuit design and research can start even before the advanced semiconductor technology is fully developed..

[14] "Silicon Integration Initiative, Inc. ," "Silvaco's Open-Cell 15nm and 45nm FreePDK Libraries have been made available to Universities and Si2 Members at no charge.".

[15] Christopher Batten, "ECE 5745 Tutorial 5: Synopsys/Cadence ASIC Tools," (2021), Tutorial discussing how to use Synopsys and Cadence automatic synthesis of integrated circuit tools to map a register transfer language design at gate level down to standard cells and ultimately silicon. For ECE 5745 course at Cornell University.

[16] "FreePDK45 V1.4 Process Development Kit," Copy of FreePDK45 V1.4 Process Development Kit for the 45 nm technology made available by Github user baichen318.

[17] "The LayoutEditor," This commercial LayoutEditor by "juspertor GmbH" evolved from the 2004 Open Source project. They offer a free version for experimentation, which we have used briefly to get the lay of the land as it were in regard to the layout of integrated circuits. We found it quite difficult to use, possibly because of the seemingly haphazard limitations of the free version (functional and in regard to which necessary data files were omitted)..

[18] "Ngspice Users Manual Version 27," Use the manual for the ngspice-27 software release. It will be the ngspice.pdf.gz document in the compressed package for documents related to this

older release.

[19] Donald O. Pederson, "A Historical Review of Circuit Simulation," IEEE TRANSACTIONS ON CIRCUITS AND SYSTEM **CAS-31** (1984).

[20] "gschem Schematic Editor," gEDA project has produced and continues working on a full GPL'd suite and toolkit of Electronic Design Automation tools. The link here is for a Debian (Linux) package of the gschem portion of gEDA. It appears that gEDA development has slowed or ceased, but the schematic editor is useful for the Linux environment.

[21] "ECE 410 CMOS technology lecture slides ," 2007 ECE 410 lecture slides by Prof. Andrew Mason at Michigan State University College of Engineering..

[22] William H. Hayt and John A. Buck, *Engineering Electromagnetics*, eighth ed. (McGraw-Hill Companies, Inc., New York, 2012).

[23] Ivan E. Sutherland and Robert F. Sproull, "Logical Effort: Designing for Speed on the Back of an Envelope," (1991), Sutherland and Sproull, of Sun Microsystems, presented this paper at Advanced Research in VLSI 1991 University of California at Santa Cruz.

[24] "EDA NC State University," Electronic Design Automation classes and technology support at North Carolina State University.

[25] Steve C. Cripps, *RF Power Amplifiers for Wireless Communications*, 2nd ed. (Artech House, Inc., Norwood, MA, 2006) Section 4.4.2 Nonlinear capacitors–characterization and analysis in particular.

[26] J. C. Chen et al, "An on-chip, atto-farad interconnect charge-based capacitance measurement (CBCM) technique," PROCEEDINGS OF THE IEDM , 69–72 (1996).

[27] "Implications of Slow or Floating CMOS Inputs," Texas Instruments Applications Report SCBA004E – JULY 1994 – REVISED JULY 2021, Section 1: Characteristics of Slow or Floating CMOS Inputss.

[28] "Little Logic Data Book," (2001), Glossary of symbols, terms and definitions in Texas Instruments design reference for their small size logic devices..

[29] Lin Chao et al, "Intel's 45nm CMOS Technology," Intel Technology Journal **12** (2008), DOI:10.1535/itj.1202.

[30] "Understanding and Interpreting Standard-Logic Data Sheets," (2016), Texas Instruments Applications Report SZZA036C – December 2002– Revised June 2016, Section 4.9.2 Progagation Delay Time.

[31] Hank Zumbahlen, ed., in *Basic Linear Design* (Analog Devices, Inc., Norwood, MA, U.S.A, 2007) Chap. 1, pp. 76–78, Sections 1.64, 1.65.

[32] Eugene P. Wigner, "The Unreasonable Effectiveness of Mathematics in the Natural Sciences," Communications on Pure and Applied Mathematics **XIII** (1960), Lecture delivered at New York University May 11, 1959 by Wigner (who received a Nobel Prize in Physics 1963 "for his contributions to the theory of the atomic nucleus and the elementary particles, particularly through the discovery and application of fundamental symmetry principles.".

[33] "Introduction to Electronics," (1999), Written by retired Prof. Bob Zulinski, Dept. Electrical Engineering, Michigan Technological University, Houghton MI. Apparently no longer available at MTU, but found a copy at the website of Petter Braeken, an engineer and educator in Norway at http://brakken.no/el2/dokumenter/elint200.pdf.

[34] "Integrated Microelectronic Devices," (2007), Massachusetts Institute of Technology OpenCourseWare Spring 2007 graduate course by Prof. Jesús del Alamo covering the physics of microelectronic semiconductor devices for silicon integrated circuit applications. MIT Course Number 6.720J if url changes in future..

[35] "BSIM4 4.8.1 MOSFET Model," (2017), department of Electrical Engineering and Computer Sciences at the University of California, Berkeley, MOSFET SPICE models for circuit simulation and CMOS technology development. The url may change without warning, so you may have to search within the bsim.berkeley.edu or berkeley.edu domain to find a pertinent manual or source code.

[36] "CMOSedu.com," This is the personal website of R. Jacob Baker, associated with Department of Electrical and Computer Engineering, Boise State University. He is a CMOS circuit guru with several books published (see cmosedu.com for titles and information). Link is to discussion of variation of g_m and f_T with overdrive voltage..

[37] Jensen et al , U.S. Patent No. 6,781,434 B2 (24 Aug. 2004), describes a method to cancel the charge dump spikes on the source and drain of switching MOSFET transistors due to charging and discharging of overlap capacitances between gate and drain, gate and source within the devices.

[38] Fan et al , U.S. Patent No. 6,404,222 B1 (11 Jun. 2002), describes a method to limit measurement error due to the return of different size negative currents when using the CBCM charge-based capacitance measurement technique.

[39] Yongwang Ding and Ramesh Harjani, "A UNIVERSAL ANALYTIC CHARGE INJECTION MODEL," IEEE International Symposium on Circuits and Systems (2000), Authors presented this paper at ISCAS 2000, May 28-31, 2000, Geneva, Switzerland.

[40] S. Turgis et al, "INTERNAL POWER MODELLING AND MINIMIZATION IN CMOS INVERTERS," ACM (1997), Article identification: 1997 ACM/0-89791-849-5/97/0003, Authors Turgis, Daga, Portal and Auvergne at LIRMM UMR CNRS Montpelier, France.

[41] "Design Considerations for Logic Products Application Book," (1997), Collection of application reports and articles written or revised between 1992 and 1997 provided by Texas Instruments as technical reference for the design engineer. Identifier: SDYA002.

[42] J. R. Yeargan et al, "Printed Circuit Board Design for Advanced Schottky Family," Proceedings of the IEEE Region 5 Conference (1985).

[43] I. S . Gradshteyn and I. M. Ryzhik, *Table of Integrals, Series, and Products*, seventh ed. (Academic Press, Burlington, MA, USA, 2007) Translated from Russian by Scripta Technica, Inc. Edited by Alan Jeffrey, University of Newcastle, and Daniel Zwillinger, RPI..

[44] Chenming Hu, "BSIM Model for Circuit Design Using Advanced Technologies," IEEE International Symposium on Circuits and Systems (2001), 2001 Symposium on VLSl Circuits Digest of Technical Papers. DOI: 10.1109/VLSIC.2001.934176 · Source: IEEE Xplore.

[45] T. Ytterdal, Y. Cheng and T. A. Fjeldly , "Mosfet device physics and operation," (John Wiley and Sons, Ltds, 2003) pp. 1 – 45, Chapter 1 of Device Modelling for Analog and RF CMOS Circuit Design. isbn = 9780470863800. doi = 10.1002/0470863803.ch1. May request copy of Chapter 1 from authors at researchgate.net.

[46] Yuhua Cheng and M. Jamal Deen, "MOSFET Modeling for RF IC Design," IEEE TRANSACTIONS ON ELECTRON DEVICES **52** (2005), Copy available for download at Y. Cheng academia.edu account 21698939.

[47] "Star-Hspice Manual," (1998), Manual is intended for design engineers who use Star-Hspice to develop, test, analyze, and modify circuit designs. Avant! Corporation.

[48] Mounika Vanga, *Generating Linear Models for the MOS Transistors and Implementing Them In Obtaining Bode Surface Plots*, Master's thesis, Northern Illinois University, Department of Electrical Engineering (2016), Thesis examined generating equivalent models for MOSFETs where roots of transfer function are shifted by substituting reactive component equivalents to permit creating three-dimensional Bode surface plots, making detection of roots easy..

[49] Richard Feynman , "Feynman Caltech Lectures; Vol. II," (1963), Vol. II (electromagnetism and matter) of two-year introduction to physics lectures at Caltech 1961 to 1963 (by 1965 Nobel Prize in Physics laureate, Richard Feynman).

[50] W. C. Elmore, "The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifiers," Journal of Applied Physics **19** (1948), When transient response of a linear network to a unit step is monotonic rise to constant, a delay time and rise time can be defined in such a way as to be computed simply from the Laplace system function of the network.

[51] Liu and Hu, "BSIM4 and MOSFET modeling by Liu and Hu," Reportedly explanations of negative capacitance values in the BSIM4 internal parameters and how the values are evaluated are found in Chapter 5.2 of this book (for purchase).

[52] "Principles of Semiconductor Devices," (2011), B. Van Zeghbroeck with ECEE department at the University of Colorado has posted well-written lectures (including numerous figures and illustrations) covering semiconductors. See Chapter 7 MOS field effect transistors in particular.