IRRADIATION AND TEMPERATURE CHARACTERIZATION FOR A 32NM RF SILICON-ON-INSULATOR CMOS PROCESS

By

Timothy D. Haeffner

Thesis

Submitted to the Faculty of the

Graduate School of Vanderbilt University

In partial fulfilment of the requirements

for the degree of

MASTER OF SCIENCE

in

Electrical Engineering

May, 2015

Nashville, Tennessee

Approved:

Professor Lloyd W. Massengill

Professor Ronald D. Schrimpf

ACKNOWLEDGMENTS

I would like to thank The Institute of Space and Defense Electronics, my advisor Dr. Lloyd Massengill and Dr. Ron Schrimpf for allowing me the opportunity to complete my Master's Degree in Electrical Engineering. Specifically I would like to thank Dr. Daniel Loveless, Dr. Enxia Zhang, Dr. Andrew Stenberg, and Dr. Jeff Kauppila for their help during the design process, data collection, data post-processing and analysis they all provided. I am also thankful to Dr. Dan Fleetwood, Dr. Mike Alles and Dr. Ron Schrimpf for providing key technical insight to this work.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

CHAPTER I

INTRODUCTION

Significant improvements in high-frequency (> 10 GHz) performance (i.e., power gain, f_T and f_{MAX}) have been observed in commercial sub-100 nm silicon-on-insulator (SOI) complementary metal-oxide-semiconductor (CMOS) technologies over previous generation CMOS technologies. Unity current gain cutoff frequencies (f_T) exceeding 200 GHz have been reported [1], offering the possibility of highly integrated communications systems-on-chip in CMOS technology. Fig. 1 demonstrates the impact of technology scaling on f_T.

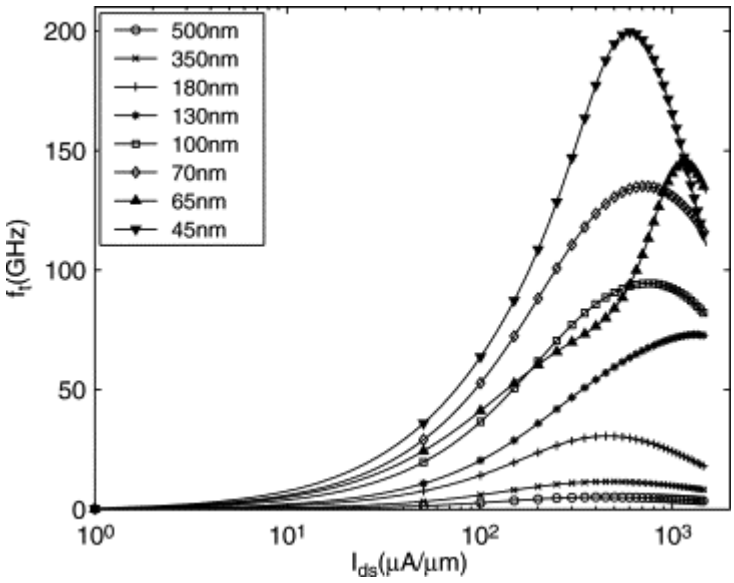


Fig. 1. Impact of technology scaling on f_T [2].

For space applications, it is important to consider total ionizing dose (TID) effects, which can cause degradation in DC and AC parameters for low frequency and radio frequency (RF) operating conditions. Relatively small TID parametric degradation, especially when compounded with the effects of process variation and temperature

extreme, can lead to analog RF circuit failures [3], [4]. The strict design guidelines for high-frequency RF applications require precise device characteristics and operation, hence the importance of understanding the effects of TID on DC and RF operation.

In this thesis, the impacts of total ionizing dose (TID), temperature and RF bias on the DC and RF performance of a commercial 32 nm RF SOI CMOS technology are presented. Temperature dependence is shown to be the overwhelmingly dominant single factor affecting the DC and RF performance, with the combined effects of elevated temperature and TID showing the most pronounced degradation. The most significant effect due to TID is an increase in off-state leakage current. Key DC and RF parameters of this 32 nm RF process degrade less than those of an otherwise similar 45 nm RF SOI CMOS process. The implications of the combined TID and temperature response are discussed for low-power RF design.

CHAPTER II

BACKGROUND OVERVIEW

The need to understand radiation effects on semiconductors and implement mitigation techniques is primary to the success of many electronic designs. For in-flight applications, the radiation environment includes total ionizing dose (TID) and single event effects (SEE) that differ in degree depending on altitude, inclination, and orbital paths in proximity to the South Atlantic anomaly. Space and military applications have their own set of radiation environments that must carefully be considered due to high levels of ionizing or total-dose radiation. Ground based electronic systems experience SEEs caused from cosmic rays that penetrate the atmosphere and particle-emitting radioisotopes in packaging materials. For more than 40 years, engineers have worked to understand various radiation effects mechanisms and develop ways to minimize, prevent, and eliminate the unwanted effects in electronic designs.

Radiation Effects

Three categories effectively summarize radiation effects for CMOS devices, total ionizing dose (TID), single-event effects (SEE) and dose-rate ionizing radiation. TID can be defined as the total amount of energy deposited by ionized particles that results in creation of electron-hole pairs. When ionizing radiation interacts with a semiconductor material electron-hole pairs are created in the semiconductor material or the oxide (which is the most sensitive). The electrons generated by the ionizing radiation can either recombine or be swept away, but the holes have much lower mobility than

electrons. The holes can become trapped in the gate oxide or the STI (shallow trench isolation) oxide causing parametric device changes [5]. Fig. 2 illustrates the process of how ionizing radiation creates electron-hole pairs for a semiconductor device.

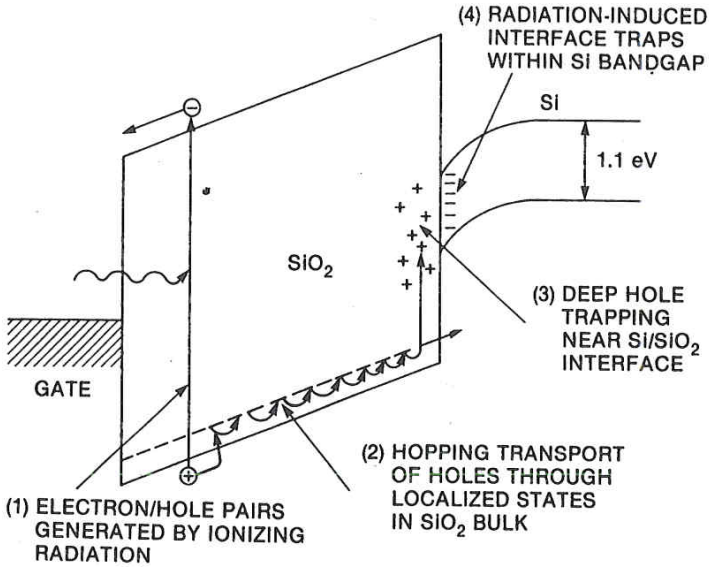


Fig. 2. Schematic showing ionizing radiation induced electron-hole pair generation for a semiconductor device [5].

TID, which is the focus of this work, is considered a long term or cumulative ionization effect that causes device parametric degradation which effects threshold voltage and leakage current. Fig. 3 demonstrates the TID response of 45 nm RF SOI CMOS *n*MOSFET *I-V* characteristics. The off-state leakage current increases significantly with dose.

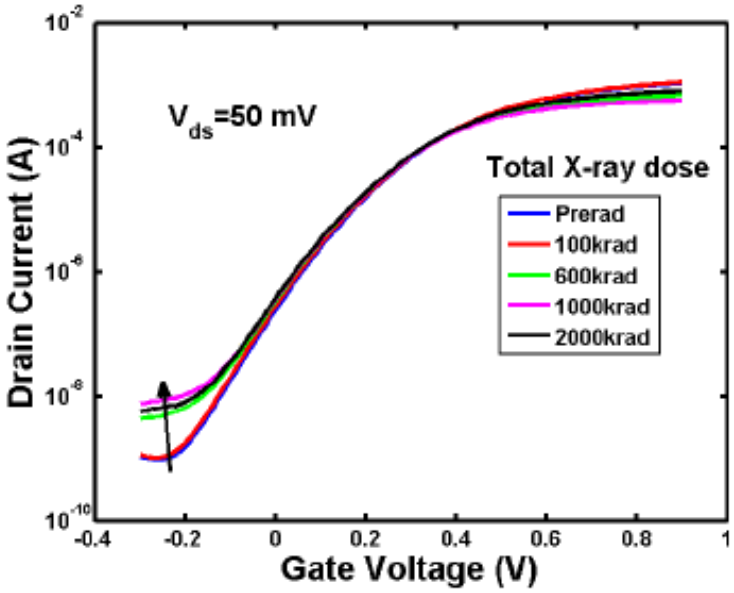


Fig. 3. I_{DS}-V_{GS} characteristics for a 45 nm RF SOI CMOS *n*MOSFET after TID irradiation showing a significant increase in off-state leakage [6].

SEE's and dose-rate ionizing radiation are considered transient effects. A high dose-rate event causes photocurrents in a semiconductor that can cause rail-span collapse or burnout of a metal line depending on the intensity of the event. SEE's are caused by individual particles. They are considered a short term effect and occur when a high-energy particle strikes a semiconductor device (i.e., drain) leaving behind a track of excess charge. Depending on the movement and location of the charge left behind, the result can cause temporary damage (i.e., SRAM bit-flips) or even permanent damage such as transistor gate rupture if current is high enough from the particle strike.

Silicon-on-Insulator Technology Overview

SOI technology (the focus of this work) has been used for decades in radiation hardened electronic designs because of its intrinsic benefits that mitigate some radiation

effects (i.e., latch-up) [7]. SOI technologies allow designers a CMOS type transistor that is dielectrically isolated which eliminates conduction paths to other devices or junctions. There are two types of SOI devices, fully depleted (FD) and partially depleted (PD) devices. They differ geometrically in the thickness of the active silicon film (body) between source and drain. FD devices have a very thin layer of silicon under the gate that is fully depleted of mobile carriers over the entire body. PD devices have a thicker layer of silicon under the gate that is partially depleted of mobile carriers. There is a neutral zone isolating the top device from the buried oxide which results in a "floating body" effect. The PD device's electrical behavior depends on previous bias conditions resulting from the floating body potential. The difference is significant enough between the two types of devices that their electrical characteristics require different sets up I-V equations. Fig. 1(a) shows a fully depleted transistor. This work focuses on partially depleted SOI devices. Fig. 1(b) shows an example of a partially depleted SOI transistor.

There are also unwanted parasitics inherent to SOI devices (when irradiated) that degrade device parameter performance requiring careful consideration [7]. For example, the parasitic back gate device shown in Fig. 1(b) is sensitive to TID effects (i.e., leakage), because of positive trapped charge in buried oxide due to the ionizing radiation. The parasitic bipolar transistor also shown in Fig.1(c) is caused by the floating body node and sensitive to SEE ion strikes. Charge that is collected during an ion strike can be amplified by the parasitic bipolar device, another unwanted characteristic associated with PD SOI devices.

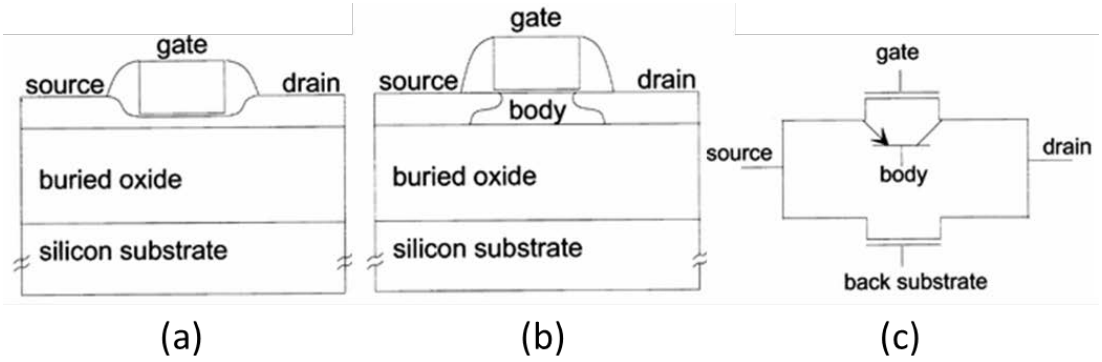


Fig. 4 (a) SOI fully depleted transistor; (b) SOI partially depleted transistor; and (c) equivalent electrical structure activated by irradiation [7].

CMOS Technology Enhancements for RF Designs

As CMOS technologies are scaled into the sub-100 nm range, the technology performance characteristics in the hundreds of GHz range are attractive for many RF SOC platforms (i.e., GPS, cell phones, satellite communication systems and weapon systems). Cutting edge sub-100 nm SOI CMOS technologies studied in this work require new and innovative process enhancements (i.e., strained silicon and high-k dielectrics) because just scaling devices can no longer meet the necessary technology performance requirements. Strained silicon such as silicon-germanium (SiGe) improves carrier mobility which improves devices performance. High-k metal gates increase drive current and g_m. Fig. 5 illustrates the CMOS device progression from the 130 nm technology node down to the 32 nm technology node [8].

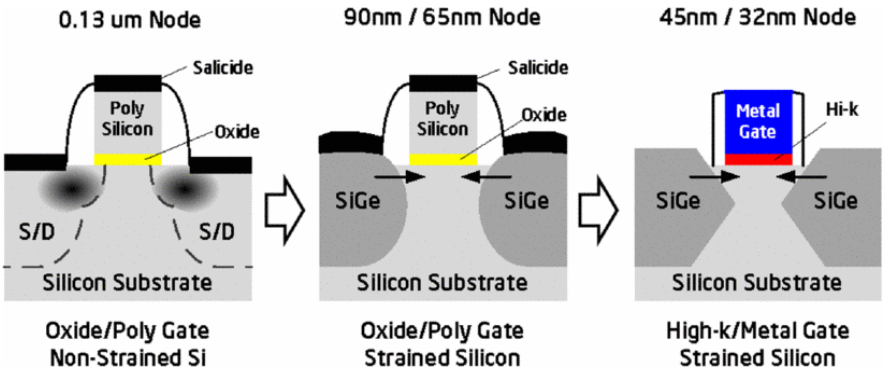


Fig. 5. Evolution of CMOS transistor architecture in the last decade, from a non-strained oxide/poly gate structure to a high-k/metal gate strained silicon transistor at the 32 nm node [8].

Along with speed and current drive enhancements, unwanted parasitic gate capacitance and resistance increase as technologies scale adversely affecting device performance. Therefore it important to characterize technology improvements carefully so RF designers understand the tradeoffs involved in order meet strict design requirements

[2]. In Fig. 6 the impact of technology scaling on CMOS parasitic capacitance and resistance is shown.

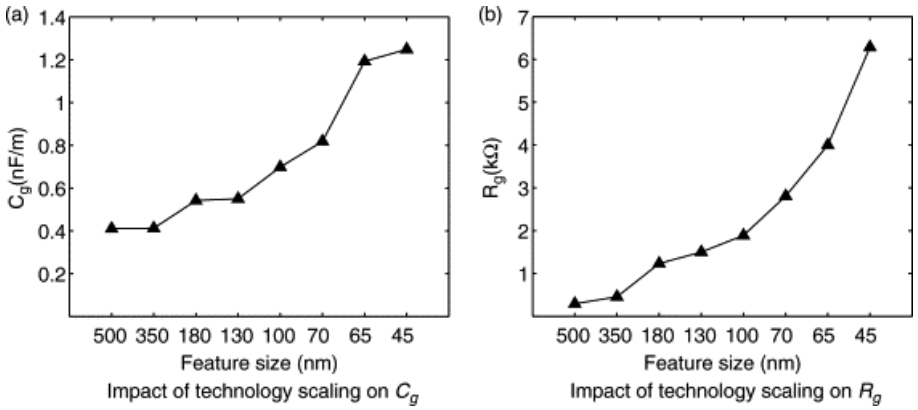


Fig. 6. Impact of technology scaling on CMOS gate capacitance and resistance [2].

The effect of capacitive parasitics can be realized in the cut-off frequency equation f_T for a MOSFET shown below. f_T is indirectly proportional to the parasitic gate capacitance (Cgs and Cgd) which undesirably affects the cut-off frequency and directly proportional to the transconductance (g_m).

$$f_T = \frac{g_m}{2\pi(Cgs + Cgd)}$$

Temperature/Mobility Fundamentals

Temperature is another key design reliability parameter in CMOS/SOI electronic designs. Designers must ensure that semiconductor devices operate correctly across a broad range of temperatures (i.e., -40ºC to 125ºC). Temperature can change the operation of semiconductor devices significantly. Fig. 7 illustrates SOI and bulk CMOS *p*MOSFET I-V characteristics versus temperature. Carrier mobility (μ), threshold voltage (V_{th}), and

leakage current (Ileak) are affected by increasing temperature. Carrier mobility in a semiconductor device increases at low temperature and decreases at high temperatures [9].

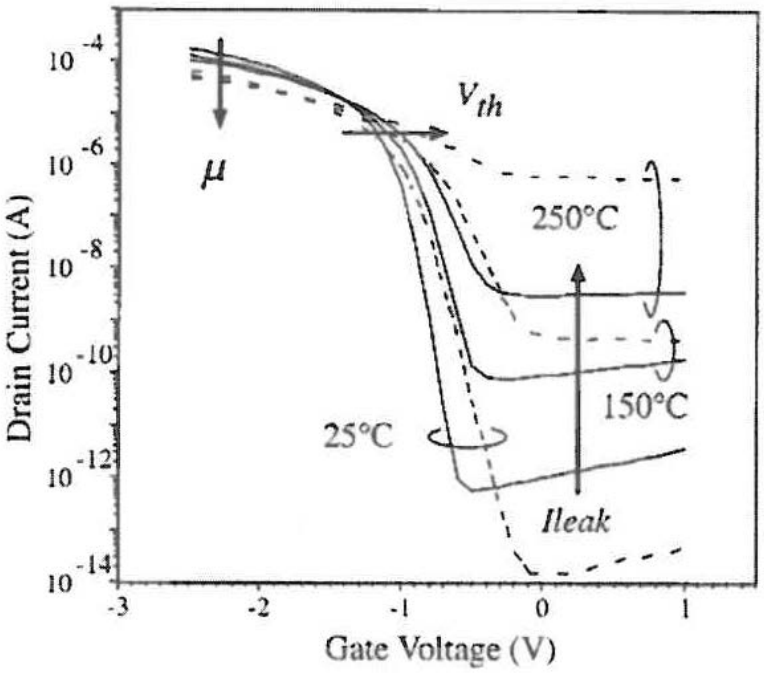


Fig. 7. SOI *p*MOSFET: I_{DS}-V_{GS} characteristics illustrating temperature dependence for SOI (solid line) and bulk CMOS (dashed line) [9].

In this work, competing temperature mechanisms are shown to be an important effect in low-power RF designs. Drain current (I_{DS}) increases with elevated temperature at low bias, whereas it decreases with elevated temperature at high biases [10,11].

Thesis Organization

This thesis presents experimental testing of multiple transistors in a commercial 32 nm RF SOI CMOS process to characterize how key DC and RF (f_T and S_{21}) parameters respond to TID, temperature and RF experiments. The experimental results are used to help understand the separate and combined effects of irradiation, temperature and RF experiments for key design parameters. The 32 nm experimental results are then compared with similar devices in another commercial 45 nm RF SOI CMOS process (previously published). Finally, the implications of the combined TID and temperature response are discussed for low-power RF design.

- Chapters 3 and 4 describe the test chip design effort, contents of the test chip and the setup for the temperature, TID and RF stress experiments setup.
- In Chapter 5, experimental temperature dependence results of the 32 nm transistors for key DC and RF design parameters are shown.
- Chapter 6 illustrates the TID pre-irradiation and post-irradiation experimental results for the 32 nm key DC and RF design parameters.
- Chapter 7 demonstrates the combined response of TID and temperature dependence experimental results for the 32 nm devices.
- Chapter 8 compares the 32 nm process experimental results and previously published 45 nm process experimental results for key DC and RF parameters.
- Chapter 9 illustrates the RF experimental results for the 32 nm process.
- Chapter 10 concludes this thesis.

CHAPTER III

32NM TEST CHIP DESIGN

This chapter provides information about the test chip design used this work. The focus of this work is the test chip designed in a commercial 32 nm RF SOI CMOS process. Fig. 8 is a snapshot of the 32 nm test chip layout design.

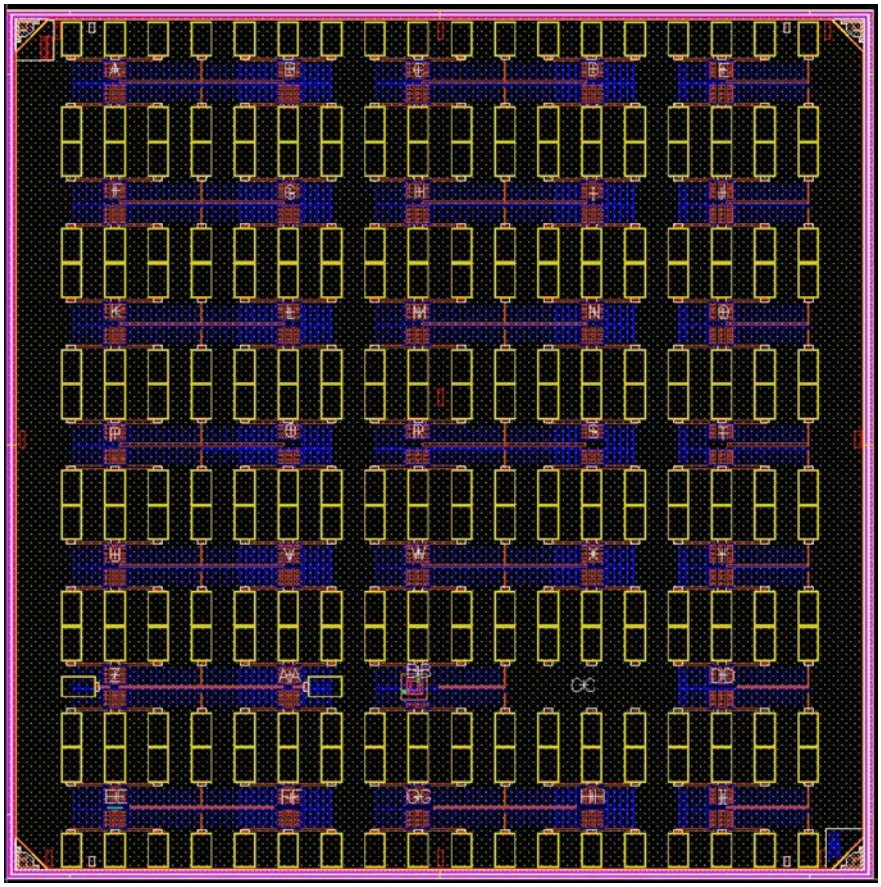


Fig. 8. 32 nm test chip layout.

The 32 nm test chips contain different types of *n*MOSFET and *p*MOSFET devices. The four devices shown in Table 1 are the focus of this work, varying in dimension and number of gate fingers. The data shown in this experiment are for floating body *n*MOSFET and *p*MOSFET devices. The full chart of devices on the test chip can

be found in Appendix A.

Table 1. 32nm transistors used in this study.

Device	Width/Length	Total Width	Fingers
A) NMOS	2 μm/56 nm	24 μm	12
B) NMOS	1 μm/56 nm	24 μm	24
C) PMOS	2 μm/56 nm	24 μm	12
D) PMOS	1 μm/56 nm	24 μm	24

The process details for the 32 nm and 45 nm RF SOI CMOS technologies compared in this work have been published previously; the main difference between the 32 nm and 45 nm nodes is the gate stack composition [12,13]. The 32 nm technology utilizes a high-κ dielectric with a metal gate, whereas the 45 nm process utilizes a SiO_2 gate dielectric and a poly-crystalline silicon gate [12]. Fig. 9 illustrates a transmission electron microscopy (TEM) cross-section of a 32 nm device for the technology presented in this work.

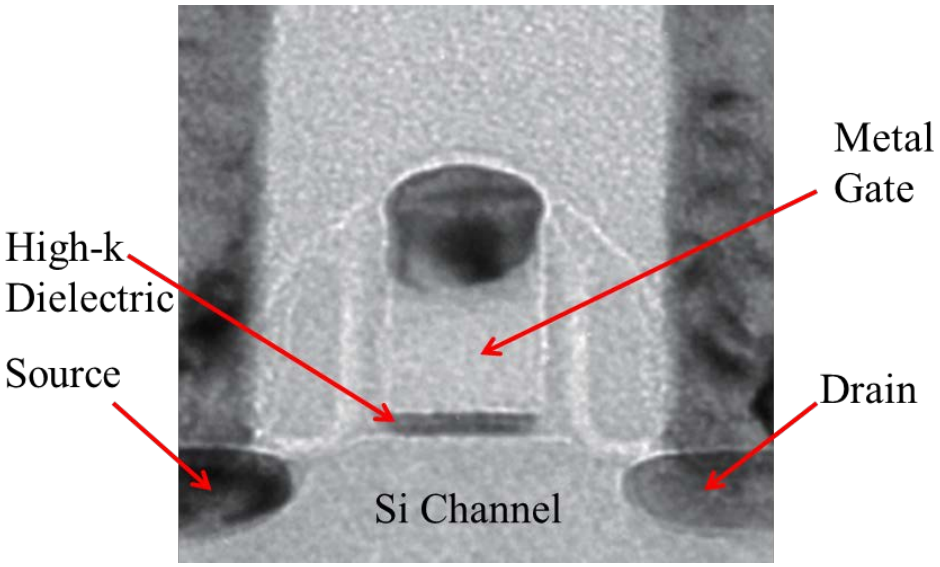


Fig. 9. 32 nm SOI device cross-section showing the metal gate and high-κ dielectric [12].

Ground-signal-ground probe pads with a 100 μm center-to-center pitch were used

to contact the devices through co-planar waveguides designed with a 50 Ω characteristic impedance (Z_O). Z_O was designed to be constant up to a frequency of 50 GHz to enable high-speed measurement of the transistor parameters. Fig. 10 illustrates the layout of a single transistor on the test chip and indicates the RF probe pads and coplanar waveguides. The gate (input) and drain (output) of each *n*MOS or *p*MOS transistor were connected to the signal pads and the source was connected to an RF probe pad shared by two devices. The test chip also included calibration test structures (Open, Short, Through and Load) for de-embedding the on-chip parasitic impedances for high-frequency S-parameter measurements [13,14].

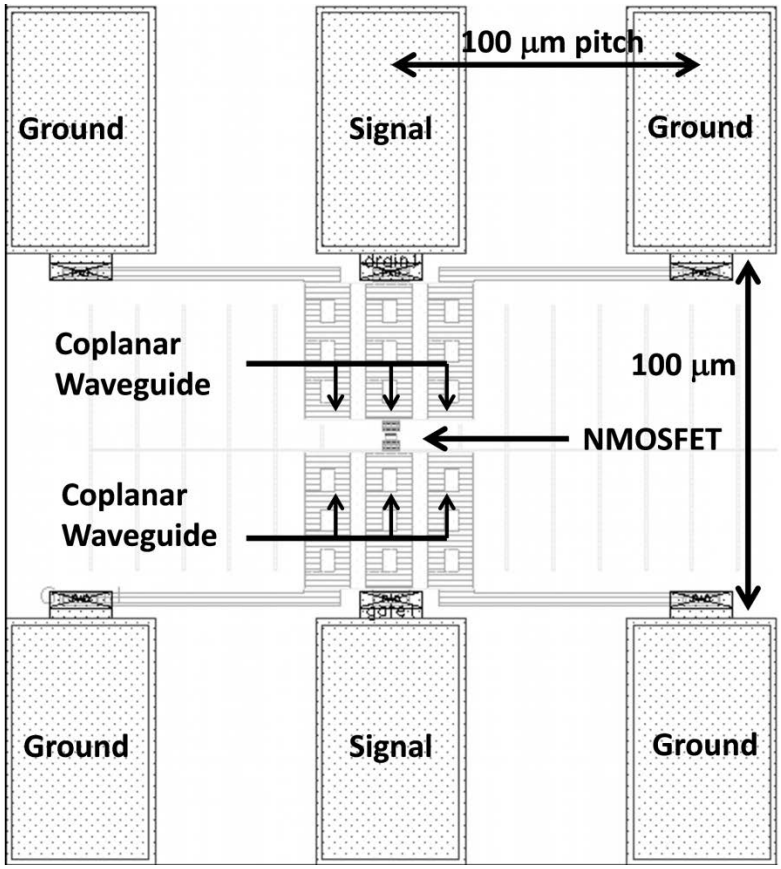


Fig. 10. Layout of a single 32 nm transistor connected to 100 μm pitch RF probe pads through 50 Ω coplanar waveguides.

CHAPTER IV

TEMPERATURE, TID, AND RF BIAS EXPERIMENTAL SETUP

This chapter provides the experimental details for data presented in this paper. For further experimental test setup details and data for the 32 nm test chip please see Appendix A.

Experimental Conditions

The setup for the experiment to measure and de-embed the high-frequency S-parameters and cable parasitics is shown in Fig. 11. The high-frequency RF measurements were made from 10 MHz to 50 GHz using an Agilent N5245A network analyzer. Devices were irradiated with an ARACOR 10-keV X-ray source at a dose rate of 31.5 krad(SiO_2)/min. The IC was mounted in a high-speed package manufactured from a brass block (shown in Fig. 12) and each transistor was bonded to 50 Ω microstrips [13].

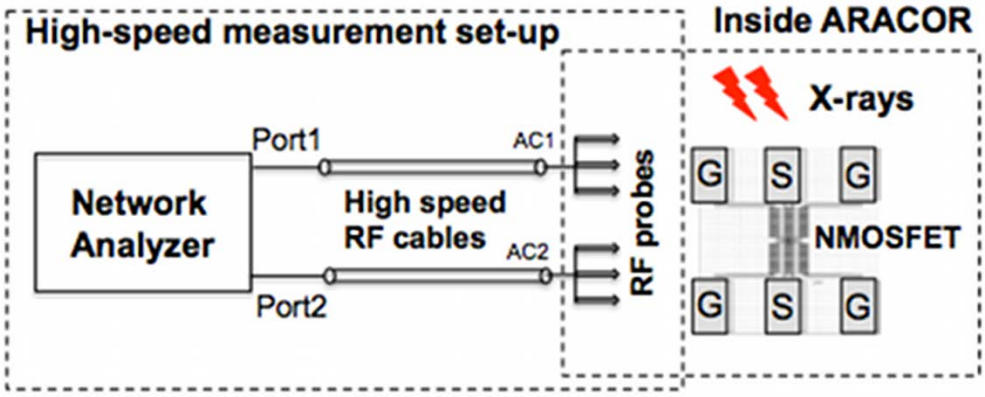


Fig. 11. RF and TID experiment set-up [3].

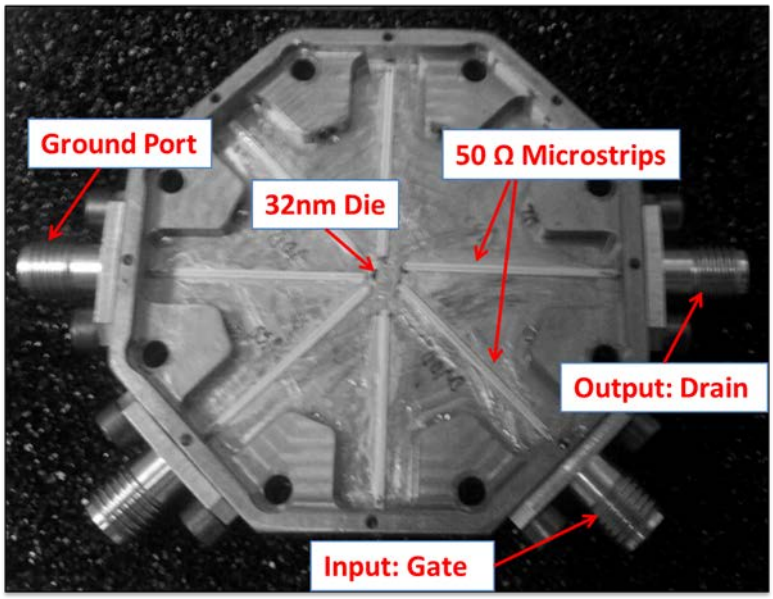


Fig. 12. High-speed brass package with 32 nm die.

During TID irradiation, the devices were biased in the "ON" condition (*n*MOSFETS: V_G=1.0 V, V_S =V_D=0 V and *p*MOSFETS: V_G=-1.0 V, V_S=V_D=0 V). The "ON" bias condition provides the maximum electric field across the gate stack. For direct comparison purposes to previously published 45 nm data [3], the same biases were utilized. Previously published data show that the "ON" bias condition is the worst case bias condition for these technologies [14]. The *I-V* characteristics and S-parameters were measured for TID levels of 0, 100 and 500 krad(SiO_2) at different bias configurations.

Two extremely useful Python scripts (developed by Dr. Andrew Sternberg who is currently employed at The Institute for Space and Defense Electronics at Vanderbilt University) were used to automate the data collection process. The script controlled the measurement equipment and cut the time to collect the *I-V* characteristics and S-parameters by nearly 10 times. The scripts are part of Appendix B.

The die temperature was controlled with a power source and resistive heater

attached to the high-speed package. An Omega dual input type J/K thermometer and hermetically sealed tip thermocouple were used to measure the die temperature. The thermocouple was attached to the brass package. The die temperature was also measured with an infrared temperature-measuring device. During TID experiments the devices were irradiated at 25 ºC and 130 ºC. DC and RF measurements were made immediately at the same temperature (25 ºC and 130 ºC) once irradiation was complete to eliminate annealing effects. Fig. 13 shows the test setup for a device being irradiated at high temperature.

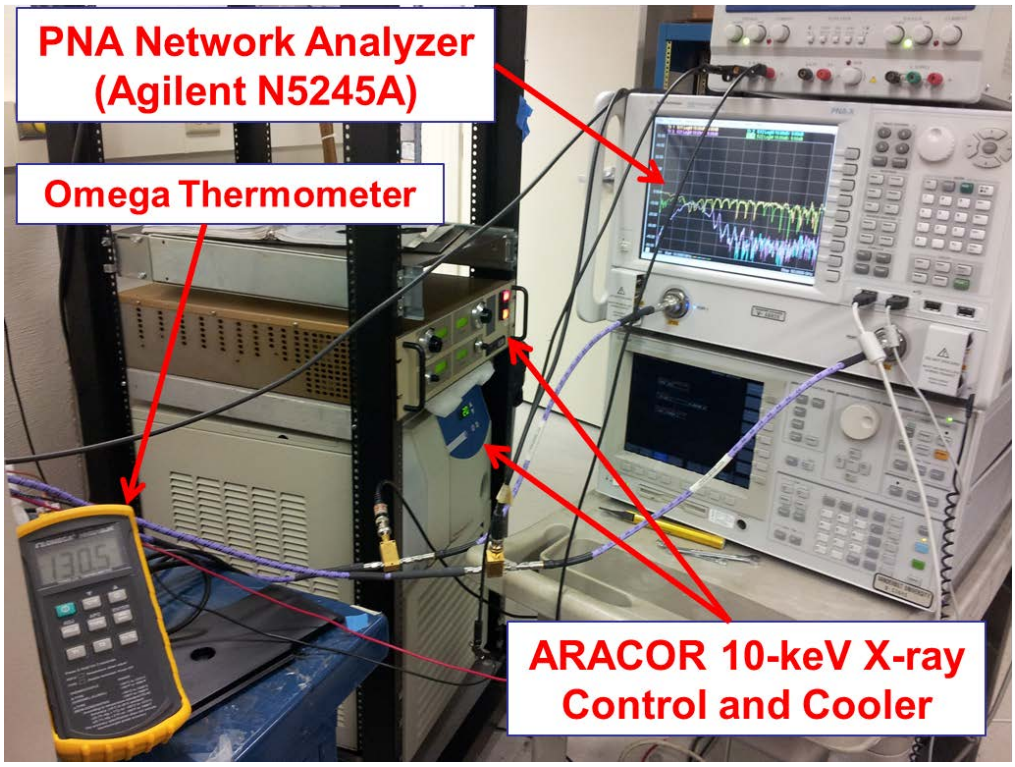


Fig. 13. High-speed RF testing setup for a 32 nm device during irradiation at high temperature.

The RF stress measurements were performed with the devices biased at peak pre-irradiation f_T conditions (*n*MOSFET: V_{GS} = 0.6 V, V_{DS} = 1.0 V). An RF signal of 3 dBm

was superimposed on the gate DC bias. The S-parameters and *I-V* characteristics were measured for different time intervals up to 25 hours. The RF stress data collection process was also automated using the Python script included in Appendix B.

CHAPTER V

TEMPERATURE DEPENDENCE OF 32NM TRANSISTORS

This chapter illustrates the temperature dependence of the 32 nm transistors in Tables 2 and 3 for key DC and RF parameters. The DC and RF parameters were measured at 25 ºC and 130 ºC; the effect of temperature on each parameter are shown for Devices A and C. The tables show the change in each parameter due to the elevated temperature with respect to room temperature: on-current (I_D), forward voltage gain (S_{21}), cut-off frequency (f_T), transconductance (g_m), leakage current and threshold voltage (Vtsat). The off-state leakage showed the most significant increase due to the elevated temperature [15].

Table 2. Device A (nMOS): Change in Peak I_D, S_{21}, f_T, g_m, Leakage Current, and Vtsat and from 25 ºC to 130 ºC with Respect to 25 ºC

Δ Peak I_D [mA]	Δ Peak S_{21} [dB]	Δ Peak f_T [GHz]	Δ Peak g_m [mS]	Δ I_D Leakage [nA]	Δ Vtsat [V]
-0.9	-0.6	-8.5	-2.7	+9386	-0.028

Table 3. Device C (pMOS): Change in Peak I_D, S_{21}, f_T, g_m, Leakage Current, and Vtsat and from 25 ºC to 130 ºC with Respect to 25 ºC

Δ Peak I_D [mA]	Δ Peak S_{21} [dB]	Δ Peak f_T [GHz]	Δ Peak g_m [mS]	Δ I_D Leakage [nA]	Δ Vtsat [V]
-0.3	-0.3	-5.7	-0.9	-8352	-0.011

Fig. 14 shows drain current versus gate voltage (I_{DS}-V_{GS}) for a drain bias of 0.2 V and 1 V for 25 ºC and 130 ºC for Device A, and illustrates competing temperature mechanisms. In this case, the drain current (I_{DS}) increases with temperature at low bias, primarily as a result of changes in threshold voltage with temperature [10]. At high bias the drain current decreases with elevated temperature, as a result of decreases in the carrier mobility [10]. This type of response is known as Inverted Temperature Dependence (ITD) [10,11]. Fig. 15 demonstrates ITD for transconductance. At higher temperatures mobility

decreases due to phonon scattering [16]. There is an intersection between the two g_m curves in Fig. 15 at the zero-temperature coefficient (ZTC) point [17]. Sensitive analog elements (e.g., current mirrors, bias circuits) are often operated at this point to minimize parametric changes with changing temperature.

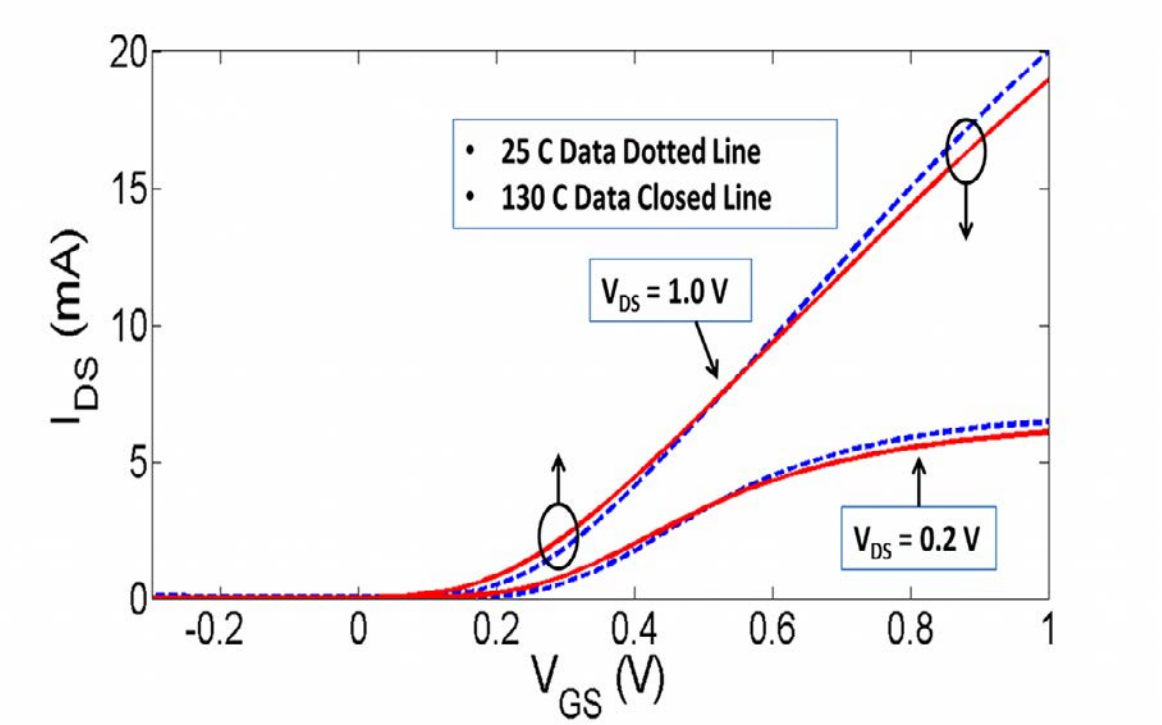


Fig. 14. Device A (nMOS): I_{DS} vs. V_{GS} (on a linear-linear scale) at 25 ºC and 130 ºC. The current enhancement and degradation depend on V_{GS} and temperature. The arrows indicate increasing current and decreasing current.

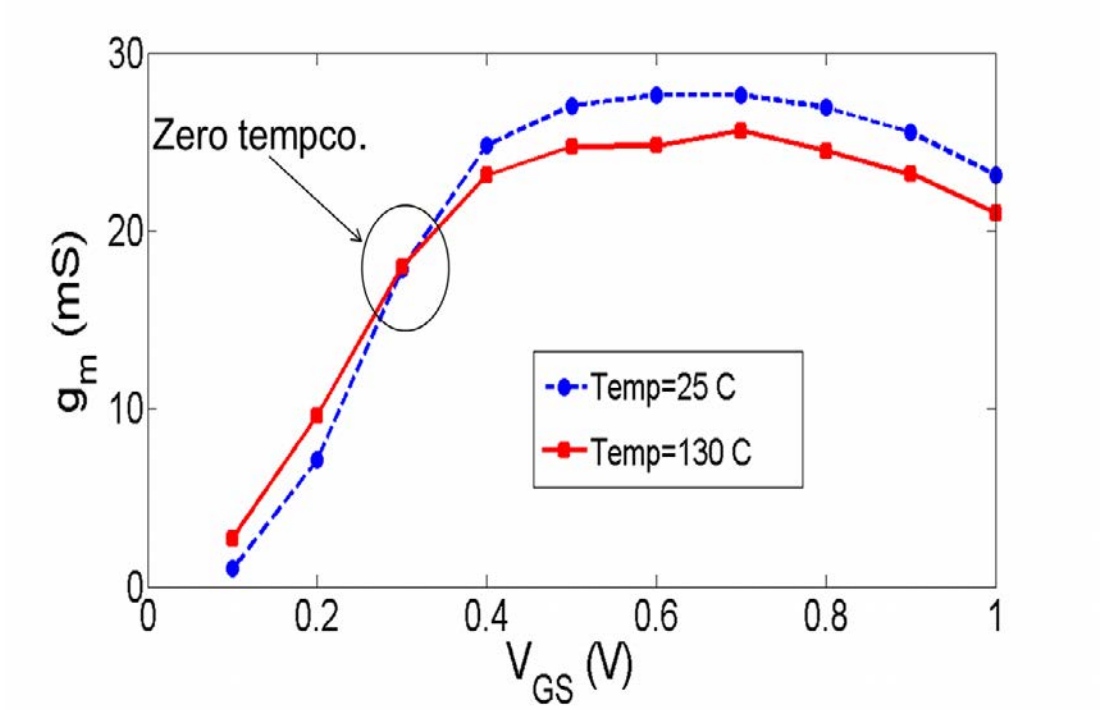


Fig. 15. Device A (nMOS): Transconductance (g_m) vs. V_{GS} at 25 ºC and 130 ºC. The intersection between the two curves indicates the ZTC point.

CHAPTER VI

TID RESPONSE OF 32NM TRANSISTORS

This chapter demonstrates the TID response of the 32 nm transistors that were tested. Figs. 16 and 17 illustrate the pre-irradiation and post-irradiation I_{DS} and I_{GS} vs. V_{GS} characteristics for Device A (*n*MOS) with V_{DS} = 1 V at 25 ºC. In Fig. 5 there is an increase in OFF-state leakage current with TID as a result of positive charge trapped in the shallow trench isolation (STI) and/or the buried oxide (BOX) [7]. This trapped charge causes parasitic leakage between the drain and the source. An increase in leakage of almost two orders of magnitude was observed for V_{GS} = -0.3 V, whereas only a 12% increase in I_{DS} was observed at V_{GS} = 0 V, since the drain current is dominated by subthreshold conduction at this gate bias. Negligible difference in gate current was observed with increasing TID. Leakage values at V_{GS} = -0.3 V are shown to illustrate the trends in quiescent current leakage when the transistor is turned fully off. To maintain measurement consistency with previously published data for the 45 nm technology [3], the gate voltage was swept from -0.3 V to 1 V. However, in the interest of practical circuit design application, the leakage value at V_{GS} = 0 V is more useful.

In Fig. 17 there is a slight decrease in on-state current (2.5% decrease at V_{GS} = 1 V). This is attributed to the (positive) ~9% increase in $V_{TH,SAT}$. Saturation threshold voltages were extracted using linear extrapolation at the peak transconductance. The positive shift in threshold voltage is attributed to electrons trapped in pre-existing interface traps in the high-κ gate dielectric (HfO_2) [19],[20], and is consistent across the devices of various width dimensions examined. In the commercial 45 nm RF CMOS SOI

process, the same device exhibited a (negative) 10% decrease in the threshold voltage due to TID [3]. The slight decrease in on-state current can be attributed solely to the shift in V_{TH}, rather than mobility degradation, as the peak transconductance (g_m) was unaffected by TID up to 500 krad(SiO_2). Fig. 18 illustrates g_m versus V_{GS} for a drain bias of 1 V at different levels of TID. Only a modest shift of the curve to the right is observed with increasing TID. On the other hand, a similar device in the 45 nm technology showed significant decreases in g_m due to TID-induced mobility degradation [3].

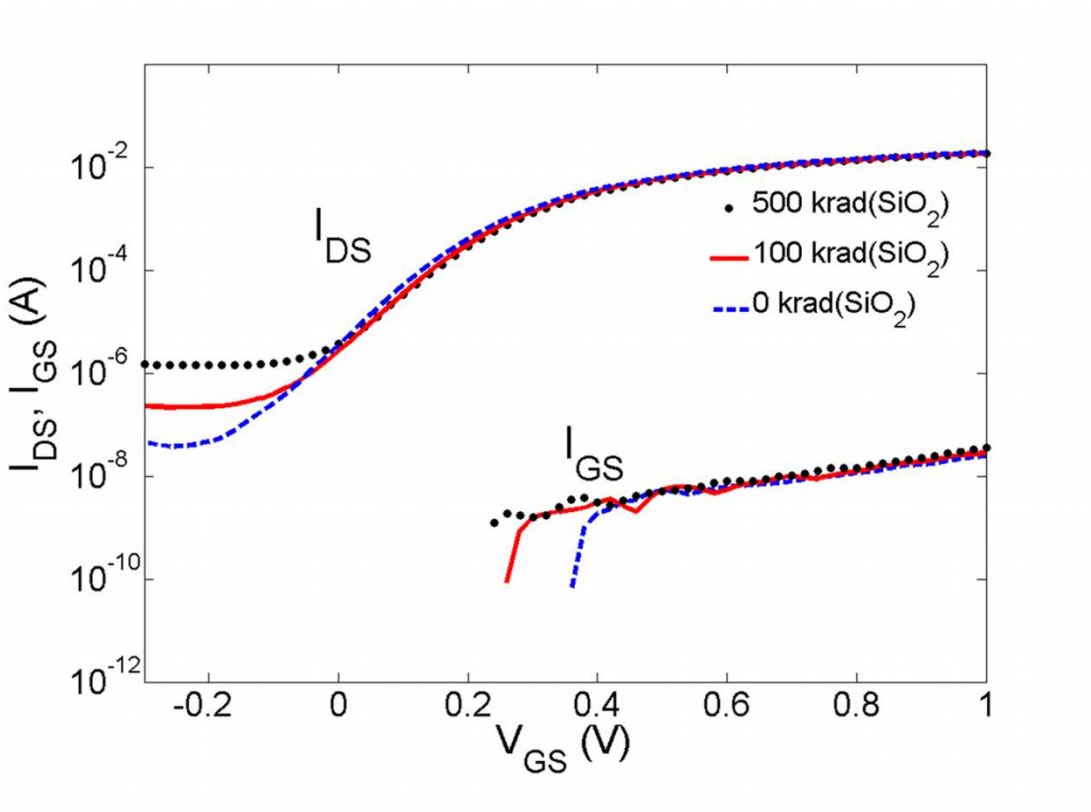


Fig. 16. Device A (nMOS): Pre- and post-irradiation I_{DS} and I_{GS} vs. V_{GS} characteristics (on a log-linear scale) for the nMOSFET with V_{DS} = 1 V at 25 ºC.

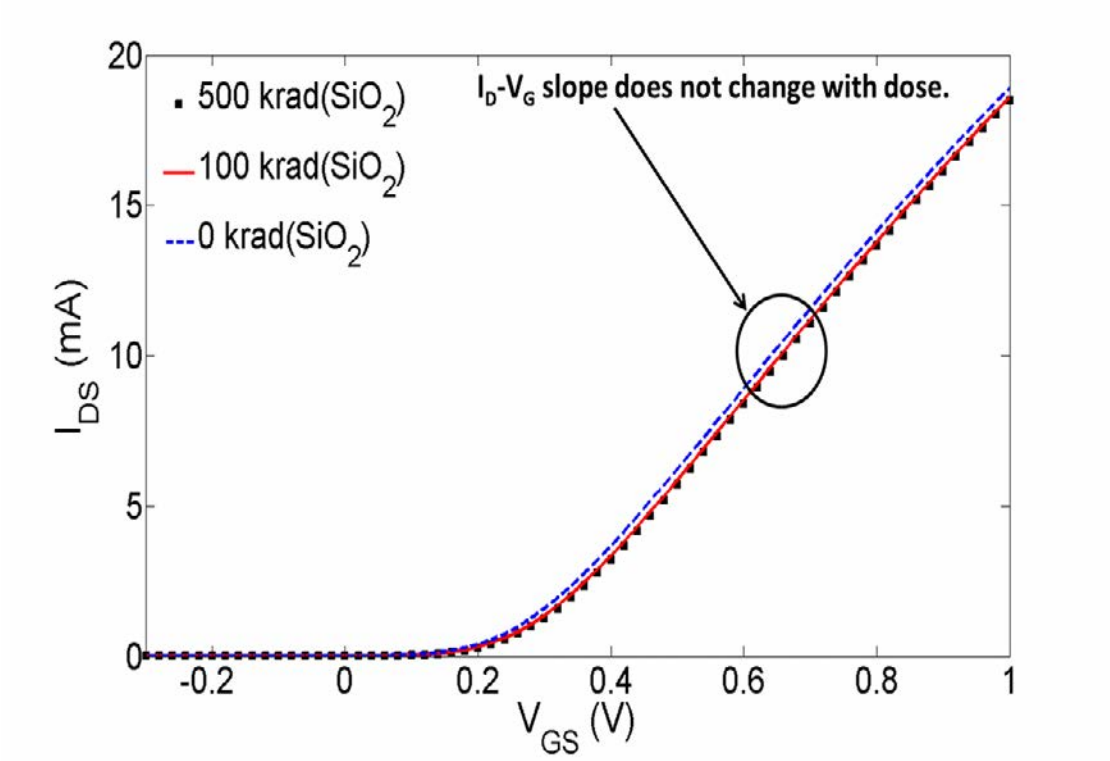


Fig. 17. Device A (nMOS): Pre- and post-irradiation I_{DS} vs. V_{GS} characteristics (on a linear-linear scale) for the nMOSFET with V_{DS} = 1 V at 25 ºC.

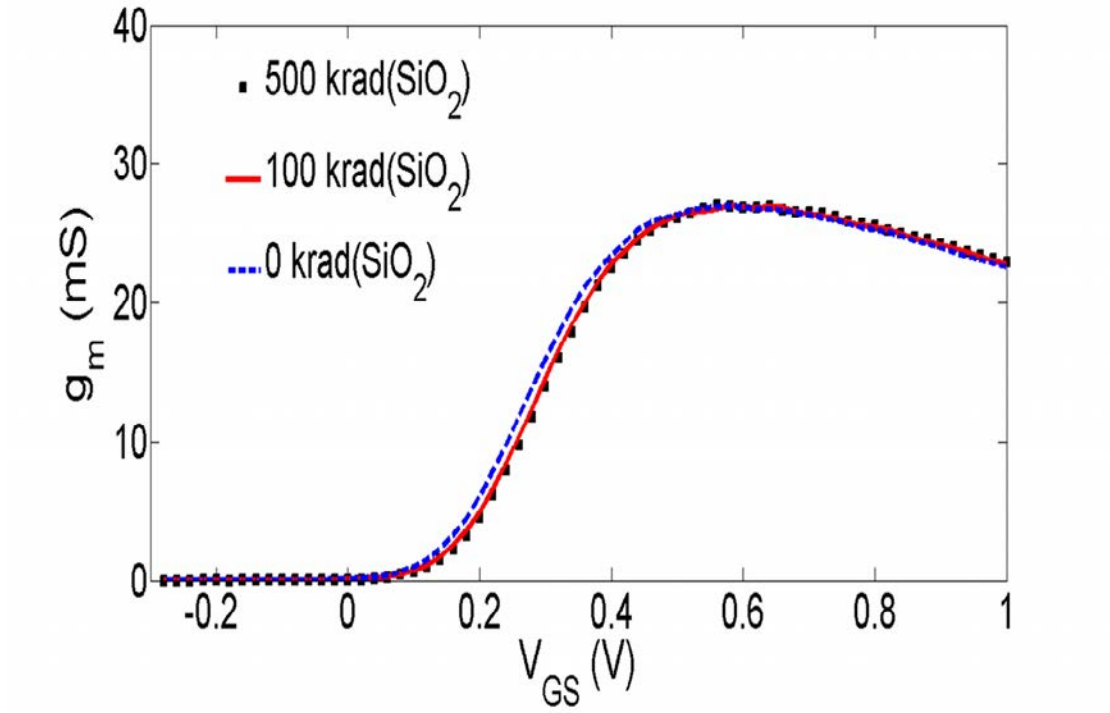


Fig. 18. Device A (nMOS): Transconductance (g_m) versus V_{GS} for a drain bias of 1 V at different levels of TID at 25 ºC.

Table 4. Device A (nMOS): Change in Peak I_D, S_{21}, f_T, g_m, Leakage Current, and Vtsat at 25 ºC and 500 krad(SiO_2) of DC and RF Parameters

Δ Peak I_D [mA]	Δ Peak S_{21} [dB]	Δ Peak f_T[GHz]	Δ Peak g_m [mS]	Δ I_D Leakage [nA]	Δ Vtsat [V]
0	-0.10	-10.1	0	+9375	0

Table 4 summarizes the degradation in the DC and RF parameters of an *n*MOS device due to irradiation at 25 ºC. The impact of TID on the DC and RF parameters is less than 5% for all parameters except off-state leakage. The off-state leakage for the devices was affected most by TID.

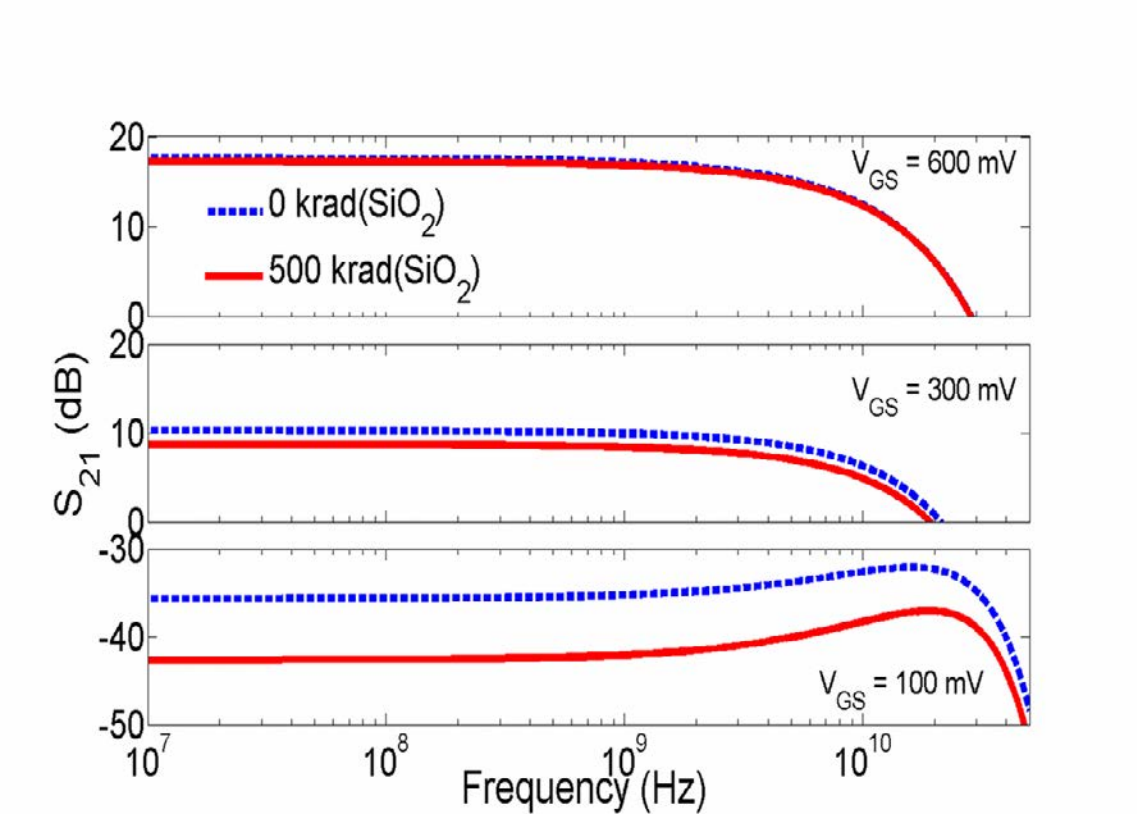


Fig. 19. Device A (nMOS): Degradation of forward voltage gain (S_{21}) with TID as a function of frequency for V_{GS} = 0.1 V, 0.3 V, and 0.6 V at 25 ºC.

Fig. 19 illustrates the pre-irradiation and post-irradiation (500 krad(SiO_2)) de-embedded S_{21} parameter (forward voltage gain in dB) for the floating body *n*MOSFET up to 50 GHz, at V_{GS} bias levels of 100 mV, 300 mV, and 600 mV (peak gain) at a V_{DS} of 1

V at 25 ºC. Dashed lines represent the pre-irradiation curves and solid lines represent the post-irradiation curves. Approximate decreases in S_{21} of 4.9 dB, 1.6 dB, and 0.4 dB for V_{GS} levels of 100 mV, 300 mV, and 600 mV, respectively, were observed up to 500 krad(SiO_2). Though very little change was observed in peak transconductance with increasing dose, the translation of transconductance versus V_{GS} to the right with increasing TID may result in S_{21} degradation. S_{21} follows the dependence of g_m and V_{GS}. This is more pronounced at lower gate biases (in the high-slope region of the g_m-V_{GS} curve).

The levels of observed degradation may be insignificant for many closed-loop analog functions. However, open-loop configurations, especially those with low bias points may be subject to significant degradation. Moreover, the combined effects of temperature and TID can result in enhanced degradation, as observed in [4] where a 22 GHz VCO, designed in this 32 nm RF CMOS technology showed decreases in operating frequency, output power, oscillation amplitude, and an increase in phase noise.

CHAPTER VII

COMBINED RESPONSE OF TID AND TEMPERATURE DEPENDENCE OF 32NM TRANSISTORS

The combined response of temperature dependence and TID experimental results are shown in this chapter. Figs. 20, *n*MOS, and 22, *p*MOS, compare pre-irradiation data at 25 ºC, 130 ºC and the combined effects of 500 krad(SiO_2) total dose irradiation and temperature at 130 ºC on cut-off frequency (f_T) versus gate bias (V_{GS}). Figs. 21, *n*MOS, and 23, *p*MOS, show 25 ºC pre- and post-irradiation data for 500 krad(SiO_2) total dose irradiation for a nominally similar device on a different die. It is necessary to do these comparisons on different die with nominally identical devices because heating a device that is irradiated at room temperature anneals out radiation damage while the performance is being measured at elevated temperature.

The cut-off frequency for the device is directly related to g_m. It is an important RF parameter because it represents the maximum fundamental frequency component the transistor can generate. For these devices, the cut-off frequency was calculated by converting the measured S-parameters to H-parameters. An extensive set of MATLAB code was written to de-embed the S-parameter parasitics that can be found in Appendix B (written by Dr. Daniel Loveless who was previously employed at The Institute for Space and Defense Electronics at Vanderbilt University). The S- and H-parameters provide accurate information regarding the current gain, voltage gain, input impedance and output impedance of a transistor as an amplifier or two-port network. The cut-off frequency is found where the h_{21} parameter (forward current gain) is equal to 0 dB on the frequency axis. The degradation due to the combined effects of TID and temperature at 130 ºC of the peak f_T for the Device A #1 (*n*MOS) device was ~17% and ~5% for Device C #1 (*p*MOS)

device. The forward voltage gain, S_{21} which is also related to g_m, degraded by 1 dB for Device A #1 (*n*MOS) and 0.6 dB for the Device C #1 (*p*MOS). The ZTC's are identified in Figs. 20 and 22 and the waveforms demonstrate the combined effects of TID and temperature worsen cut-off frequency for both devices.

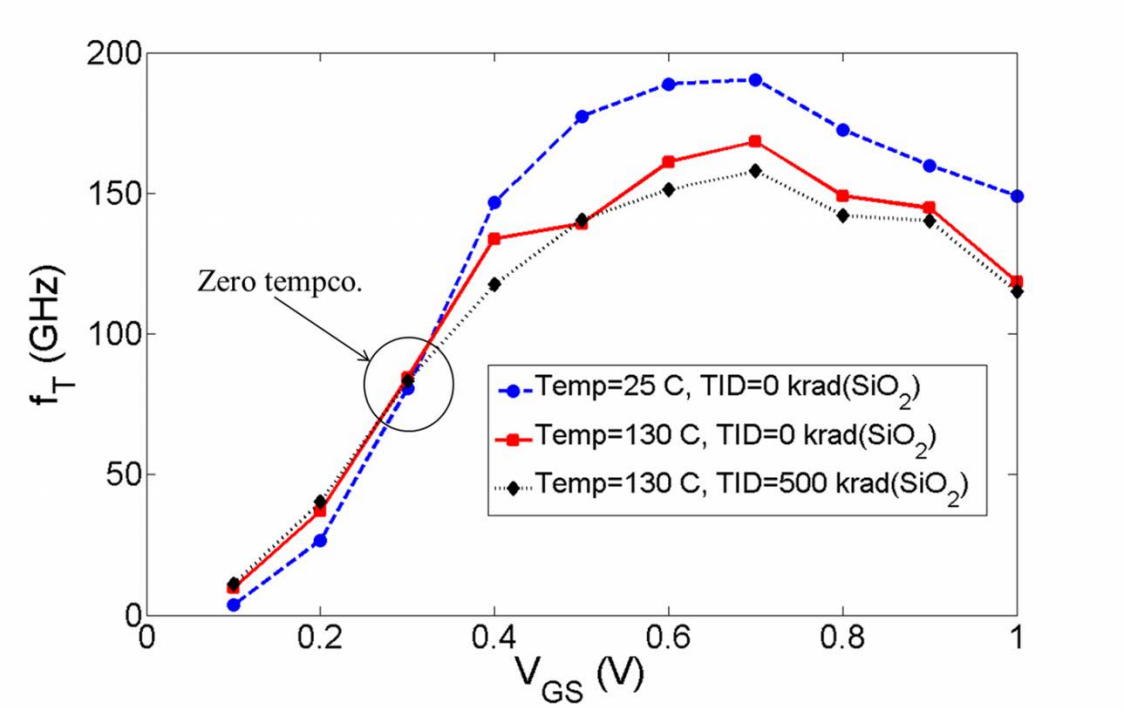


Fig. 20. Device A #1 (nMOS): Degradation in f_T with the combined effects of TID and temperature.

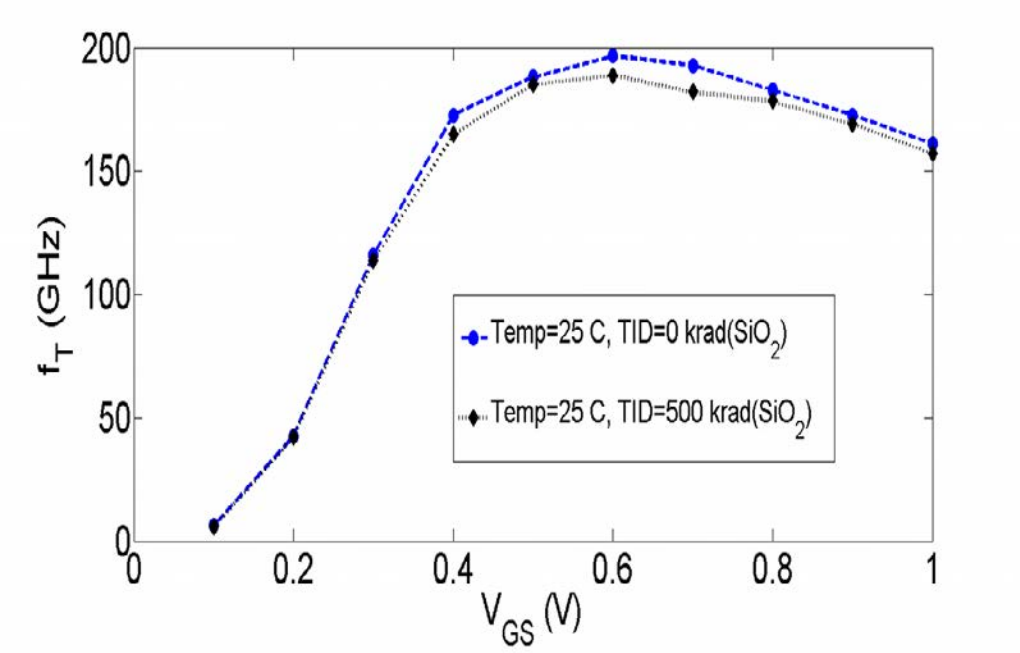


Fig. 21. Device A #2 (nMOS): Degradation in f_T with TID at 25 ºC.

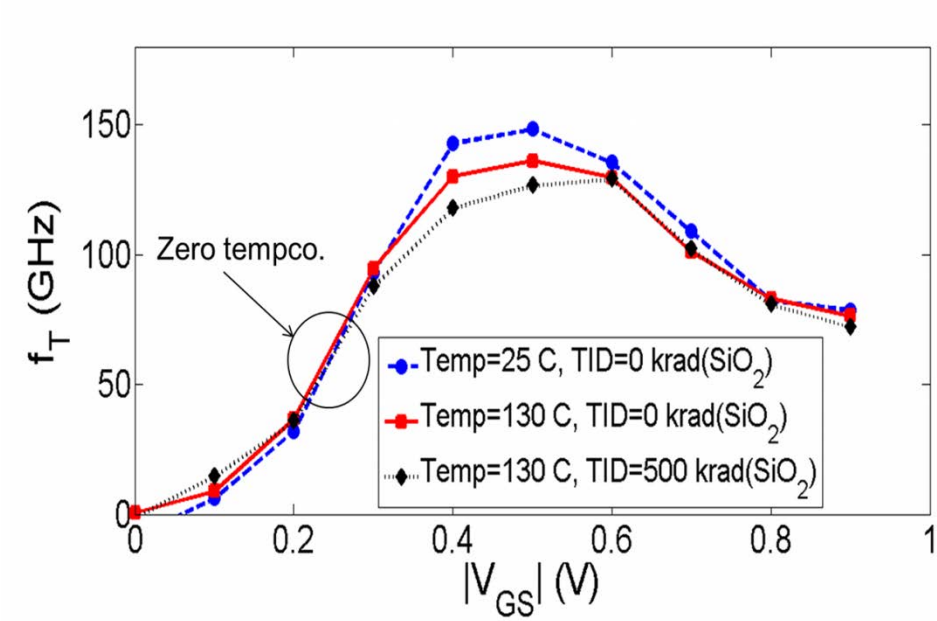


Fig. 22. Device C #1 (pMOS): Degradation in f_T with the combined effects of TID and temperature.

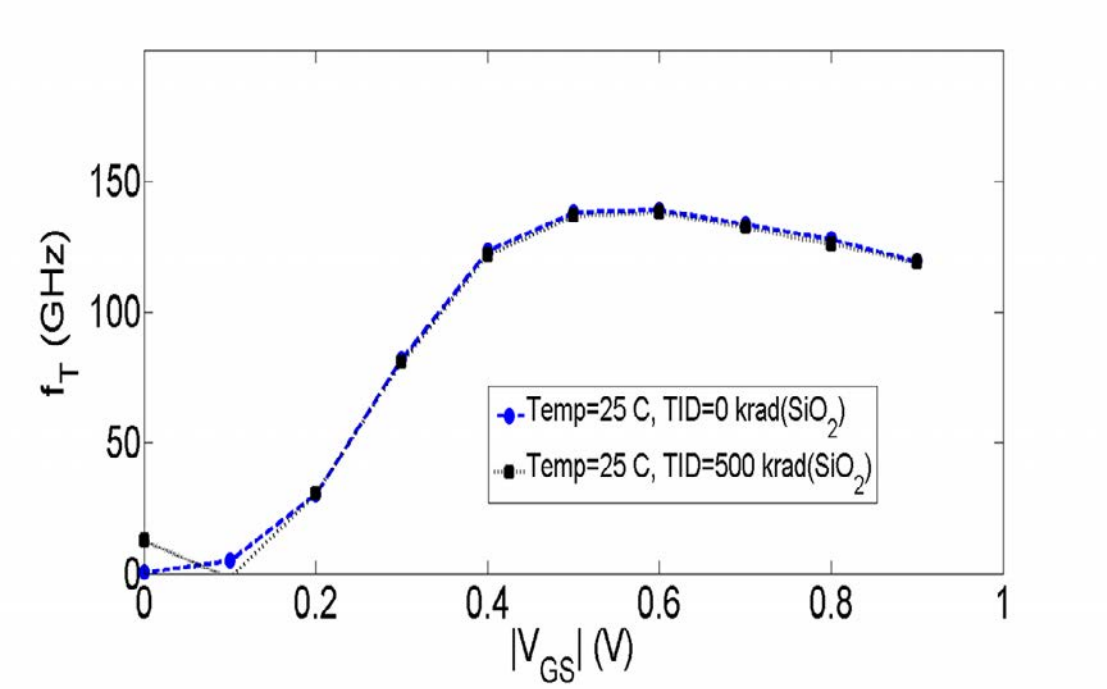


Fig. 23. Device C #2 (pMOS): Degradation in f_T with TID at 25 ºC.

CHAPTER VIII

COMPARISON BETWEEN 32NM AND 45NM DC AND RF MEASUREMENTS

This chapter provides a comparison between the experimental results for 2 similar RF SOI CMOS technologies. Tables 5 and 6 summarize a comparison of similar devices in the 32 nm technology and previously published results for the 45 nm technology [3]. The percent degradation due to the combined effects of TID and temperature for the key DC and RF parameters is consistently less for the *n*MOS device in the 32 nm technology than in the 45 nm technology. This is consistent with previous comparisons of high-K and SiO_2 gate dielectrics [21]. These differences have been attributed to the 32 nm gate stack containing more compensating electron traps than the 45 nm technology, which means reduced net positive charge [22]. The *p*MOS devices for both technology nodes consistently show less degradation due to TID and temperature than the *n*MOS devices, because *n*MOSFETS are sensitive to TID-induced STI leakage and *p*MOSFETS are not [23].

Table 5. 32 nm: Percent Degradation on-state I_D, peak g_m, and f_T, and Delta S_{21} at 25 ºC and the Combined Effects of Temperature at 130 ºC Plus TID 500 krad(SiO_2)

32nm	**$I_{D,ON}$**	**peak g_m**	**f_T**	**S_{21} [dB]**
NMOS	6%	13%	17%	Δ -1.0
PMOS	3%	6%	5%	Δ -0.6

Table 6. 45 nm: Percent Degradation on-state I_D, peak g_m, and f_T, and Delta S_{21} at 25 ºC and the Combined Effects of Temperature at 130 ºC Plus TID 500 krad(SiO_2)

45nm	**$I_{D,ON}$**	**peak g_m**	**f_T**	**S_{21} [dB]**
NMOS	20%	33%	31%	Δ -3.3
PMOS	4%	5%	3%	Δ -0.9

CHAPTER IX

32NM RF EXPERIMENTAL RESULTS

This chapter summarizes the 32 nm RF experimental data. Table 7 shows the results of the RF experiments on the DC and RF parameters of the 32 nm devices at 25 ºC. The *n*MOS transistors were biased at the peak f_T condition V_{GS} = 0.6 V and V_{DS} = 1 V (*p*MOS: V_{GS} = -0.6 V and V_{DS} = -1 V) and a RF signal was applied to the gate. The RF input signal was set to 3 dBm (0.89 V peak-to-peak). The frequency of the RF signal was 30 GHz. The devices were biased for a total time of 25 hours. During each time interval the bias was stopped and the DC and RF parameters were automatically measured. The RF experiment showed very small parametric changes. Fig. 24 shows the absolute value of the change in $V_{TH,SAT}$ versus RF stress time for Device A (*n*MOS) and Device C (*p*MOS). The results show ~3% degradation for both devices, which is less than the TID or temperature shifts in threshold voltage.

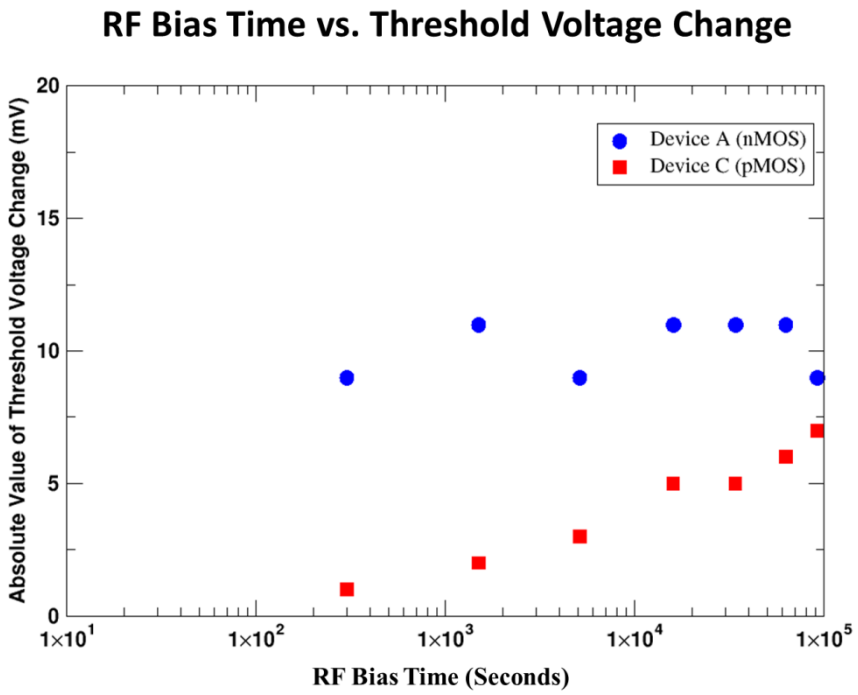


Fig. 24. RF bias time versus the absolute value of threshold voltage change (on a log-linear scale) for Device A (nMOS) and Device C (pMOS).

Table 7. RF Experiment Percent Degradation and S21 Delta at 3 dBm for 25 Hours Total at 25 ºC

Device	Peak I_D	Peak S_{21} [dB]	Peak f_T	Peak g_m	I_D Leakage	Vtsat
NMOS	0.53%	Δ 0	1.60%	0.74%	41.63%	-3.25%
PMOS	4.73%	Δ 0	6.05%	3.61%	-55.83%	-3.16%

CHAPTER X

CONCLUSIONS

This work demonstrates TID, temperature and RF experimental results for a commercial 32 nm RF CMOS SOI process. Temperature is the dominant factor in the parametric shifts of the 32 nm devices. For device A (*n*MOS), a temperature increase from 25 ºC to 130 ºC affected peak g_m, f_T, $V_{TH,SAT}$ and off-state leakage by ~12%, ~7%, ~15% and ~45%, respectively, more than the standalone effects of TID. RF bias experiment had less effect than TID. The off-state leakage current of the 32 nm devices increased significantly with TID and temperature. Inverted temperature dependence (ITD) and zero-temperature coefficients were identified in the DC and RF waveforms which would be key parameters for a low-power, RF designs in total dose and elevated temperature environments.

Moreover, the 32 nm results were compared to those measured in a 45 nm RF SOI process with similar device dimensions. The results indicate that the 32 nm technology shows less degradation due to TID and the combined effects of temperature than the 45 nm technology. While the electrical performance of the technologies is similar, the observed TID-induced degradation has differing qualities. For example, the high-κ gate dielectric in the 32 nm technology results in enhanced electron trapping and therefore additional positive charge compensation in the *n*MOS devices, as compared with the 45 nm devices with SiO_2 gate dielectrics.

The RF stress and the combined effect of TID and temperature show very little degradation. Temperature and TID-induced degradation was observed in the forward voltage gain, S_{21}, due to the increase of V_{TH} and the translation of the g_m-V_{GS} curve to the

right with increasing TID. Less degradation was observed in higher gain states near the peak transconductance. Though minimal degradation of transistor I-V characteristics and RF parameters were observed, the significant bias dependencies on the combined effects of TID and temperature response warrant careful consideration when used in high performance topologies.

REFERENCES

[1] A. Abidi, "RFCMOS comes of age," *IEEE J. Solid State Circuits*, vol. 39, pp. 549-561, Apr. 2004.

[2] H. Hassan, M. Anis, M. Elmasry, "Impact of technology scaling and process variations on RF CMOS devices," *Microelectronics Journal*, vol. 37, Issue 4, pp. 275-282, April 2006.

[3] S. Jagannathan, T. D. Loveless, E. X. Zhang. D. M. Fleetwood, R. D. Schrimpf, T. D. Haeffner, J. S. Kauppila, N. Mahatme, B. L. Bhuva, M. L. Alles, W. T. Holman, A. F. Witulski, L. W. Massengill, "Sensitivity of high-frequency RF circuits to total ionizing dose degradation," *IEEE Trans. Nucl. Sci*., vol. 60, no. 6, pp.4498-4504, Dec. 2013.

[4] S. Jagannathan, *Ph.D Dissertation*. Vanderbilt University, November 2013.

[5] T. P. Ma and P. V. Dressendorfer, *Ionizing radiation effects in MOS devices and circuits*, New York, NY: John Wiley and Sons, 1989.

[6] R. Arora, K. A. Moen, A. Madan, J. D. Cressler, E. Z. Zhang, D. M. Fleetwood, R. D. Schrimpf, A. K. Sutton, H. M. Nayfeh, "Impact of body tie and Source/Drain contact spacing on the hot carrier reliability of 45-nm RF-CMOS," *Integrated Reliability Workshop Final Report (IRW), 2010 IEEE International*, pp. 56-60, Oct. 17-21 2010.

[7] J. R. Schwank, V. Ferlet-Cavrois, M. R. Shaneyfelt, P. Paillet, P. E. Dodd, "Radiation effects in SOI technologies," *IEEE Trans. Nucl. Sci.*, vol. 50, no. 3, pp. 522-538, June 2003.

[8] C. –H Jan, M. Agostinelli, H. Deshpande, M. A. El-Tanani, W. Hafez, U. Jalan, L. Janbay, M. Kang, H. Lakdawala, J. Lin, Y. -L Lu, S. Mudanai, J. Park, A. Rahman, J. Rizk, W. -K Shin, K. Soumyanath, H. Tashiro, C. Tsai, P. Vandervoorn, J. –Y. Yeh, P. Bai, "RF CMOS technology scaling in High-k/metal gate era for RF SoC (system-on-chip) applications," *Electron Devices Meeting (IEDM), 2010 IEEE International*, pp. 27.2.1-27.2.4, Dec. 6-8 2010.

[9] L. Demeus, V. Dessard, A. Viviani, S. Adriaensen, and D. Flandre, "Integrated sensor and electronic circuits in fully depleted SOI technology for high-temperature applications," *Industrial Electronics, IEEE Trans.*, vol. 48, no. 2, pp. 272-280, April 2001.

[10] A, Sassone, A. Calimera, A. Macii, E. Macii, M. Poncino, R. Goldman, V. Melikyan, E. Babayan, S. Rinaudo, "Investigating the effects of inverted temperature dependence (ITD) on clock distribution networks," *IEEE Design, Automation & Test in Europe Conference & Exhibition*, pp. 165-166, March 2012.

[11] R. Kumar and V. Kursun, "Reversed temperature-dependent propagation delay

characteristics in nanometer CMOS circuits," *IEEE Circuits and Systems II: Express Briefs*, vol. 53, no. 10, pp. 1078-1082, Oct. 2006.

[12] Greene, Q. Liang, K. Amarnath, Y. Wang, J. Schaeffer, M. Cai, Y. Liang, S. Saroop, J. Cheng, A. Rotondaro, S-J. Han, R. Mo, K. McStay, S. Ku, R. Pal, M. Kumar, B. Dirahoui, B. Yang, F. Tamweber, W-H. Lee, M. Steigerwalt, H. Weijtmans, J. Holt, L. Black, S. Samavedam, M. Turner, K. Ramani, D. Lee, M. Belyansky, M. Chowdhury, D. Aime, B. Min, H. van Meer, H. Yin, K. Chan, M. Angyal, M. Zaleski, O. Ogunsola, C. Child, L. Zhuang, H. Yan, D. Permanaa, J. Sleight, D. Guo, S. Mittl, D. Ioannou, E. Wu, M. Chudzik, D-G. Park, D. Brown, S. Luning, D. Mocuta, E. Maciejewski, K. Henson, and E. Leobandung, "High performance 32 nm SOI CMOS with high-k/metal gate and 0.149 μm^2 SRAM and ultra low-k back end with eleven levels of copper," *IEEE Sym. on VLSI Tech*., pp. 140-141, 2009.

[13] H. M. Nayfeh, N. Rovedo, A. Bryant, S. Narasimha, A. Kumar, X. Yu, N. Su, J. W. Sleight, R. R. Robison, W. Rauch, H. Mallela, and G. Freeman, "Impact of lateral asymmetric channel doping on 45-nm-technology N-type SOI MOSFETs," *IEEE Trans. Electron Devices*, vol. 56, pp. 3097-3105, Dec. 2009.

[14] E. X. Zhang, D. M. Fleetwood, N. D. Pate, R. A. Reed, A. F. Witulski, and R. D. Schrimpf, "Time-domain reflectometry measurements of total-ionizing-dose degradation of *n*MOSFETs," *IEEE Trans. Nucl. Sci.*, vol. 60, no. 6, pp. 4470-4475, Dec. 2013.

[15] R. Arora, Z. E. Fleetwood, E. X. Zhang, N. E. Lourenco, J. D. Cressler, D. M. Fleetwood, R. D. Schrimpf, A. K. Sutton, G. Freeman and B. Greene, "Impact of technology scaling in sub-100 nm *n*MOSFETs on total-dose radiation response and hot-carrier reliability," *IEEE Trans. Nucl. Sci*., vol. 61, no. 3, pp. 1426-1432, June 2014.

[16] H. Cho and D. Burk, "A three-step method for the de-embedding of high-frequency S-parameter measurements," *IEEE Trans. Electron Devices*, vol. 38, no. 6, pp. 1371-1375, June 1991.

[17] R.S. Muller and T. I. Kamins, *Device Electronics for Integrated Circuits*. New York, NY: Wiley & Sons, 1986.

[18] M. Fleetwood, F. V. Thome, S. S. Tsao, P. V. Dressendorfer, V. J. Dandini, and J. R. Schwank, "High temperature silicon-on-insulator electronics for space nuclear power systems: Requirements and feasibility," *IEEE Trans. Nucl. Sci*. vol. 35, no. 5, pp. 1099-1112, Oct. 1988.

[19] R. Arora, *Ph.D Dissertation*. Georgia Institute of Technology, 2012.

[20] R. Arora, E. X. Zhang, S. Seth, J. D. Cressler, D. M. Fleetwood, R. D. Schrimpf, G. Rosa, A. K. Sutton, H. Nayfeh, and G. Freeman, "Trade-offs between RF performance and total-dose tolerance in 45-nm RF-CMOS," *IEEE Trans. Nucl. Sci*., vol. 58, no. 6, pp. 2830-2837, Dec. 2011.

[21] X. J. Zhou, D. M. Fleetwood, L. Tsetseris, R. D. Schrimpf, and S. T. Pantelides, "Effects of switched-bias annealing on charge trapping in HfO2 gate dielectrics," *IEEE Trans. Nucl. Sci.*, vol. 53, no. 6, pp. 3636-3643, Dec. 2006.

[22] M. Fleetwood, S. L. Miller, R. A. Reber, Jr., P. J. McWhorter, P. S. Winokur, M. R. Shaneyfelt, and J. R. Schwank, "New insights into radiation-induced oxide-trap charge through thermally-stimulated-current measurement and analysis," *IEEE Trans. Nucl. Sci.*, vol. 39, no. 6, pp. 2192-2203, Dec. 1992.

[23] M. R. Shaneyfelt, P. E. Dodd, B. L. Draper, and R. S. Flores, "Challenges in hardening technologies using shallow-trench isolation," *IEEE Trans. Nucl. Sci.*, vol. 45, no. 6, pp. 2584-2592, Dec. 1998.

APPENDIX A

TEST PLAN AND PARAMETER CHARACTERIZATION RESULTS OF 32NM RF SOI CMOS STRUCTURES

Executive Summary

This document details the test results from DC and RF experiments, performed at Vanderbilt University on a commercial 32 nm RF SOI CMOS technology. Test procedures are outlined below. The devices-under-test (DUT) consist of large multi-finger MOSFETs with varying lengths, widths, threshold voltage implants, oxide thickness and body contacting schemes. It also includes varactors, capacitors and inductors and other RF calibration structures such as open, short, load and through for de-embedding purposes.

DC parameters and de-embedded s-parameter measurements are provided, as well as f_T calculations. Observations from the measurements include the effects of transistor type, device width, number of fingers, gate length, body type, and threshold implant on forward voltage gain (S_{21}) and unity current-gain cutoff frequency (f_T).

The following general observations can be made for the DC parameters:

- The devices were on the weak side of the distribution, representing approximately a -3σ distribution compared to the device model.
- The gate leakage is insignificant compared to the channel leakage thanks to the use of high-k gate dielectric, and can therefore be disregarded.
- The high Vt device (UVT) reduces leakage by > order of magnitude compared to standard device (RVT) at the expense of 50% reduced drive current.
- Doubling the number of fingers reduces normalized current (/um) by about 20%.
- Body tie reduces channel leakage by as much as 75% at the expense of reduced drive current (50%).
- Device current is reasonably proportional to 1/L indicating good short channel behavior.
- A robust PFET/NFET ratio support effectiveness of stress engineering on carrier mobility.

The following general observations can be made for the RF parameters:

- The coplanar waveguide impedance was on target at 50Ω, and was purely resistive through at least 30GHz.
- The forward gain S_{21} Parameters showed minimal degradation through 1GHz, with some degradation through 10GHz. This degradation is accentuated by packaging.
- The median value for S_{21} was around 12dB for the various NFET devices and around 10dB for the various PFET devices measured.
- The peak cut-off frequency, f_T, occurred at a typical gate voltage of about 600mV, but varied between 200mV and 1.0V for the various device types.
- f_T was about 20GHz higher for the NFET Compared to PFET devices
- In comparison with the 45nm CMOS technology previously evaluated, the 32nm CMOS offered about 8dB higher S_{21} forward gain, but about 45% lower f_T at an average.

Institute for Space and Defense Electronics
Vanderbilt University
Nashville, TN 37212

Primary Investigator: Daniel Loveless

Faculty: Lloyd Massengill, Mike Alles, Tim Holman

Support Engineers/Staff: Tim Haeffner, Mike McCurdy, Andrew Sternberg, Nadim Haddad

Graduate Student: NA

Document last edited by: Tim Haeffner

Date: 2015-03-10

0.0 Description:

This document details the test results from DC and RF experiments, performed at Vanderbilt University on a commercial 32 nm RF SOI CMOS technology. Test procedures are outlined below

1.0 Overview:

The RF test chips were fabricated in a commercial 32 nm RF SOI CMOS technology (TAPO ID: P87636, VU ID: 32-VuRF-1). The devices-under-test (DUT) consist of large multi-finger MOSFETs with varying lengths, widths, threshold voltage implants, oxide thickness and body contacting schemes. It also includes varactors, capacitors and inductors and other RF calibration structures such as open, short, load and through for de-embedding purposes. ***Table 1*** details the devices.

DC parameters and de-embedded s-parameter measurements are provided, as well as f_T calculations. Observations from the measurements include the effects of transistor type, device width, number of fingers, gate length, body type, and threshold implant on forward voltage gain (S_{21}) and unity current-gain cutoff frequency (f_T).

On-chip calibration structures include Pad Open (open reflect at pad), Open (open reflect at device), Short (short reflect at device), Through (input connected to output), Load 1 (input with 50 Ω load), and Load 2 (output with 50 Ω load). Equipment calibration structures were utilized for de-embedding the network analyzer, RF cables, and RF and DC probe parasitic elements. Raw s-parameter (S_{11}, S_{12}, S_{21}, S_{22}) measurements were performed on each device; parasitic de-embedding was performed using calibration structures to determine the true s-parameters. S-parameters were converted to h-parameters to determine unity gain cutoff frequency (f_T).

Table 1. RF Devices on P87636

ID	Device	Type	Body	Total Width	Width/Length	Fingers
A	NFET	AVT	Floating	24u	2u/56nm	12
B	NFET	AVT	Floating	24u	1.6u/56nm	15
C	NFET	AVT	Floating	24u	1u/56nm	24
D	NFET	AVT	Floating	15u	1u/56nm	15
E	NFET	AVT	Floating	12u	0.8u/56nm	15
F	PFET	AVT	Floating	24u	2u/56nm	12
G	NFET	RVT	Floating	12u	0.8u/40nm	15
H	NFET	SVT	Floating	12u	0.8u/40nm	15
I	NFET	UVT	Floating	12u	0.8u/40nm	15
J	NFET	TO	Floating	24u	1.6u/100nm	15
K	NFET	AVT	Tied	12u	2u/56nm	6
L	NFET	AVT	Tied	12u	1u/56nm	12
M	NFET	AVT	Tied	12u	0.8u/56nm	15
N	NFET	AVT	Tied	12u	0.8u/100nm	15
O	NFET	AVT	Tied	12u	0.8u/230nm	15
P	NFET	AVT	Tied	15u	1u/56nm	15
Q	UGNCAP	Var.	-	2u	490nm	6
R	UGNCAP	Var.	-	2u	230n	12
S	VNCAP	Metal	-	10u	9.862u	M1-M3
T	VNCAP	Metal	-	10u	9.868u	M1-B3
U	PFET	AVT	Tied	12u	0.8u/230n	15
V	PFET	AVT	Tied	12u	0.8u/100nm	15
W	PFET	AVT	Tied	12u	1u/56nm	12
X	PFET	AVT	Floating	24u	1u/56nm	24
Y	PFET	AVT	Floating	12u	0.8u/56nm	15
Z	NFET	AVT	Tied	24u	2u/56nm	12
AA	NFET	AVT	Tied	12u	1u/56nm	12
BB	Tcoil3	Ind. 2u	10u coil space	6.5u tap width	40u	8 inner/outer
CC	PAD	Open at top metal				
DD	Open 1	Open at M1				
EE	Short					
FF	Thru					
GG	Load 1	50 Ω	Input 1			
HH	Load 2	50 Ω	Input 2			
II	Open 2	Open by removing active regions of NFET1				

All device terminals on the DUT are connected to probe pads. ***Table 2*** groups similar devices and indicates voltage requirements. Each device was labeled with a unique identifier and directly contacted using high-speed RF probes. Normal ESD handling procedures were followed to avoid an ESD event.

Table 2. Voltage Requirements

Device(s)	V_{DD} +/- 10%	Port
A-I, K-P, U-AA	0.9 V OR 1.0 V	Thin oxide FETs
J	1.5 V	Thick oxide FETs
Q-R	± 0.9 V	NMOS varactors
S-T	2.5 V	Metal (Vertical Natural) capacitors
BB	2.5 V	Inductor
CC-II	1.5 V, +/- .15 V	Calibration structures

2.0 Test Setup

The test plan detailed the experimental procedures for measuring the following DC and RF performance metrics:

1) ID-VG sweep and transconductance (g_m)
2) ID-VD sweep
3) S-parameters
4) Unity current gain cut-off frequency (f_T)

The input and output signals of the test structures are connected to I/O pads in ground-signal-ground configuration using 50 Ω coplanar waveguides.

The (center-to-center) pitch between any two pads is 100 µm. 100 µm pitched high-speed RF probes (in ground-signal-ground configuration) were used to probe the input and output signals. The test structures do not contain matching networks. Parasitic de-embedding was performed in order to extract the device parameters (section 3.1). DC probes were used to provide DC bias to the source and substrate pads; however, as the parasitics of the DC probe on the substrate will impact the S-parameter measurements, it was calibrated (section 3.2.1)

The Programmable Network Analyzer (PNA) was used to measure the S-parameters of the devices in the DUT. A Parameter Analyzer (PA) was used to provide DC bias, and was coupled to the device through a 50 Ω bias-T.

The test set-up is shown in ***Figure 1.*** The PNA is connected to the AC port of the bias-T using a 50-Ω high-speed RF cable. The PA is connected to the DC port of the bias-T using a BNC/coaxial cable. The other end of bias-T (containing the AC and DC signal) is connected to the high speed RF probes which connects to the I/O pads (in ground-signal-ground configuration) on the DUT. The DC bias for other terminals in the DUT (such as source, substrate and body) is applied directly from PA using DC probes.

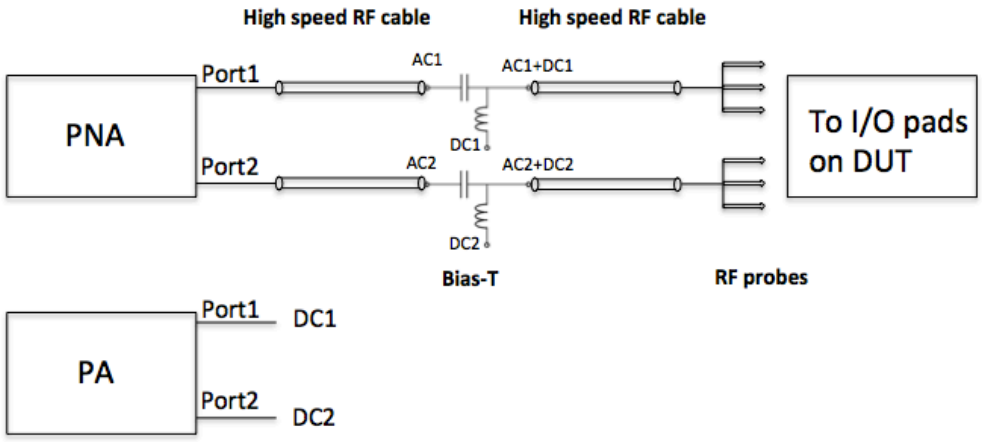


Figure 1 shows the block diagram RF measurement test-setup.

Test equipment used is presented in ***Table 3***.

Table 3. Test Equipment Used

Name	**Model #**	**Rating**	**Applications**
Programmable network analyzer (PNA)	Agilent Technologies: N5245A	10 MHz - 50 GHz	Measure S-parameters
Parameter Analyzer (PA)	HP: 4156A	I: 1 fA to 1 A (20 fA offset accuracy), V: 1 µV to 200 V	1) Measure device DC characteristics (I_d-V_g) 2) Provide DC bias
RF probes	GGB Industries: 40A-GSG-100-P-N	100 µm pitch, DC - 40 GHz	Probe RF signal in GSG configuration
DC probes	Cascade Microtech		Provide DC bias to the DUT
Bias-T	Picosecond pulse labs 5542-219	10 KHz – 50 GHz	Provide AC + DC signal to the DUT
BNC/coaxial cables			Provide DC bias

MATLAB was used for de-embedding the parasitic elements. A set of custom MATLAB script has been developed to convert the measured S-parameters from PAD, OPEN, SHORT, THROUGH, and LOAD calibration structures to Y-parameters in order to de-embed the probe, package, pad, wire, and equipment parasitic components. De-embedded S-parameters may be converted to H-parameters in order to determine the f_T (see section 3.3). ***Figure 2*** and ***3*** illustrate the RF probe and test setup.

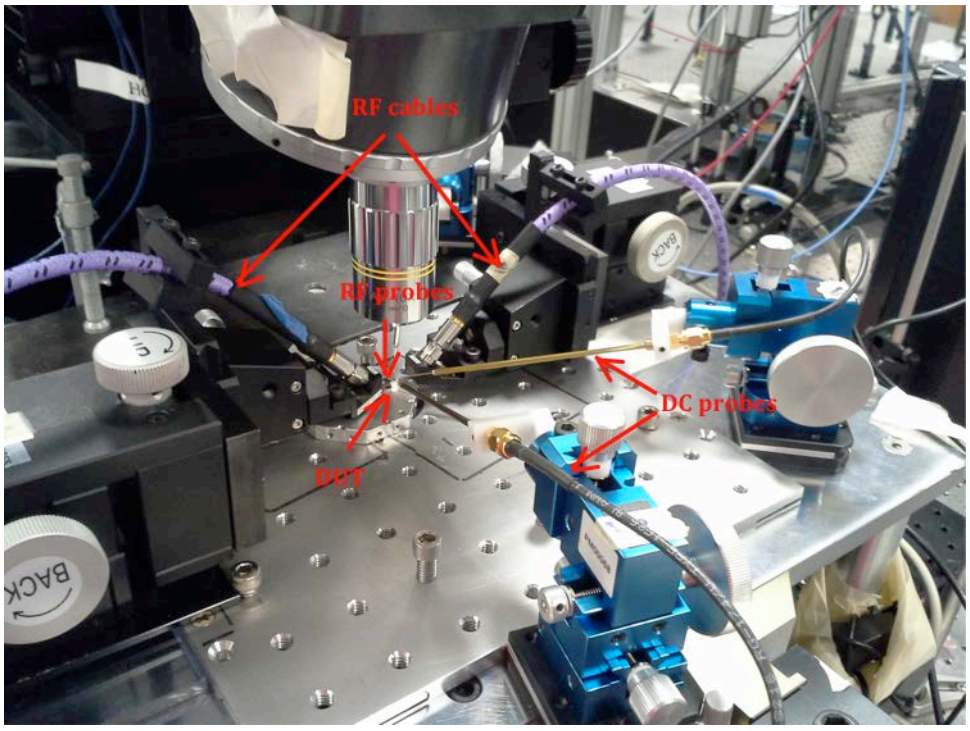


Figure 2. Image of the RF probe system.

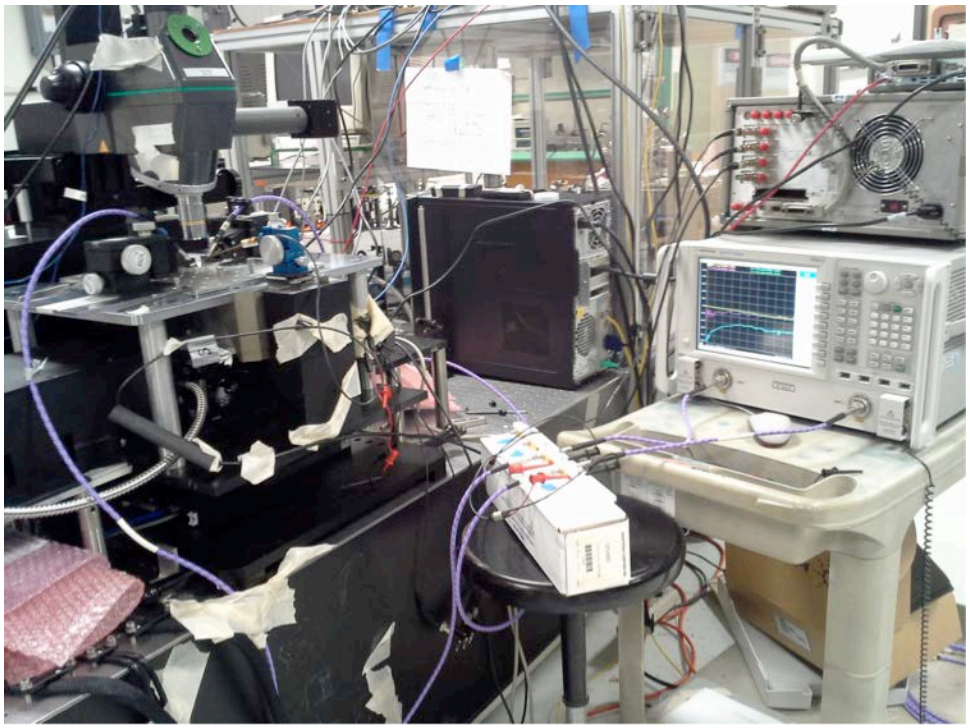


Figure 3. Image of the RF test system.

3.0 Test Procedures Used

3.1 DC Measurements (ID-VG)

The DC parameters of a device were measured directly using the PA. The transconductance of the DUT was extracted from the ID-VG measurements.

3.1.1 NFET characterization:

ID-VG:

1) Apply 0 V to source terminals.
2) Apply first drain bias (0-V_{DD}).
3) Sweep gate bias from -0.3 V to V_{DD}. Record drain and gate currents at each bias condition.
4) Repeat 2-3 for next drain bias.

3.1.2 PFET characterization:

ID-VG:

1) Apply V_{DD} to source terminals.

2) Apply first drain bias (V_{DD}-0)
3) Sweep gate bias from V_{DD} to +0.3 V. Record drain and gate currents at each bias condition.
4) Repeat 2-3 for next drain bias.

3.1.3 DIODE, CAPACITOR, VARACTOR characterization:

1) Apply 0 V to the B terminal.
2) Sweep A terminal bias from 0 V to V_{DD} in 100 mV increments. Record terminal A current at each bias condition.
3) Repeat 1-2 for next bias.

3.2 S-parameters Measurements

The S-parameters of a device were measured directly using the PNA. Since there are no matching networks at the input and output terminals, parasitic de-embedding needed to be performed.

3.2.1 De-embedding parasitics using calibration structures

Calibration needed to be performed at three different stages in the test-setup. Once the calibration was done at each stage, the measured data was corrected to account for the parasitics. Thus, the RF reference plane was moved to that corresponding point. The objective was to move the reference plane to the input/output pads of the test structures so that we obtain the true S-parameters of the device. The following calibration was done in order to move the reference plane to the input/output pads of the device.

1) Calibration at the connectors of PNA: PNA was left unconnected and the calibration was done automatically by the PNA using the built-in ECal function. The measured data was automatically corrected by the PNA to take into account the parasitics of the port 1 and port 2 connectors.
2) Calibration at the RF probe tip: This was done to obtain the effects of parasitics up to the probe tips (from high-speed RF cables, bias-T and RF probe). This was accomplished by measuring S-parameters of the calibration structures (OPEN, SHORT, LOAD, THROUGH) available on the calibration substrate (CS-1513). This step moves the reference plane to the RF probes.
3) Calibration at input/output pads on the DUT: Calibration structures (OPEN, SHORT, L-LOAD, R-LOAD, THROUGH) have been included on the die. The OPEN-SHORT de-embedding procedure [1] is sufficient to 40 GHz (limitation of test equipment) and was verified accurate on a similar 45 nm RF SOI CMOS test chip fabricated (TAPO ID: P12257A, VU #: 45-VURF-1). The measurement from these

structures enabled us to de-embed the data from effects of the pad/interconnect parasitics on the die. This moves the RF reference plane to the die.

4) Calibration of the parasitic components of the DC probe used to contact the source terminal of the transistors. Gain degradation due to power loss in the parasitic elements connected to the transistor source was corrected using the following procedure.
   - Measure the impedance of the source contact and DC probe by loading the on-chip THROUGH calibration structure with the DC probe and measuring the s-parameters with the PNA. The following expression was used to determine the corrected S_{21} parameter:

     $$S_{21} = S_{21\text{-measured}} + [g_m \times |Z_S|]_{dB}$$

     where $S_{21\text{-measured}}$ is the measured parameter following the third calibration stage, $|Z_S|$ is measured impedance at the source of the MOSFET, and g_m is the measured trans-conductance of the MOSFET

5) The characteristic impedance of the coplanar waveguide was determined by measuring the s-parameters of the on-chip *Through* calibration structure loaded by known calibration substrate and load impedance. The s-parameters were converted to z-parameters in order to extract the characteristic impedance (Z_O) of the waveguide (CPW). The characteristic impedance of the 50 Ω coplanar waveguide was measured at frequencies up to 50 GHz. the waveguide characteristic impedance was on target at 50 Ohms through 40 GHz, with a Phase angle at 0 degrees through about 30 GHz, indicating the impedance to be purely resistive as seen in ***Figure 4***. It can be noted that the deviation from ideal at the higher frequencies may be caused by measurement noise near the frequency limits of the cabling and equipment (see ***Table 3***).

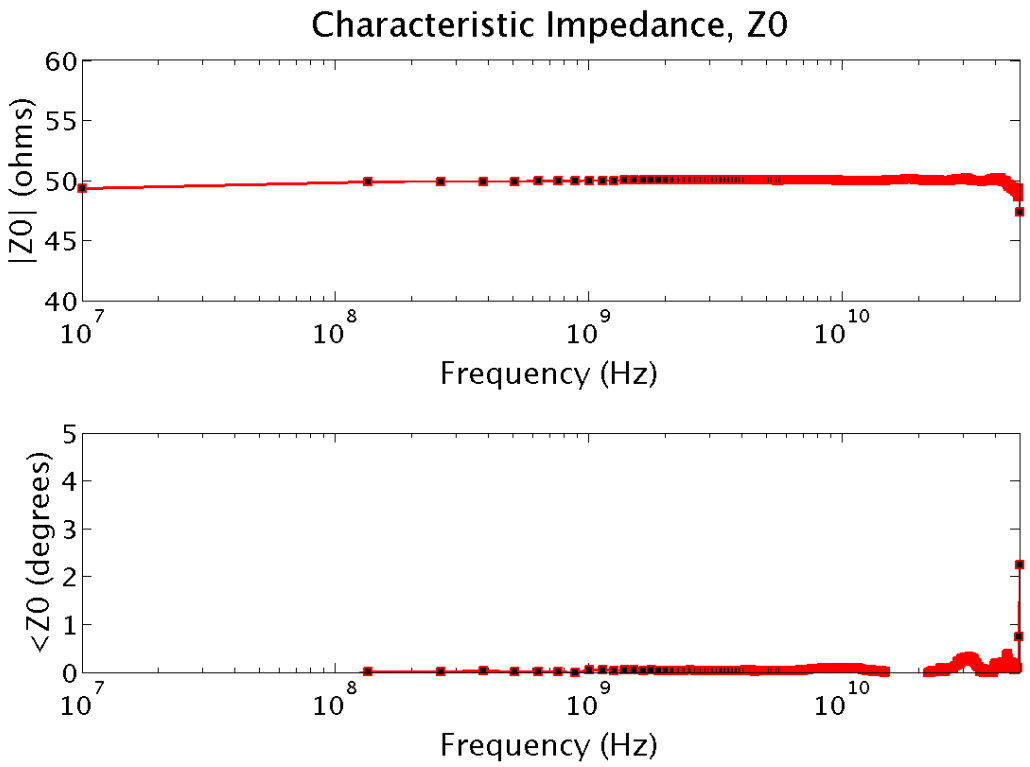


Figure 4 shows the waveguide characteristic impedance matching the target of 50 Ohms and is purely resistive.

[1] H. Cho et. Al., "A Three-Step Method for the De-Embedding of High-Frequency S-Parameter Measurements," *IEEE Transaction on Electron Devices,* Vol. 38, No. 6, Jun 1991.

3.2.2 Test Flow For S-parameter Measurement

1) First, the S-parameters of the on-chip calibration structures (OPEN, SHORT, L-LOAD, R-LOAD, THROUGH) were measured for de-embedding purposes.
2) The required device were probed using high-speed probes/DC probes.
   (For MOSFETs: gate pad connected to port 1 and drain pad connected to port 2 of the PNA, source and substrate are grounded through a 50 Ω terminator using DC probes and PA)
3) DC performance (Id-Vg) were measured on PA to check the functionality of device.
4) Necessary bias was applied to device.
5) S-parameters were measure using PNA.
6) Again, DC performance (Id-Vg) measure was measured on PA to check the functionality of device.
7) Steps 4-6 were repeated for each bias condition.

8) Steps 2-7 were repeated with different devices on DUT.

Once the S-parameters of the device were measured, the de-embedding procedure (3.1.1) was used to determine the true device S-parameters.

3.2.3 Validation of De-embedding Procedure

The de-embedding of parasitics up to probe tips was done using the automatic calibration function available in the PNA (section 3.2.1). The manual de-embedding (using MATLAB) was carried out for the same test structures in the CS1513 calibration substrate and the results validated with the results of PNA.

3.3 f_T Measurement

The de-embedded S-parameters were converted to H-parameters in MATLAB. The frequency at which the H21 parameter equals 0 db (obtained by extrapolating the h_{21} vs frequency curve) is defined as the f_T of a device. f_T was measured for gate bias between 0 to 1 V with 100 mV steps and drain bias of 1 V.

4.0 DC Measurement Results

DC measurements were taken for the following devices: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, AA, BB, CC, DD, EE, FF, GG, HH, II corresponding to structure ID defined in ***Table 1***. Data plot examples are presented in Figure 3 for device type E: 12 µm floating body minimum channel length AVT NFET with 15 fingers, showing Id-Vd, Id-Vg, g_m-Vg on a linear scale and Id/Ig-Vg on a logarithmic scale. The device is weaker than nominal model prediction but is within the expected process range (-3σ). ***Figure 6*** shows the simulated typical, +3σ and -3σ process corners (model version V1.2.0.1) compared to the measured data.

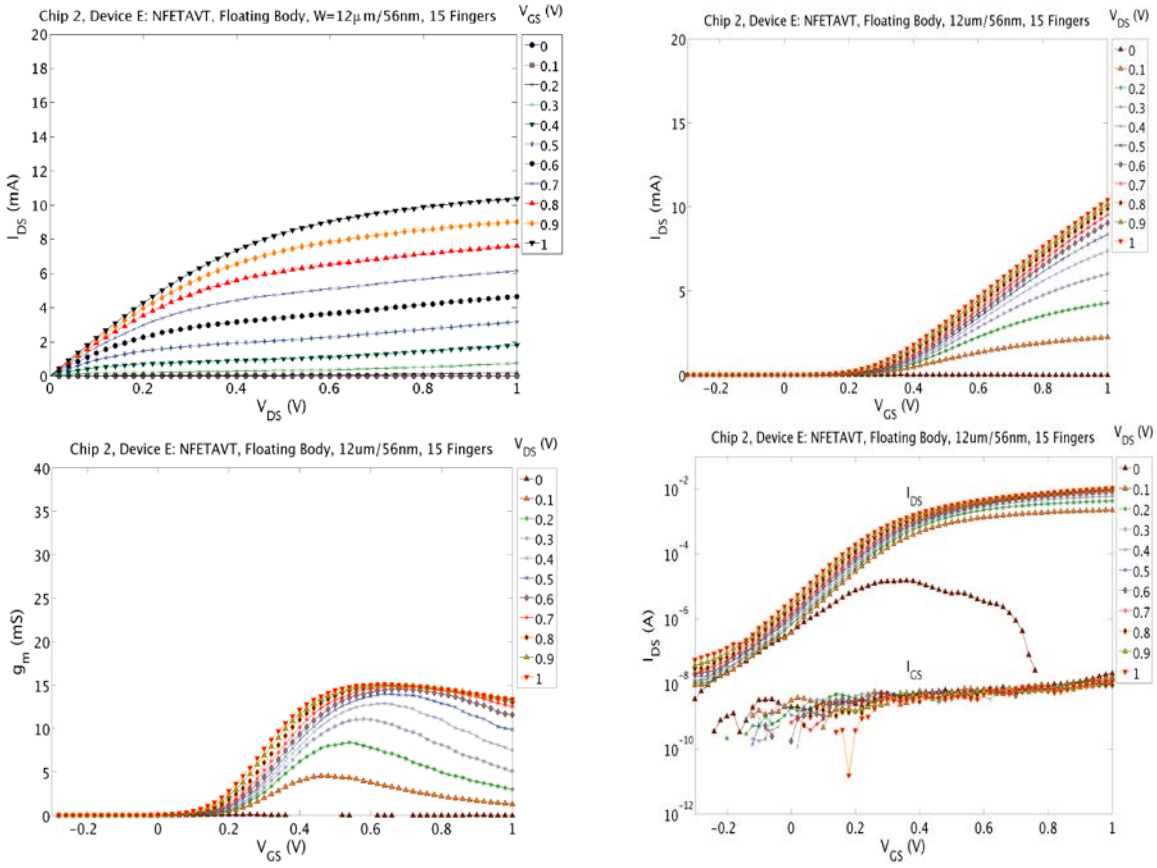
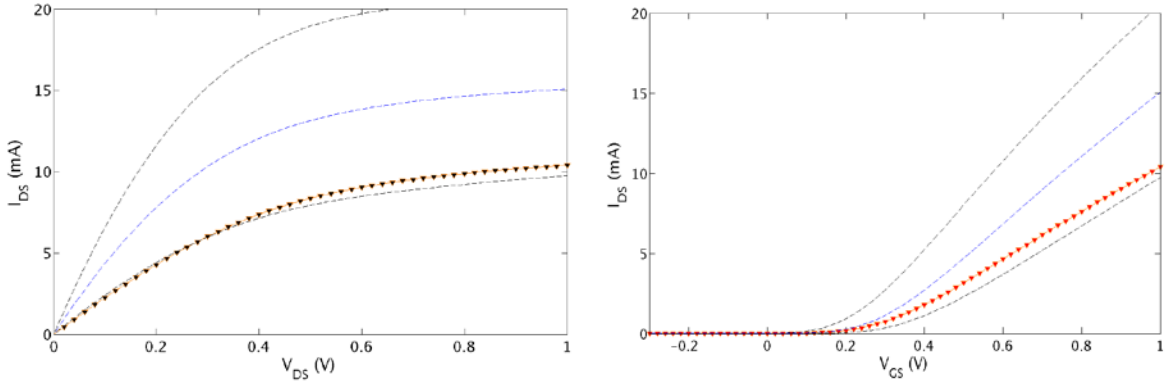


Figure 5 showing plots of relevant device data for Type E device.



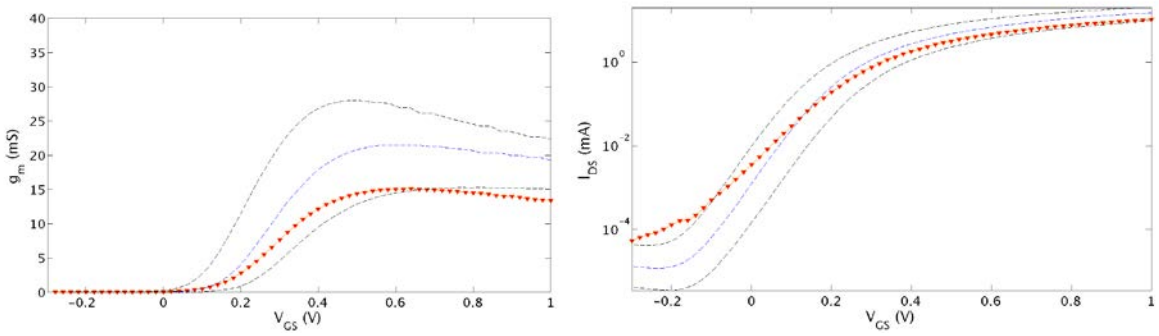


Figure 6 showing comparison of measured Type E device characteristics to simulated typical (blue dotted line), +3σ (upper black dotted lines) and -3σ process corners (lower black dotted lines) with model version V1.2.0.1.V_{DS} and V_{GS} are set to 1.0 V for I_{DS}-V_{GS} and I_{DS}-V_{DS} measurements, respectively.

Figure 7 presents plots for device type Y, 12 µm floating body minimum channel length AVT PFET. ***Figure 8*** shows the simulated typical, +3σ and -3σ process corners (model version V1.2.0.1) compared to the measured data. Again, the PFET is also on the week end of the distribution.

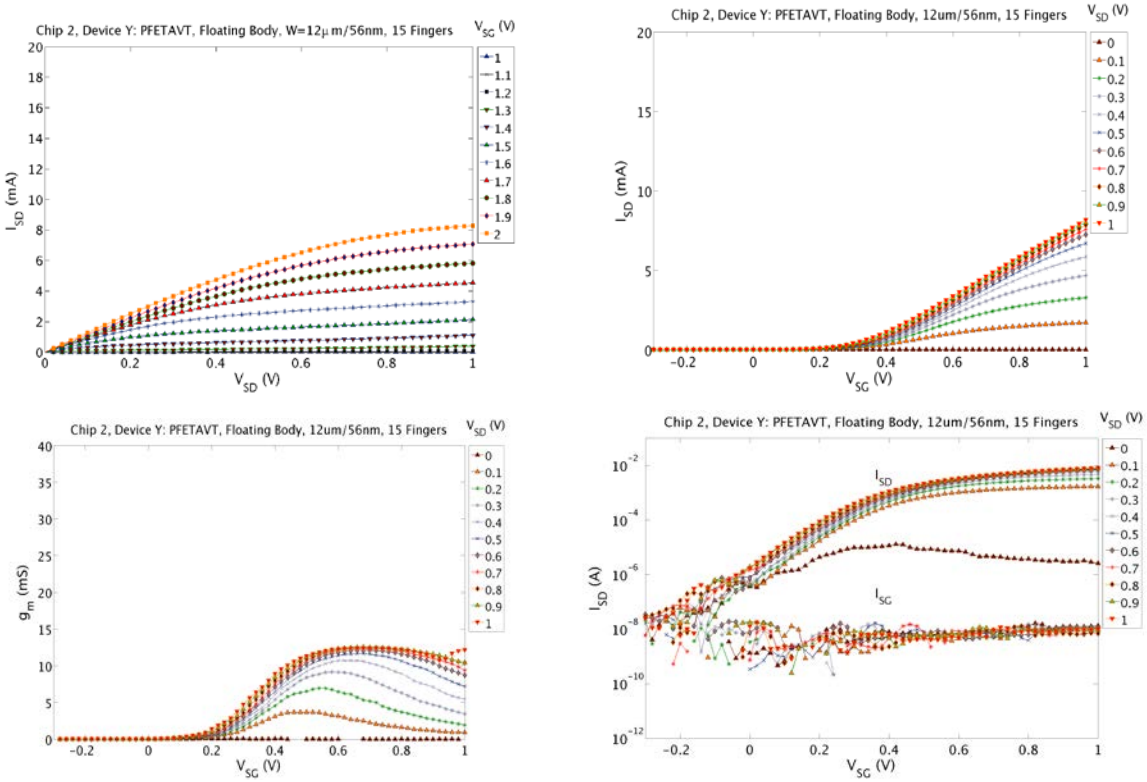


Figure 7 showing plots of relevant device data for Type Y device.

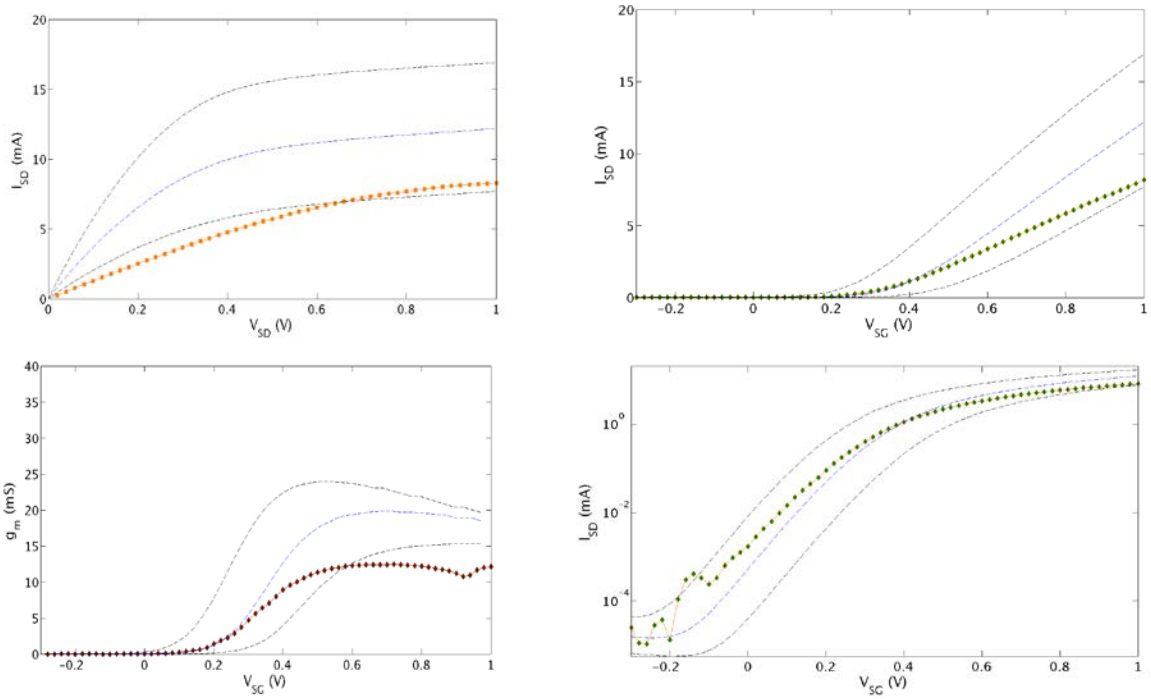


Figure 8 showing comparison measured Type Y device characteristics to simulated typical (blue dotted line), +3σ (upper black dotted lines) and -3σ process corners (lower black dotted lines) with model version V1.2.0.1.V_{SD} and V_{SG} are set to 1.0 V for I_{SD}-V_{SG} and I_{SD}-V_{SD} measurements, respectively.

In order to put the large amount of data into perspective, a comparison was performed between the various devices to estimate the effect of the following on Idsat, Ioff and Ig:

- Body doping (device threshold)
- Device geometry (number of fingers)
- Body contact (floating vs. tied)
- Device channel length
- NFET vs. PFET

In the following figure quads, the first plot is in log-linear scale to put all the values in perspective, while the remaining plots are on linear-linear scale to emphasize the difference between values.

4.1 Body Doping (RVT/SVT/UVT)

Comparison between the RVT (G), SVT (H) and UVT (I) device parameters is presented in ***Figure 9.*** As one would expect, the drive current drops with increased Vt (25% and 50% correspondingly), while device leakage (Ioff) drops by over an order of magnitude (15x and

30x). There is no consistent trend in gate leakage and the values were insignificant compared to Ioff.

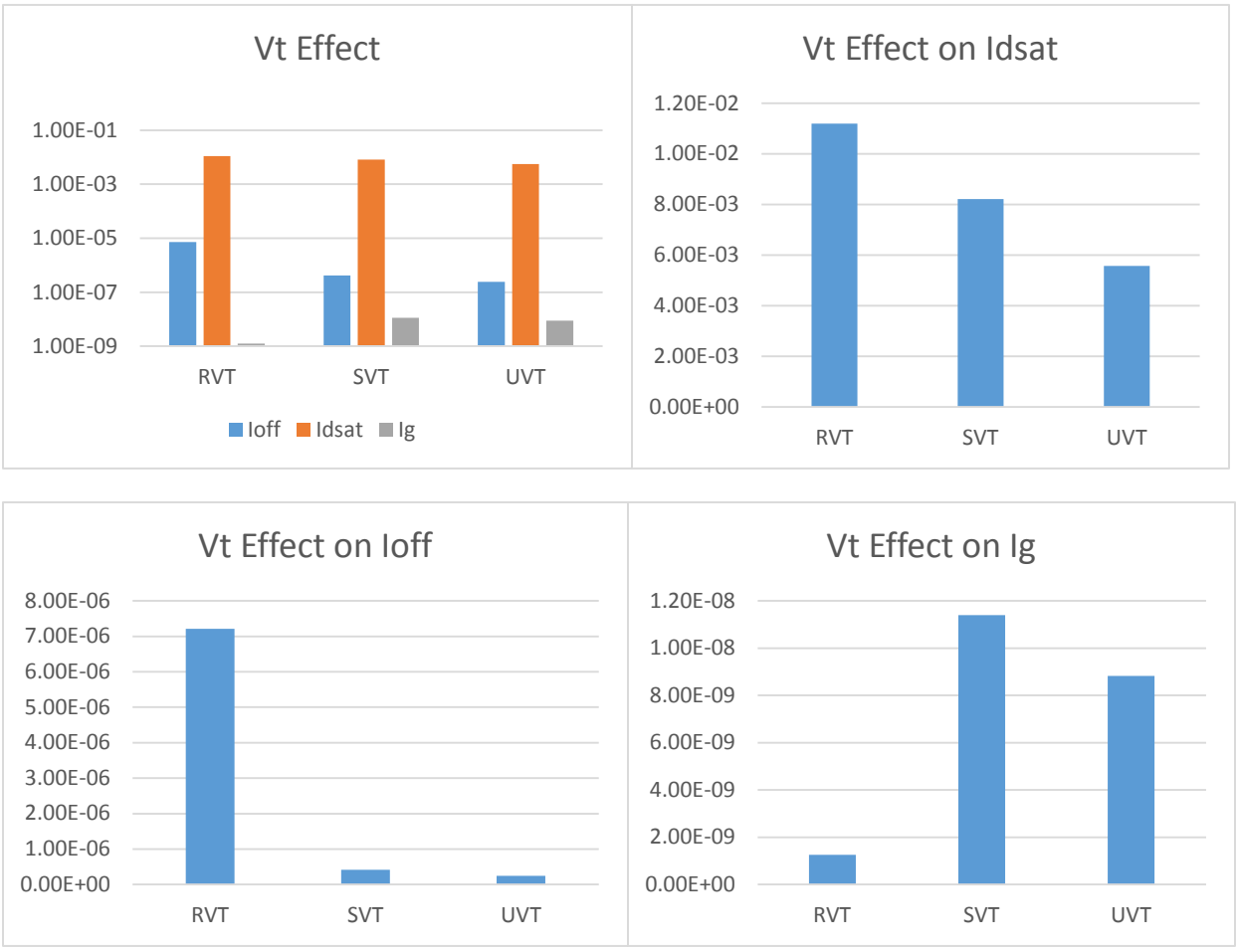


Figure 9. Idsat, Ioff and Ig (in units of A) for devices of varied threshold voltage (Vt).

Drain-Induced Barrier Lowering (DIBL is a significant effect in deep sub-micrometer devices. This is due to the influence of the drain potential on the depletion region under the gate. DIBL lowers the potential necessary to turn the transistor on, hence it lowers the threshold voltage of the device as drain potential increases. ***Figure 10*** shows the Id-Vg curves for the RVT, SVT and UVT devices at low (Vdd=0.1V) and High (Vdd=1.0V) drain bias. The reduction of threshold with drain bias is ≈150mV and is almost equivalent for all three device types (≈130mV for RVT, ≈140mV for SVT and ≈170mV for UVT).

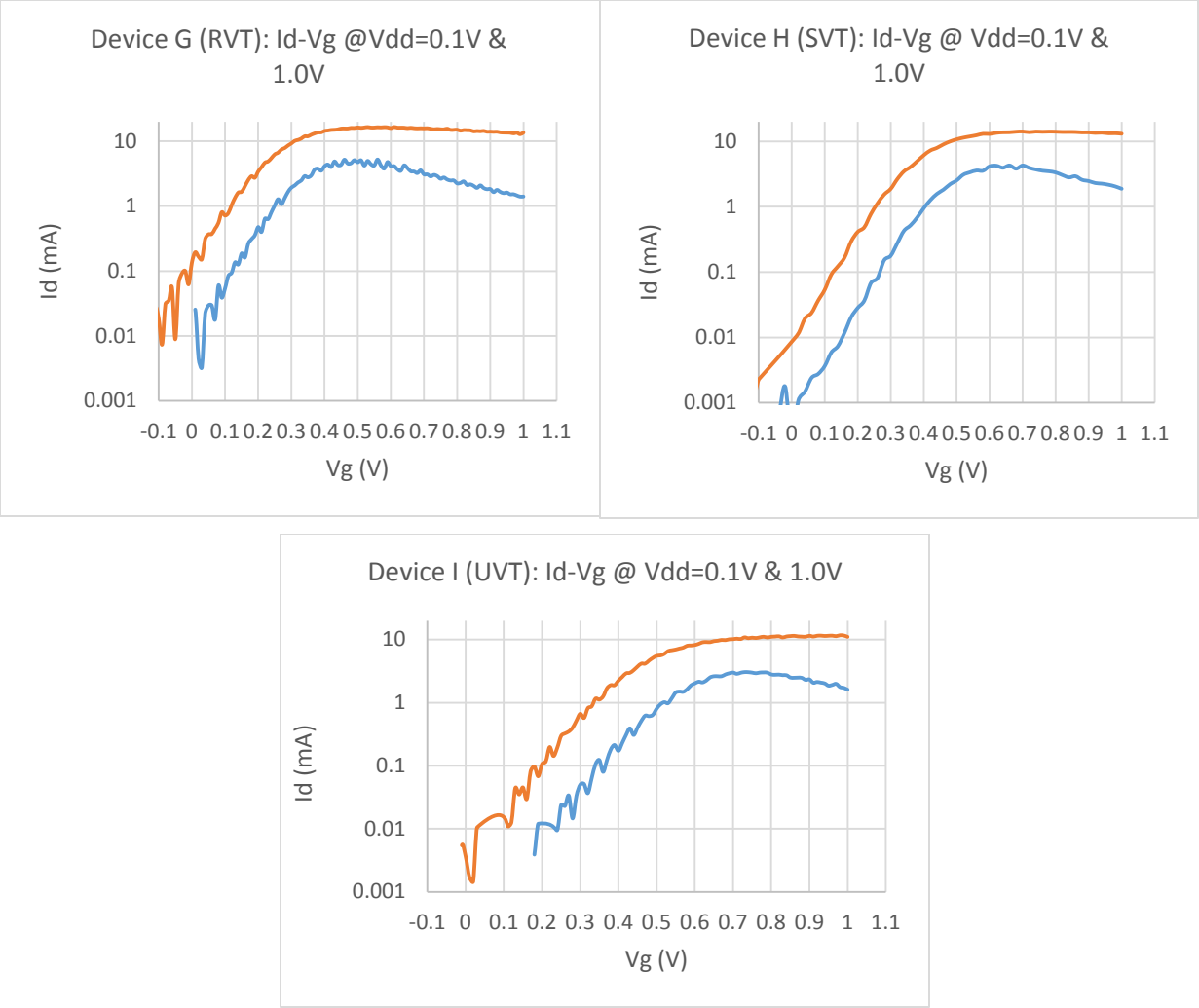


Figure 10. Id-Vg curves at high and low drain bias for RVT, SVT and UVT devices.

4.2 Device Geometry

A comparison between devices A, B and C with equal device total widths but varying number of fingers shows some indication of ΔW effect that limits device drive (by 20%) with increased number of fingers (2X) as seen in ***Figure 11*****.** Gate leakage seams to go down with the number of figures, but again it is insignificant compared to Ioff.

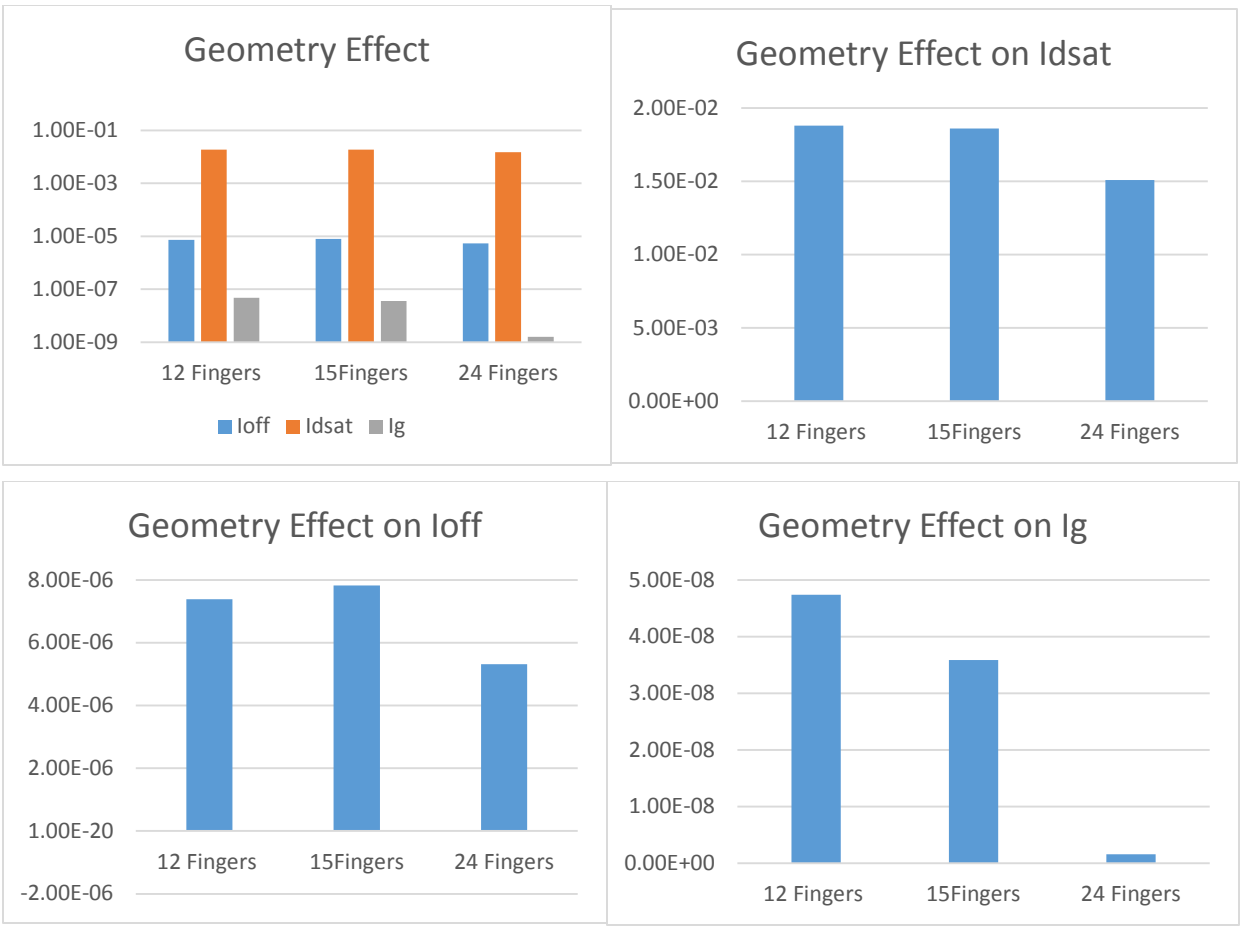


Figure 11. Idsat, Ioff and Ig (in units of A) for devices of varied geometry (number of gate fingers)

4.3 Body Tie Effect

A floating body in a partially depleted SOI technology is expected to increase the device drive current at the expense of a significant increase in device leakage. ***Figure 12*** shows a comparison between device A (floating body) and a corresponding device Z (body tie). The body tie cuts the device drive current in half and reduces the device leakage to one quarter the floating device values. Again, the gate leakage is insignificant compared to Ioff.

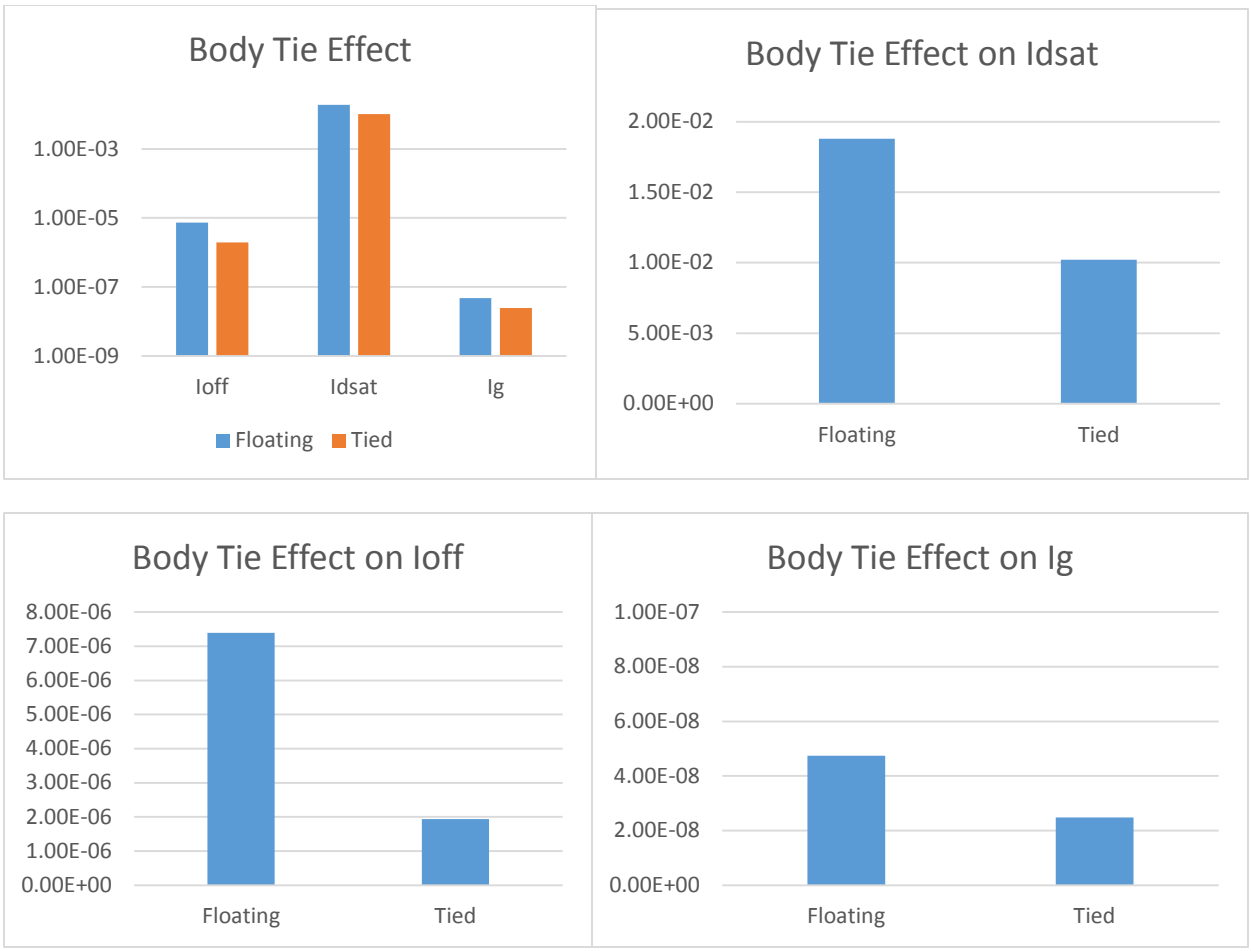


Figure 2. Idsat, Ioff and Ig (in units of A) for devices with and without body ties.

4.4 Device Channel Length

The length effect is illustrated by comparing tied devices M (56nm), N (100nm) and O (250nm) as shown in ***Figure 13***. The measurements are consistent with expectation in that Idsat goes down with channel length roughly in proportion to 1/L. Ioff had a much stronger relationship approximating $1/L^2$, while gate leakage was roughly proportional to gate area, or L.

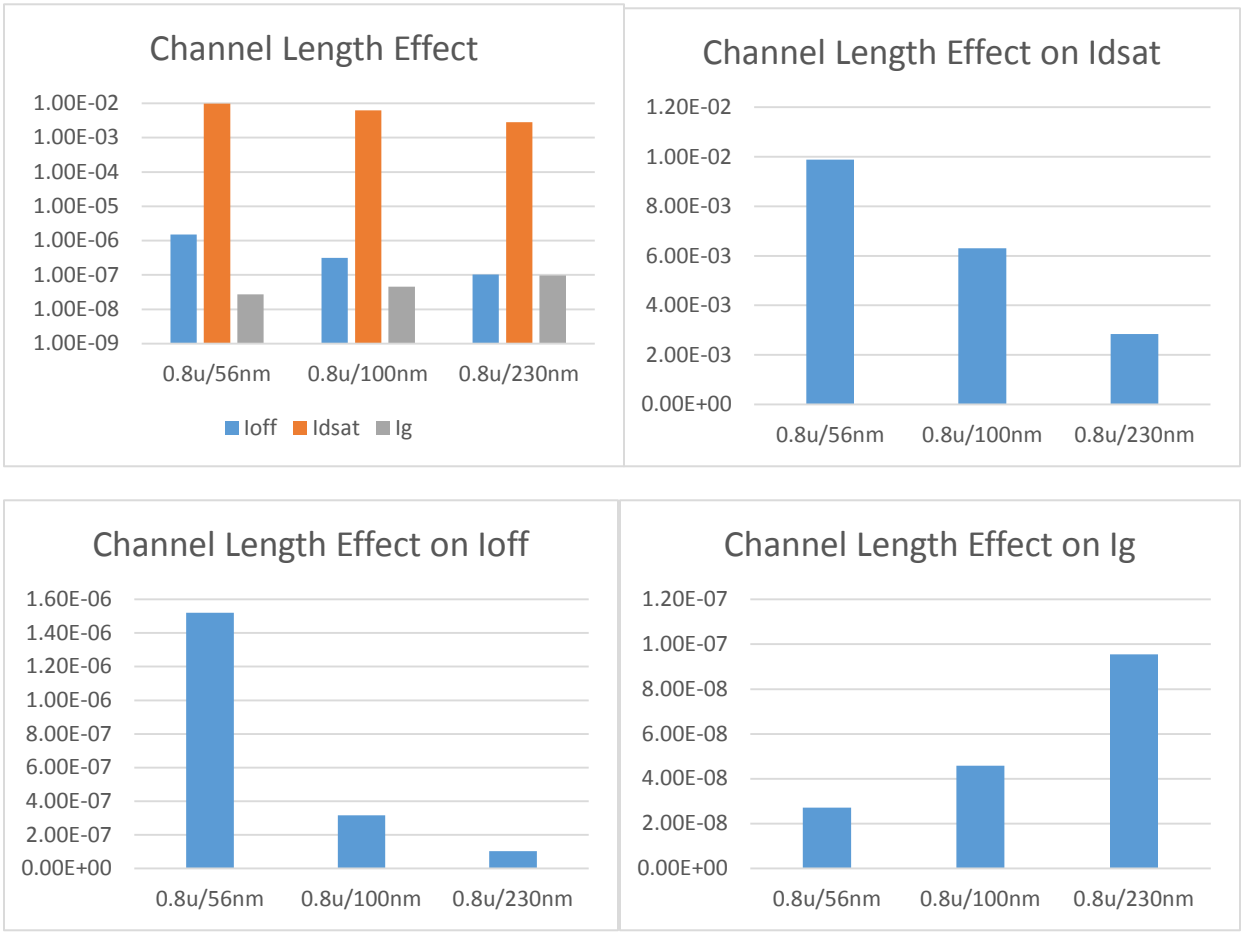


Figure 3. Idsat, Ioff and Ig (in units of A) for devices of varied channel length.

4.5 NFET vs. PFET

Figure 14 shows comparisons between same geometry NFETs and PFETs for two different geometries (24um/24 Fingers and 12um/15 Fingers). The PFET drive current is lower by 1/3 to 1/5 compared to the NFET, with a corresponding reduction in leakage. This indicates improved hole mobility achieved by stress engineering. Again, gate leakage is insignificant in comparison to Ioff.

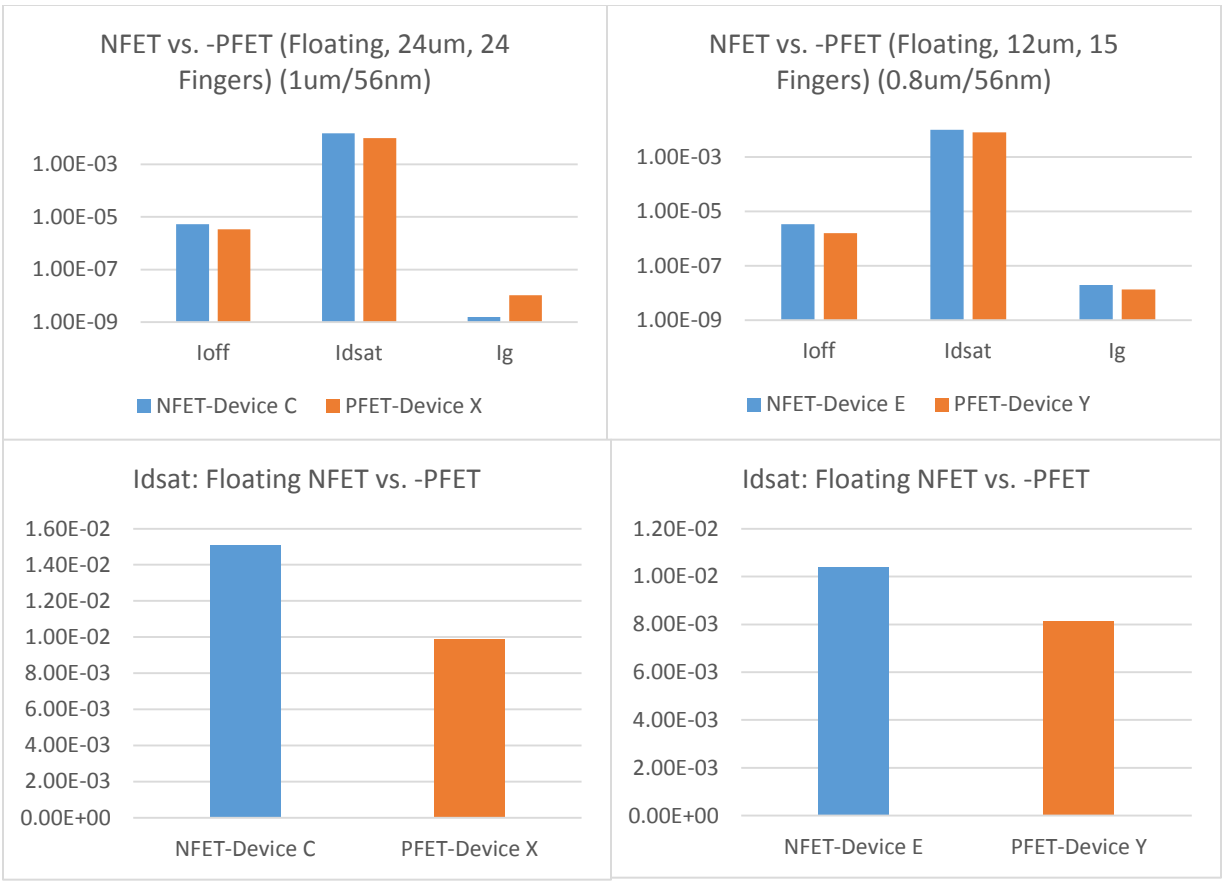


Figure 4. Idsat, Ioff and Ig (in units of A) for floating body NFET and PFET devices.

5.0 RF Data and Extraction

5.1. S-Parameters

The procedure to measure and de-embed the parasitics from the S-parameters is described above in Sections 2 and 3. The following figures illustrate examples of measured and de-embedded S_{21} (forward voltage gain) versus frequency for comparable devices E, Y, C, and X. In ***Figure 15*** (Device E), the significant roll off in the S_{21} curves at high frequency can be attributed to high-speed package parasitics that are present in Chip 6 due to the packaging of the device, rather than the probing of the device. The three other devices in ***Figures 16, 17,*** and ***18*** were measured used the probing technique described in Section 2.

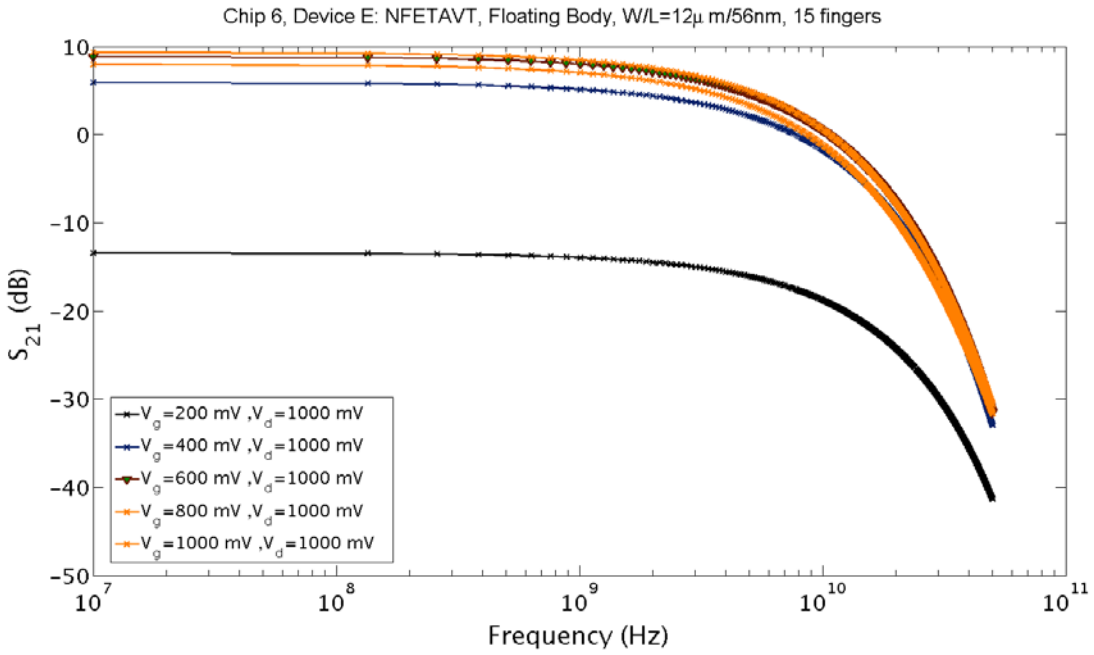


Figure 5 shows S-parameter S_{21} vs. Frequency for Device E.

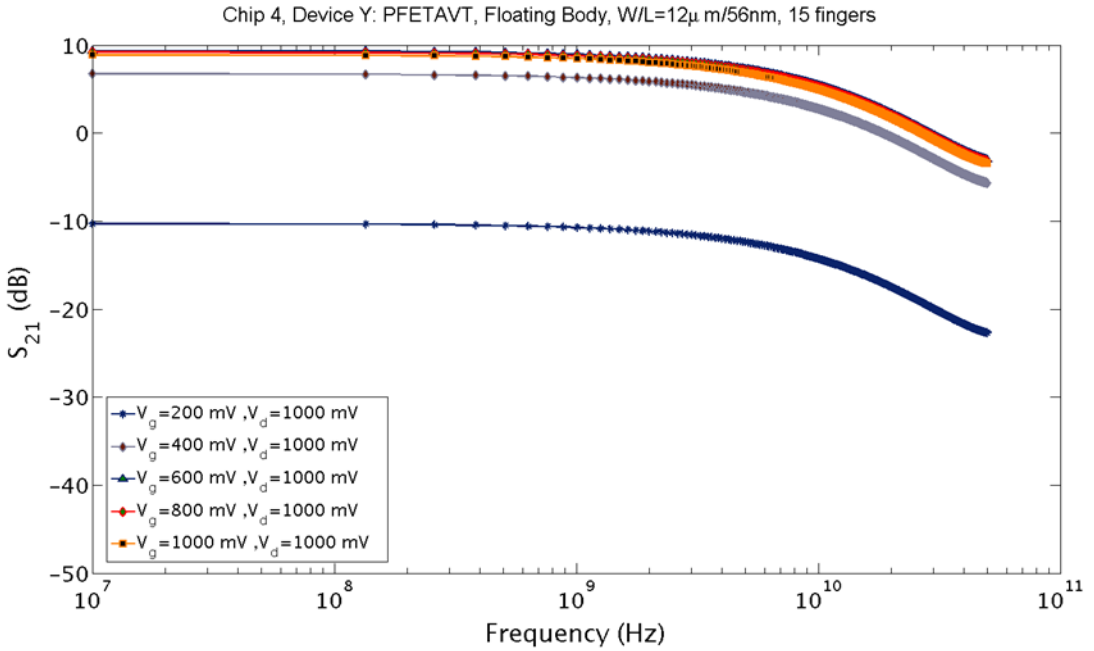


Figure 6 shows S-parameter S_{21} vs. Frequency for Device Y.

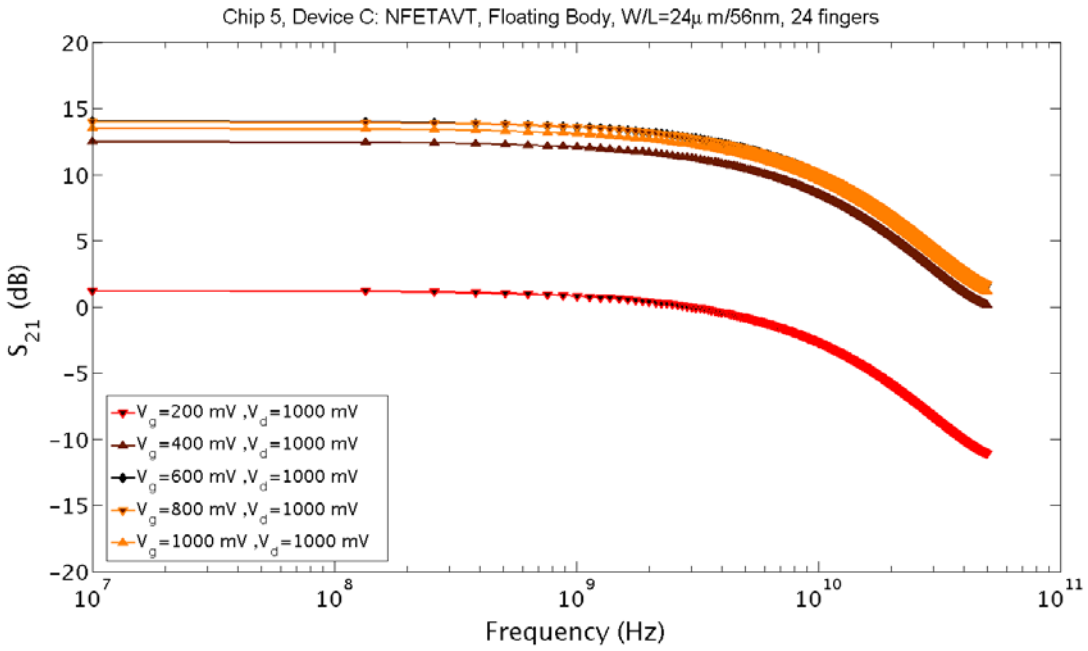


Figure 7 shows S-parameter S_{21} vs. Frequency for Device C.

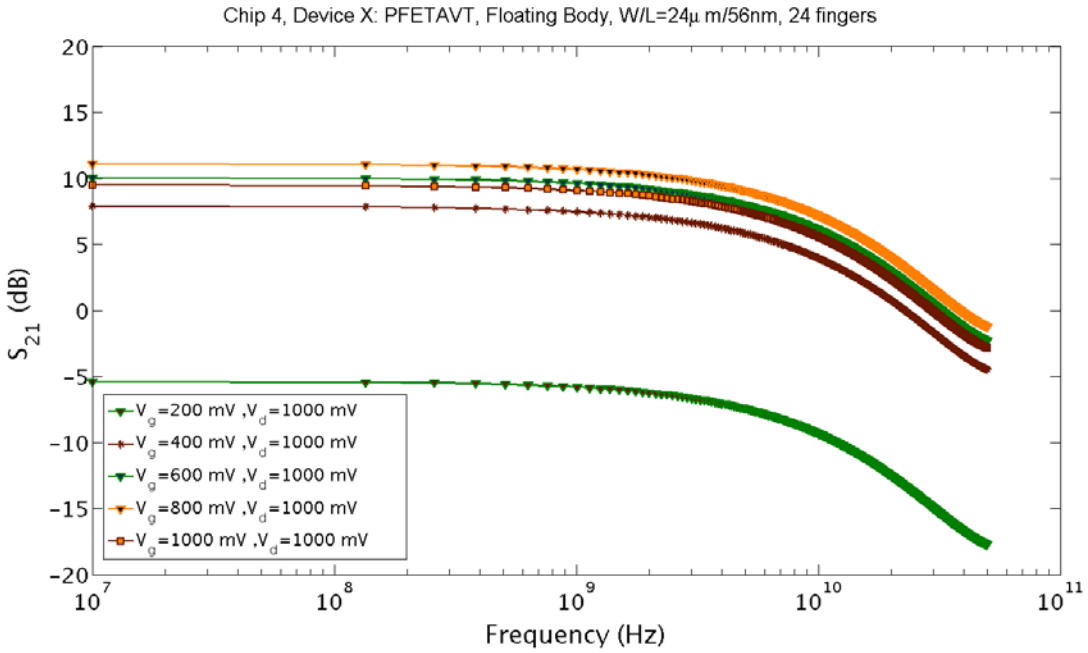


Figure 8 shows S-parameter S_{21} vs. Frequency for Device X.

Table 4 shows a compilation of the peak forward voltage gain (S_{21}) for each device. There are at least 1-3 samples measured for each device.

Table 4. Device Peak S21 S-parameters

ID	Device	Peak(S_{21}) [dB]	Peak(S_{21}) [dB]	Peak(S_{21}) [dB]	Average Peak(S21) [dB]	Peak(S_{21}) Standard Deviation
A	NFET, AVTFB, W = 2 µm x 12 (24 µm), L = 56 nm	16	16.5	16.4	**16.3**	**0.25**
B	NFET, AVTFB, W = 1.6 µm x 15 (24 µm), L = 56 nm	16.3	-	-	**16.3**	**0.00**
C	NFET, AVTFB, W = 1 µm x 24 (24 µm), L = 56 nm	16.3	-	-	**16.3**	**0.00**
D	NFET, AVTFB, W = 1 µm x 15 (15 µm), L = 56 nm	13.3	13.2	-	**13.3**	**0.05**
E	NFET, AVTFB, W = 0.8 µm x 15 (12 µm), L = 56 nm	9.6	9.3	-	**9.5**	**0.15**
F	PFET, AVTFB, W = 2 µm x 12 (24 µm), L = 56 nm	15.5	15.9	15.4	**15.6**	**0.20**
G	NFET, RVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	12.6	-	-	**12.6**	**0.00**
H	NFET, SVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	11.5	10.7	10.2	**10.8**	**0.40**
I	NFET, UVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	11.2	11.3	-	**11.3**	**0.05**
J	NFET, TOFB, W = 0.8 µm x 15 (12 µm), L = 100 nm	9.5	9.9	-	**9.7**	**0.20**
K	NFET, AVTBC, W = 2 µm x 6 (12 µm), L = 56 nm	12.3	11.8	-	**12.1**	**0.25**
L	NFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	12.2	-	-	**12.2**	**0.00**
M	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 56 nm	11.9	-	-	**11.9**	**0.00**
N	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 100 nm	9.7	-	-	**9.7**	**0.00**
O	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 230 nm	6	-	-	**6.0**	**0.00**
P	NFET, AVTBC, W = 1 µm x 15 (15 µm), L = 230 nm	10.2	-	-	**10.2**	**0.00**
U	PFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 230 nm	1.5	1.2	-	**1.4**	**0.15**
V	PFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 100 nm	6.9	6.6	-	**6.8**	**0.15**
W	PFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	3.5	6.4	-	**5.0**	**1.45**
X	PFET, AVTFB, W = 1 µm x 24 (24 µm), L = 56 nm	15.3	-	-	**15.3**	**0.00**
Y	PFET, AVTFB, W = 0.8 µm x 15 (12 µm), L = 56 nm	10.8	-	-	**10.8**	**0.00**
Z	NFET, AVTBC, W = 2 µm x 12 (24 µm), L = 56 nm	10.9	-	-	**10.9**	**0.00**
AA	NFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	-	-	-	**-**	**-**

5.2. f_T Extraction

f_T extractions were made on the various devices listed in **Table 1**. Maximum f_T values for the various devices are presented in ***Table 5*** at Vdd=1.0V and the Vg at which value the maximum f_T was obtained. It can be seen from Vg column that the maximum f_T occurs at Vg value that varies between 100mV and 1000mV depending on the device.

Table 5. Maximum f_T Values for the Various Devices and Corresponding Applied Voltages

Die#	Device	Vg (mV)	Vd (mV)	fT(gm/Cg)|HIGH (GHz)
5	A	700	1000	168.7
4	B	500	1000	177.2
5	C	600	1000	210.4
4	F	400	1000	133.1
4	X	400	1000	159.1
4	Y	400	1000	85.9
5	D	700	1000	161.3
5	E	700	1000	101
4	G	500	1000	120.6
4	H	700	1000	97.3
4	I	800	1000	101.4
4	J	900	1000	100.9
4	K	600	1000	98.5
5	L	700	1000	141.6
5	M	700	1000	122.4
4	N	800	1000	81.2
4	O	1000	1000	49
4	P	900	1000	83.8
5	U	100	1000	35.5
5	V	200	1000	71.2
5	W	300	1000	69.2
4	Z	700	1000	111.1
6	E	800	1000	72.9

The various extracted peak f_T values, averages and standard deviations are listed in **Table 6** below.

***Table 6*. Device Peak f_T Parameters**

ID	Device	Peak(f_T) [GHz]	Peak(f_T) [GHz]	Peak(f_T) [GHz]	Average Peak(f_T) [GHz]	Peak(f_T) Standard Deviation
A	NFET, AVTFB, W = 2 µm x 12 (24 µm), L = 56 nm	162.9	190.3	185.9	**179.7**	**12.01**
B	NFET, AVTFB, W = 1.6 µm x 15 (24 µm), L = 56 nm	159.6	-	-	**159.6**	**0.00**
C	NFET, AVTFB, W = 1 µm x 24 (24 µm), L = 56 nm	163.9	-	-	**163.9**	**0.00**
D	NFET, AVTFB, W = 1 µm x 15 (15 µm), L = 56 nm	102.9	91.9	-	**97.4**	**5.50**
E	NFET, AVTFB, W = 0.8 µm x 15 (12 µm), L = 56 nm	71.8	62.9	-	**67.4**	**4.45**
F	PFET, AVTFB, W = 2 µm x 12 (24 µm), L = 56 nm	135	135.4	129.4	**133.3**	**2.74**
G	NFET, RVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	90.3	-	-	**90.3**	**0.00**
H	NFET, SVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	76.4	68.5	81.0	**75.3**	**5.16**
I	NFET, UVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	101.4	99.5	-	**100.5**	**0.95**
J	NFET, TOFB, W = 0.8 µm x 15 (12 µm), L = 100 nm	78.1	76.0	-	**77.1**	**1.05**
K	NFET, AVTBC, W = 2 µm x 6 (12 µm), L = 56 nm	86.6	73.3	-	**80.0**	**6.65**
L	NFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	106.8	-	-	**106.8**	**0.00**
M	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 56 nm	94.8	-	-	**94.8**	**0.00**

N	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 100 nm	60.1	-	-	**60.1**	**0.00**
O	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 230 nm	36.2	-	-	**36.2**	**0.00**
P	NFET, AVTBC, W = 1 µm x 15 (15 µm), L = 230 nm	65.5	-	-	**65.5**	**0.00**
U	PFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 230 nm	31.1	27.5	-	**29.3**	**1.80**
V	PFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 100 nm	55.3	56.3	-	**55.8**	**0.50**
W	PFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	38.4	50.2	-	**44.3**	**5.90**
X	PFET, AVTFB, W = 1 µm x 24 (24 µm), L = 56 nm	142.6	-	-	**142.6**	**0.00**
Y	PFET, AVTFB, W = 0.8 µm x 15 (12 µm), L = 56 nm	76.8	-	-	**76.8**	**0.00**
Z	NFET, AVTBC, W = 2 µm x 12 (24 µm), L = 56 nm	84.7	-	-	**84.7**	**0.00**
AA	NFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	-	-	-	**-**	**-**

6.0 Capacitance and Inductance Measurements

Unfortunately, accurate capacitance and inductance measurements could not be obtained due to probing difficulties on the test-chip structures.

7.0 RF Parameter Analysis

7.1. The Effects of MOS Type:

The differences (Δ) in measured and de-embedded peak S_{21} and f_T values for PFET and NFET devices with identical design parameters are provided in ***Table 7*** where

$$\Delta = S_{21}|\,|f_T\,(\text{NFET}) - S_{21}|\,|f_T\,(\text{PFET})$$

In general the peak S_{21} is equivalent to 5.5 dB greater for NFET devices than PFET devices. The peak f_T values for NFET floating body devices range between 20 GHz to 55 GHz greater than the peak f_T values for PFET floating body devices. Similarly, the peak f_T values for NFET body contacted devices range between 4 GHz to 50 GHz greater than the peak f_T values for PFET body contacted devices.

Table 7. The Differences (Δ) in Measured and De-embedded Peak S_{21} and f_T Values for PFET and NFET Devices with Identical Design Parameters

EFFECTS OF MOS TYPE (Δ = NFET parameter – PFET parameter)						
Devices (NFET, PFET)	**V_{TH} and Body Type**	**W (µm)**	**N (fingers)**	**L (nm)**	**$\Delta S_{21\|peak}$ (dB)**	**$\Delta f_{T\|peak}$ (GHz)**
A, F	AVTFB	24	12	56	0.7	46.4
C, X	AVTFB	24	24	56	1.0	21.3
E, Y	AVTFB	12	15	56	-1.3	-9.4
M, W	AVTBC	12	15	56	6.9	50.5
N, V	AVTBC	12	15	100	2.9	4.3

Comparing the floating body NFET and PFET devices, C and X, shows the maximum f_T value for the PFET is 28% lower than the NFET; while the corresponding ratio for the contacted body devices E and Y is 14% lower for the PFET than the NFET. ***Figure 19*** shows two plots for the f_T vs. gate bias for the NFET and PFET devices of the same geometry. The left plot is for floating body devices, while the right plot is for contacted body devices. (Note: the PFET gate bias is negative

with respect to the source.) The floating body devices exhibit about a factor of two higher f_T than the contacted body devices for both NFET and PFET, which is slightly higher than the device width ratio of 24um/15um.

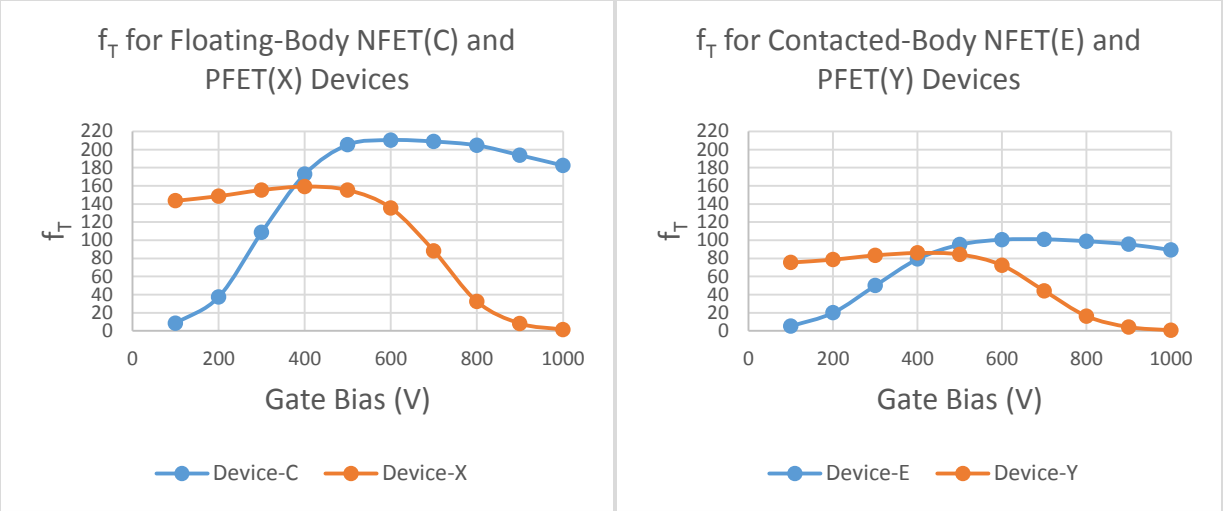


Figure 19. A comparative f_T vs. gate bias for NFET and PFET devices. The left plot is for floating body, and the right plot is for contacted body devices.

7.2. The Effects of Device Width:

The differences (Δ) in measured and de-embedded peak S_{21} and f_T values for devices with varying widths (all other parameters equal) are provided in **Table 8**, where

$$\Delta = S_{21}|\,|f_T\,(W1) - S_{21}|\,|f_T\,(W2)$$

Table 8. The Differences (Δ) in Measured and De-embedded Peak S_{21} and f_T Values for Devices with Varying Widths (All Other Design Parameters Being Equal)

EFFECTS OF DEVICE WIDTH (Δ = parameter(W1) – parameter(W2))						
Devices (W1, W2)	**MOS, V_{TH} and Body Type**	**W1, W2 (μm)**	**N (fingers)**	**L (nm)**	**$\Delta S_{21\|peak}$ (dB)**	**$\Delta f_{T\|peak}$ (GHz)**
B, E	NFET, AVTFB	24,12	15	56	5.8	92.2

As expected, S_{21} increases with increasing device width due to the increase in trans-conductance with increasing width. Also, f_T behaved similarly in spite of the increase in gate capacitance with device width.

7.3. The Effects of Number of Fingers:

The differences (Δ) in measured and de-embedded peak S_{21} and f_T values for devices with varying number of fingers (all other parameters equal) are provided in **Table 9** where

$$\Delta = S_{21}|\,|f_T\,(N1) - S_{21}|\,|f_T\,(N2)$$

Narrow width effects are anticipated to have some effect on gain as the unintentional device stress can increase with increasing fingers (assuming a constant total width), thereby decreasing the effective threshold voltage in small W devices and increasing the trans-conductance. However, S_{21} appears to be relatively independent of the number of fingers (the total gate width being constant), as S_{21} varies by less than 10% for 12, 15, and 24-fingered devices. Additionally, and unlike the 45nm technology, f_T is shown to increase with increasing number of fingers. This observed behavior may be related to the decrease in gate capacitance through the use of low-K gate dielectric.

Table 9. The Differences (Δ) in Measured and De-embedded Peak S_{21} and f_T Values for Devices with Varying Number of Fingers (All Other Design Parameters Being Equal)

EFFECT OF NUMBER OF FINGERS (Δ = parameter(N1) – parameter(N2))						
Devices (N1, N2)	**MOS, V_{TH} and Body Type**	**W (μm)**	**N1, N2 (fingers)**	**L (nm)**	**$\Delta S_{21\|peak}$ (dB)**	**$\Delta f_{T\|peak}$ (GHz)**
C, A	NFET, AVTFB	24	24, 12	56	1.6	15.8
C, B	NFET, AVTFB	24	24, 15	56	0.0	4.3
M, K	NFET, AVTBC	12	15, 6	56	-0.2	14.8
X, F	PFET, AVTFB	24	24, 12	56	-0.3	9.3

The geometry effect is illustrated in ***Figure 20***, that shows the comparison for the curves for Devices A, B, and C, that have the same effective width but varied number of fingers. It is worth noting that Device C, with 24 fingers had a slightly higher f_T even though it exhibited somewhat lower drive current as seen in ***Figure 10***.

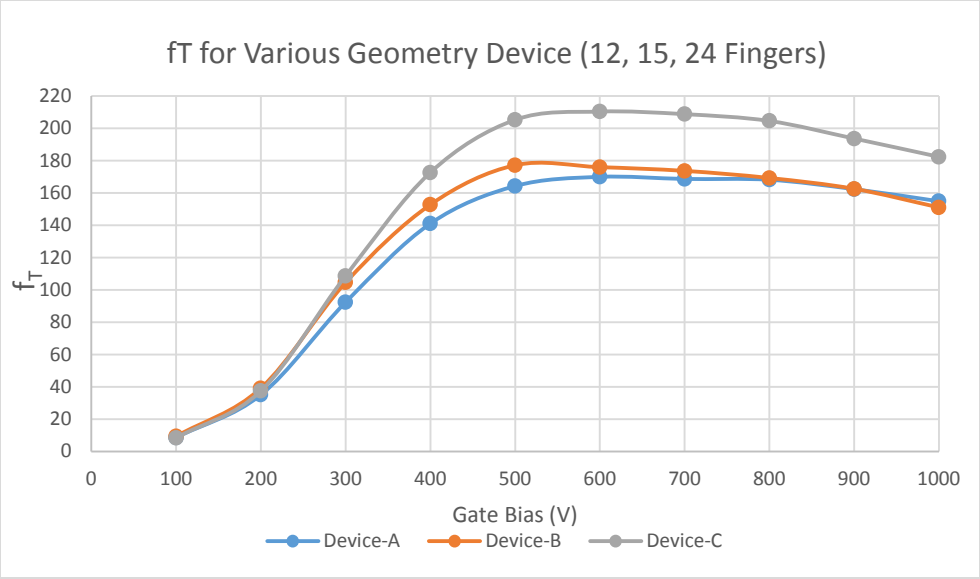


Figure 20. The effect on f_T for devices with the same 24um total width but varying number of fingers.

7.4. The Effects of Gate Length:

The differences (Δ) in measured and de-embedded peak S_{21} and f_T values for devices with varying gate lengths (all other parameters equal) are provided in Table 10 where

$$\Delta = S_{21}|\,|f_T\,(L1) - S_{21}|\,|f_T\,(L2)$$

Table 10. The Differences (Δ) in Measured and De-embedded Peak S_{21} and f_T Values for Devices with Varying Gate Lengths (All Other Design Parameters Being Equal)

EFFECT OF GATE LENGTH (Δ = parameter(L1) – parameter(L2))						
Devices (L1, L2)	**MOS, V_{TH} and Body Type**	**W (μm)**	**N (fingers)**	**L1, L2 (nm)**	**$\Delta S_{21\|peak}$ (dB)**	**$\Delta f_{T\|peak}$ (GHz)**
N, M	NFET, AVTBC	12	15	100, 56	-1.2	-34.7
O, N	NFET, AVTBC	12	15	230, 100	-3.7	-23.9
U, V	PFET, AVTBC	12	15	230, 100	-5.4	-26.5

Increasing gate length has the same effect on peak S_{21} as decreasing the gate width. That is, S_{21} decreases with increasing gate length (or decreasing gate width) due to decreasing trans-conductance with increasing length (or decreasing gate width). f_T appears to increase with decreasing gate length, due to the increase in trans-conductance and decrease in capacitance with decreasing length.

Comparing he values for devices M, N and O, representing increasing channel lengths, shows the maximum values to decrease by approximately 35% and 60% correspondingly, and the gate bias required to achieve the maximum values increasing from 700mV to 800mV and 1000mV. ***Figure 21*** presents the three curves for f_T vs. gate bias.

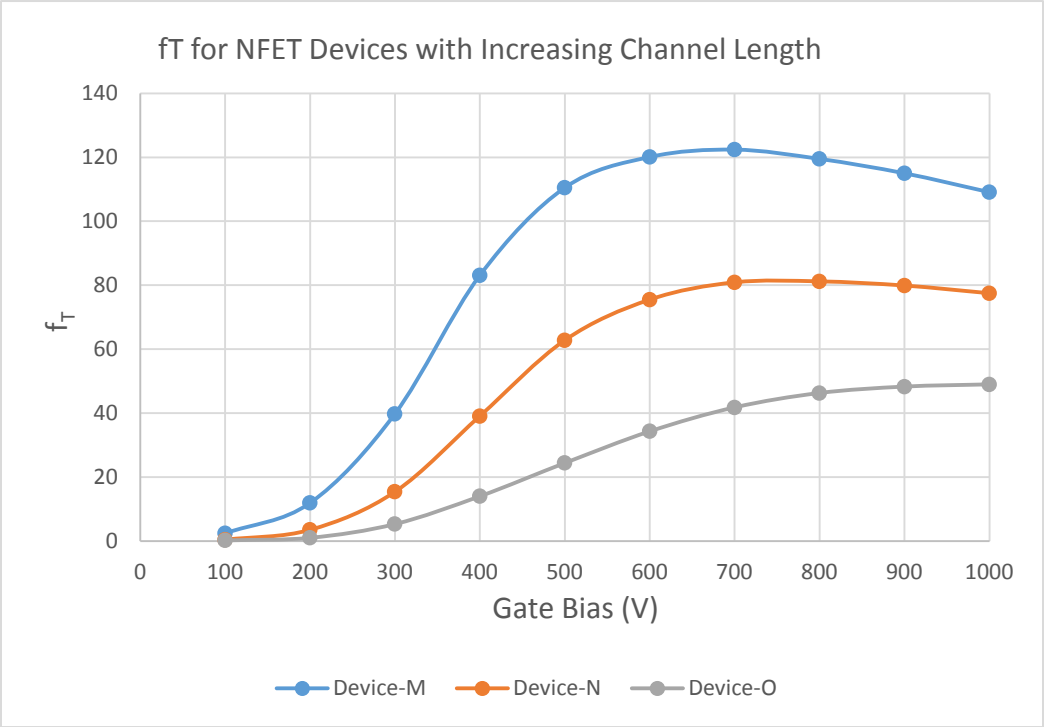


Figure 21. f_T vs. gate bias for the three NFET devices with increasing channel lengths

7.5. The Effects of Body Type (Floating Body vs. Body Contacted):

The differences (Δ) in measured and de-embedded peak S_{21} and f_T values for devices with varying body types (all other parameters equal) are provided in **Table 11** where

$$\Delta = S_{21}|\,|f_T\,(BT1) - S_{21}|\,|f_T\,(BT2)$$

Table 11. The Differences (Δ) in Measured and De-embedded Peak S_{21} and f_T Values for Devices with Varying Body Types (All Other Design Parameters Being Equal)

EFFECT OF BODY TYPE (Δ = Floating body parameter – Body tied parameter)						
Devices	**MOS, V_{TH} and Body Type 1, 2**	**W (μm)**	**N (fingers)**	**L (nm)**	**$\Delta S_{21\|peak}$ (dB)**	**$\Delta f_{T\|peak}$ (GHZ)**
A, Z	AVTFB, AVTBC (NFET)	24	12	56	5.4	95.0
E, M	AVTFB, AVTBC (NFET)	12	15	56	-2.4	-27.4

Peak S_{21} values for the floating-body NFET devices of equal dimensions are a third higher than the Body contacted devices. This is attributed to the increase transconductance associated with the floating body. However, the inverse was observed for the PFET devices. f_T behaved similarly.

7.6. The Effects of Threshold Voltage:

The differences (Δ) in measured and de-embedded peak S_{21} and f_T values for devices with varying body types (all other parameters equal) are provided in **Table 12** where

$$\Delta = S_{21}|\,|f_T\,(VT1) - S_{21}|\,|f_T\,(VT2)$$

Table 12. The Differences (Δ) in Measured and De-embedded Peak S_{21} and f_T Values for Devices with Varying Threshold Voltages (All Other Design Parameters Being Equal)

EFFECT OF THRESHOLD VOLTAGE (Δ = Type 1 parameter – Type 2 parameter)						
Devices	**V_{TH} Type 1, Type 2 (MOS Type)**	**W (μm)**	**N (fingers)**	**L (nm)**	**$\Delta S_{21\|peak}$ (dB)**	**$\Delta f_{T\|peak}$ (GHZ)**
G, H	RVT, SVT (NFET)	12	15	40	0.9	15.0
H, I	SVT, UVT (NFET)	12	15	40	0.3	-25.2

Minor relationship between threshold voltage and S_{21} was observed as the measured S_{21} for the RVT device was slightly greater than the S_{21} for the SVT device, which in turn was slightly greater than that for the UVT device. Comparing the values for increasing threshold voltage devices G, H and I, shows the f_T values decreasing with increased threshold, and the maximum value occurring at increasing gate voltage of 500mV, 700mV and 800mV respectively. This is expected since the higher Vt devices require a higher gate voltage to achieve the same overdrive. ***Figure 22*** shows the plots for f_T vs. gate bias for the three device types. The three curves tend to merge at maximum gate bias.

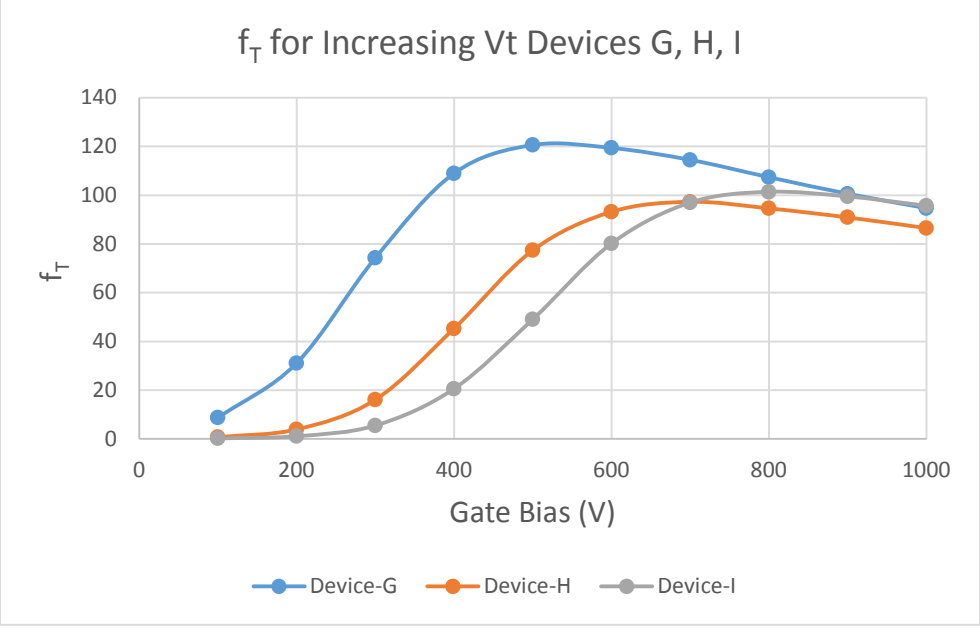


Figure 22. f_T vs. gate bias for the three NFET devices with increasing threshold voltage.

8.0 Technology Comparison: 32nm vs. 45nm CMOS

Tables 13 and ***14*** show a compilation of the measured and de-embedded peak S_{21} and f_T values for the 32nm devices compared to the previously reported values for the 45nm devices. The device description includes the MOS type (NFET and PFET), threshold variant or oxide type (analog V_T – AVT, regular V_T – RVT, super-high V_T – SVT, ultra-high V_T – UVT or thick-oxide – TO), body type (floating body – FB, and body contacted – BC), gate finger width (WF), number of fingers (N), total gate width (W = WF x N (W)), and gate length (L).

It can be seen that the 32nm technology generally offers a significantly higher S_{21} parameter (about 8dB at an average), but a significantly lower f_T values (about 45% at an average) than the 45nm technology.

Table 13. Comparison of Peak S21 Values for the Various 32nm and 45nm Technology Devices

		45nm	32nm	Delta(32nm-45nm)
ID	**Device**	**Peak(S_{21}) [dB]**	**Peak(S_{21}) [dB]**	**Peak(S_{21}) [dB]**
A	NFET, AVTFB, W = 2 µm x 12 (24 µm), L = 56 nm	8.9	16.3	7.4
B	NFET, AVTFB, W = 1.6 µm x 15 (24 µm), L = 56 nm	8.3	16.3	8.0
C	NFET, AVTFB, W = 1 µm x 24 (24 µm), L = 56 nm	8.9	16.3	7.4
D	NFET, AVTFB, W = 1 µm x 15 (15 µm), L = 56 nm	5.1	13.3	8.2
E	NFET, AVTFB, W = 0.8 µm x 15 (12 µm), L = 56 nm	4.3	9.5	5.2
F	PFET, AVTFB, W = 2 µm x 12 (24 µm), L = 56 nm	7.3	15.6	8.3
G	NFET, RVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	3.8	12.6	8.8
H	NFET, SVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	4.8	10.8	6.0
I	NFET, UVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	3.7	11.3	7.6
J	NFET, TOFB, W = 0.8 µm x 15 (12 µm), L = 112 nm	-1.6	9.7	11.3
K	NFET, AVTBC, W = 2 µm x 6 (12 µm), L = 56 nm	3.4	12.1	8.7
L	NFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	2.8	12.2	9.4
M	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 56 nm	3.4	11.9	8.5
N	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 112 nm	1.4	9.7	8.3
O	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 232 nm	NA	6.0	-
P	NFET, AVTBC, W = 1 µm x 15 (15 µm), L = 232 nm	-0.6	10.2	10.8
U	PFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 232 nm	-16	1.4	17.4
V	PFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 112 nm	-1.3	6.8	8.1
W	PFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	0.9	5.0	4.1
X	PFET, AVTFB, W = 1 µm x 24 (24 µm), L = 56 nm	6.4	15.3	8.9
Y	PFET, AVTFB, W = 0.8 µm x 15 (12 µm), L = 56 nm	0.2	10.8	10.6
Z	NFET, AVTBC, W = 2 µm x 12 (24 µm), L = 56 nm	9.2	10.9	1.7
AA	NFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	1.1	-	-
	Average			174.7/21=8.3

Table 14. Comparison of Peak f_T Values for the Various 32nm and 45nm Technology Devices

		45nm	32nm	% (32nm/45nm)
ID	**Device**	**Peak(f_T) [GHz]**	**Peak(f_T) [GHz]**	**Peak(f_T) [GHz]**
A	NFET, AVTFB, W = 2 µm x 12 (24 µm), L = 56 nm	296.7	179.7	0.605662285
B	NFET, AVTFB, W = 1.6 µm x 15 (24 µm), L = 56 nm	238.1	159.6	0.670306594
C	NFET, AVTFB, W = 1 µm x 24 (24 µm), L = 56 nm	203.1	163.9	0.80699163
D	NFET, AVTFB, W = 1 µm x 15 (15 µm), L = 56 nm	285.2	97.4	0.341514727
E	NFET, AVTFB, W = 0.8 µm x 15 (12 µm), L = 56 nm	313.2	67.4	0.215197957
F	PFET, AVTFB, W = 2 µm x 12 (24 µm), L = 56 nm	187.3	133.3	0.711692472
G	NFET, RVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	348	90.3	0.259482759
H	NFET, SVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	325.9	75.3	0.23105247
I	NFET, UVTFB, W = 0.8 µm x 15 (12 µm), L = 40 nm	153	100.5	0.656862745
J	NFET, TOFB, W = 0.8 µm x 15 (12 µm), L = 112 nm	106.8	77.1	0.721910112
K	NFET, AVTBC, W = 2 µm x 6 (12 µm), L = 56 nm	214.6	80	0.37278658
L	NFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	213.3	106.8	0.500703235
M	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 56 nm	192.9	94.8	0.491446345
N	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 112 nm	124.3	60.1	0.483507643
O	NFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 232 nm	-	36.2	-
P	NFET, AVTBC, W = 1 µm x 15 (15 µm), L = 232 nm	62.7	65.5	1.044657097
U	PFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 232 nm	39.2	29.3	0.74744898
V	PFET, AVTBC, W = 0.8 µm x 15 (12 µm), L = 112 nm	86.7	55.8	0.643598616
W	PFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	176.9	44.3	0.250423968
X	PFET, AVTFB, W = 1 µm x 24 (24 µm), L = 56 nm	142.2	142.6	1.00281294
Y	PFET, AVTFB, W = 0.8 µm x 15 (12 µm), L = 56 nm	193	76.8	0.397927461
Z	NFET, AVTBC, W = 2 µm x 12 (24 µm), L = 56 nm	155	84.7	0.546451613
AA	NFET, AVTBC, W = 1 µm x 12 (12 µm), L = 56 nm	193.9	-	-
	Average			11.694/21=0.557

9.0 Summary

This document details the test results from DC and RF experiments, performed at Vanderbilt University for a commercial 32 nm RF SOI CMOS technology. The devices-under-test (DUT) consist of large multi-finger MOSFETs with varying lengths, widths, threshold voltage implants, oxide thickness and body contacting schemes. It also includes varactors, capacitors and inductors and other RF calibration structures such as open, short, load and through for de-embedding purposes.

DC parameters and de-embedded s-parameter measurements are provided, as well as f_T calculations. Observations from the measurements include the effects of transistor type, device width, number of fingers, gate length, body type, and threshold implant on forward voltage gain (S_{21}) and unity current-gain cutoff frequency (f_T).

The following general observations can be made for the DC parameters:

- The devices were on the weak side of the distribution, representing approximately a -3σ distribution compared to the device model.
- The gate leakage is insignificant compared to the channel leakage thanks to the use of high-k gate dielectric, and can therefore be disregarded.
- The high Vt device (UVT) reduces leakage by > order of magnitude compared to standard device (RVT) at the expense of 50% reduced drive current.
- Doubling the number of fingers reduces normalized current (/um) by about 20%.
- Body tie reduces channel leakage by as much as 75% at the expense of reduced drive current (50%).
- Device current is reasonably proportional to 1/L indicating good short channel behavior.
- A robust PFET/NFET ratio support effectiveness of stress engineering on carrier mobility.

The following general observations can be made for the RF parameters:

- The coplanar waveguide impedance was on target at 50Ω, and was purely resistive through at least 30GHz.
- The forward gain S_{21} Parameters showed minimal degradation through 1GHz, with some degradation through 10GHz. This degradation is accentuated by packaging.
- The median value for S_{21} was around 12dB for the various NFET devices and around 10dB for the various PFET devices measured.
- The peak cut-off frequency, f_T, occurred at a typical gate voltage of about 600mV, but varied between 200mV and 1.0V for the various device types.
- f_T was about 20GHz higher for the NFET Compared to PFET devices
- In comparison with the a similar 45nm technology previously evaluated, the 32nm CMOS offered about 8dB higher S_{21} forward gain, but about 45% lower f_T at an average.

APPENDIX B

MATLAB CODE and PYTHON AUTOMATION SCRIPTS

MATLAB SCRIPT OVERVIEW: The following MATLAB scripts (pages 3-35) were written by Dr. Daniel Loveless. They were central to this thesis work for the characterization of the 32 nm RF SOI CMOS technology. The scripts are used de-embed, manipulate, extract and plot the S-parameters, I-V characteristics, and unity gain cut-off frequency (f_T) parameters.

RF MATLAB Analysis Code:

Top Level - **analyzeRF_20140528_new.m**

Referenced *.m scripts:

1. **deembed.m** (s-param de-embedding routine)
2. **getDevice.m** (parses filename for device # and die #)
3. **getGM.m** (grabs the gm for a specific bias condition)
4. **getMarkerProperties.m** (generates random marker type and colors)
5. **getVGVD.m** (parses filename for Vg and Vd values)
6. **getZ0.m** (determines the characteristic Z)
7. **setFigSize.m** (sets the parameters for figure dimension)
8. **setGM.m** (analyzes DC data and creates a file of GM values)
9. **setPLOT.m** (list of good marker colors and styles)

PYTHON Script Overview: The Python scripts (pages 36-58) were written by Dr. Andrew Sternberg. The scripts automate the lab equipment to sweep the I-V characteristics and S-parameter s and then save the data to a laptop via a GPIB connection for the HP 4156 Parametric Analyzer (PA) and the Agilent N5245A Parametric Network Analyzer (PNA-X). The automation reduced the experiment data collection time by nearly **10 times**.

1. **4156_IdVg.py** (collects I-V charateristics for all experiements)

2. **pna_meas.py** (collects S-parameters during Temperature and TID experiments)
3. **pna_meas_stress.py** (collect S-parameters during the RF Stress experiments)

```
%% analyzeRF_20140528_new.m
% Vanderbilt University
% Program written by: D. Loveless
% Main program for s-parameter deembedding, fT extraction, plotting
% and compiling data
% Last update: 2014/06/25 (by Timothy D. Haeffner)

% clear all;
clc;
setFigSize;     %setFigSize.m: pulls in desired parameters for
creating image

%% Script Flags
plotSParam=0;   %0-no plot, 1-Plot S21 vs Freq
plotFT=1;       %0-no plot, 1-fT vs VGS (*** processFT must be set to
1)
plotFTALL=0;    %0-only plot 1 fT estimate (given plotFT=1), 1-plots 3
fT estimates
plotGM=0;       %0-no plot, 1-gm vs VGS
plotDC=0;

processSP=0;    %0-no process, 1-process S-parameters, create
deembedded s-param files (processed_Vg###mvVd###.s2p)
processFT=1;    %0-no process, 1-process fT using 3 methods (H21
extrapolation, and bounded estimates using gm and Cg)
processTEMP=0;  %0-grab room temp data only, 1-grab temp folder given
by #TEMPlevel
grabPOST=0;     %0-pull IDVG_pre.dat, 1-get IDVG_post.dat

writeFIG=0;     %0-do not save figure, 1-save figure
writeCompiled=0;%0-do not write s-parameter data, 1-write s-parameter
data to compiled.txt
writeFT=1;      %0-do not write fT data, 1-write fT data to fT.txt

%% Analysis Directories
baseDIR='/Users/thaeffner/Desktop/32nm_MATLAB_Scripts/';  % Top-level
direcgtory
baseGMdir=strcat(baseDIR,'/DC/');                          %
Directory (original files) for looking for backup IDVG data if
IDVG_pre(post).dat doesn't exist

evalDIRA=baseDIR;
evalDIRB='32nm_RF_Data/32nm_RF_Sparameter_Data_201404/';    % Sub-
directory containing s-parameter folders
cd(strcat(evalDIRA,evalDIRB));                             % Navigate
to s-parameter sub-directory

%% Grab the die number (e.g. 1-10, ...) and device name (e.g. A-Z,
...) in the directory
dirlist=dir(strcat(evalDIRA,evalDIRB,'*krad*'));    % List of folders
under evalDIRB
```

```
[dietmp, devtmp]=getDevice(dirlist);

%% If you want to manually run individual die and devices, uncomment
the following 2 lines and place in desired parameters
dieNUM={'15'};
devID={'F'};
% And comment th6e following 2 lines. If you uncomment the following 2
lines
% you will run all files contained in evalDIRB
% dieNUM=dietmp;
% devID=devtmp;

TEMPlevel={'25'};
TIDlevel={'500'};

% If temperature data is not being processed, ignore the TEMPlevel and
set
% to 0C for legend print outs
if processTEMP==0,
    TEMPlevel={'25'};
end

%% Begin Analysis
% Loop through each die number
for jj=1:length(dieNUM),
    if
strcmp(devID(jj),'F')==1||strcmp(devID(jj),'U')==1||strcmp(devID(jj),'
V')==1||strcmp(devID(jj),'W')==1||strcmp(devID(jj),'X')==1,
        typeVAL=2; % These are PMOS devices
    else
        typeVAL=1; % These are NMOS devices
    end

    % Grab sub-directory of TID level, device #, die #, and temp (if
necessary)
    if processTEMP==1, % If temperature data is being processed,
append the temp level in order to navigate to appropriate folder

evalDIRC=char(strcat(TIDlevel,'krad_Die',dieNUM(jj),'_Device',devID(jj
),'_',TEMPlevel,'C/'));
    else

evalDIRC=char(strcat(TIDlevel,'krad_Die',dieNUM(jj),'_Device',devID(jj
),'/'));
    end
    % Look into folder and list unprocessed .s2p files, grab Vg and Vd
    filelist=dir(strcat(evalDIRC,'Vg*.s2p'));   % List of folders
under evalDIRC
    [vgtmp, vdtmp]=getVGVD(filelist);

%% If you want to manually plot specific values together, uncomment
the following 7 and place the desired parameters in the cell
```

```
    % Dies 1, 2, and 3 have a reduced set of Vg and Vd values (special
    % cases). Otherwise, default is for VD=1V, and VG=0:.1:1 V.
    % *NOTE: If a high voltage device, YOU MUST MANUALLY ADD DESIRED
VOLTAGE LEVELS*
    % *NOTE: EVERY VgNUM MUST HAVE A CORRESPONDING VdNUM*
    if
strcmp(dieNUM,'1')==1||strcmp(dieNUM,'2')==1||strcmp(dieNUM,'3')==1,
        VgNUM={'600' '1000'};
        VdNUM={'1000' '1000'};
    else
%         VgNUM={'600' '600' '600'};
%         VdNUM={'300' '600' '1000'};
        VgNUM={'100' '200' '300' '400' '500' '600' '700' '800' '900'
'1000'};
        VdNUM={'1000' '1000' '1000' '1000' '1000' '1000' '1000' '1000'
'1000' '1000'};
%         VgNUM={'700'};
%         VdNUM={'1000'};
    end
    % And comment the following 2 lines. If you uncomment the
following 2 lines
    % you will run all files contained in evalDIRC
%     VgNUM=vgtmp;
%     VdNUM=vdtmp;

    count=1;                        % Data counter (counts number of
Vg values)
    fT=zeros(length(VgNUM),1);      % Variable holder for fT (type
array)
    gmVEC=zeros(length(VgNUM),1);   % Variable holder for gm (type
array)
    vgVEC=cell(length(VgNUM),1);
    vdVEC=vgVEC;
    spVEC=gmVEC;
    %% Data Loop
    % Loop through each measurement value
    for ii=1:length(VgNUM),
        [cvec(1,:),fvec(1,:),mvec{1}]=getMarkerProperties();    %
Generates a psuedo-random marker color (cvec), face color (fvec), and
marker symbol (mvec) for every
        [cvec(2,:),fvec(2,:),mvec{2}]=getMarkerProperties();
        [cvec(3,:),fvec(3,:),mvec{3}]=getMarkerProperties();
        [cvec(4,:),fvec(4,:),mvec{4}]=getMarkerProperties();

        %Go grab the s-parameter file to analyze

fname=strcat(evalDIRA,evalDIRB,evalDIRC,'Vg',VgNUM{ii},'mvVd',VdNUM{ii
});
        readfile = strcat(fname,'.s2p');

        % Save a logfile (log.txt) of the command window outputs
        diary(strcat(evalDIRA,evalDIRB,evalDIRC,'log.txt'));
```

```
        diary on;

        % Loopk at s-parameter file and determine if there are headers
or
        % not. Set the appropriate starting row accordingly.
        tempFile=fopen(char(readfile)); % s-param file
        line1=fgetl(tempFile);          % read the first line
        if strcmp(line1(1),'!')==1,     % header detected
            startrow1=9;                % data starts on row 9
        else
            startrow1=0;                % no header, data starts on
row 0
        end
        fclose(tempFile);               % close the file

        %% Main Analysis
        % Find the GM file: First, look if the GM.csv file exists.  If
it doesn't exist,
        % look for the IDVG_pre(post).dat file, and create GM.csv.  If
that IDVG_pre(post).dat
        % doesnt exist, look in another chip directory (#baseGMdir).

        baseIDVGdir=strcat(evalDIRA,evalDIRB,evalDIRC); % Analysis
directory
        IDVGprefile='IDVG_pre.dat';                     % IDVG pre
sweep file
        IDVGpostfile='IDVG_post.dat';                   % IDVG post
sweep file
        if grabPOST==1,
            IDVGfile=IDVGpostfile;
        else
            IDVGfile=IDVGprefile;
        end
        if plotDC==1,
            if ii==1,

plotIDVG(strcat(evalDIRA,evalDIRB,evalDIRC,IDVGfile),typeVAL,devID(jj)
,dieNUM(jj),TEMPlevel,TIDlevel);
            end
        end
        % First check for GM.csv in the current analysis directory
        if exist(strcat(baseIDVGdir,'GM.csv'),'file')==2,
            % If GM.csv file exists, then use it (set variable gmfile)
            display(strcat({'GM file found for Chip '},dieNUM(jj),{',
Device '},devID(jj)));
            gmfile=strcat(baseIDVGdir,'GM.csv');
        % Else, look for IDVG_pre(post).dat
        elseif exist(strcat(baseIDVGdir,IDVGfile),'file')==2,
            % If IDVG_pre(post).dat exists, then create a new GM.csv
file
            % (by calling setGM.m)
```

```
            display(strcat('Found',{' '},IDVGfile,{'. Attempting to
extract the GM...'}));
            gmfile=setGM(baseIDVGdir,IDVGfile,typeVAL); %typeVAL=1-for
NMOS, 2-for PMOS
        % Else, go the original measurements directoy, and look for
one
        % (there may be a die mismatch which will probably increase
error)
        else
            display('No IDVG.dat file or GM.csv file.  Looking for
another one ...');

gname=strcat(devID(jj),'/Chip',dieNUM(jj),'Device',devID(jj),'_GM.csv'
);
            gmfile=char(strcat(baseGMdir,gname));
            if exist(char(gmfile),'file')==2,
                display(strcat('File',{' /'},gname,{' is being used to
extract GM'}));
            else
                display(strcat('File',{' /'},gname,{' does not exist.
Trying another file...'}));

gmlist=dir(char(strcat(baseGMdir,devID(jj),'/*_GM.csv')));
                if isempty(gmlist),
                    display('No GM file found. Exiting');
                    break;
                else
                    gname=gmlist(1).name;
                    gmfile=strcat(baseGMdir,devID(jj),'/',gname);
                    display(strcat('Using file',{' '},gmfile,{'
instead.'}));
                end
            end
        end

        % Look in the GM file and find the value at VG and VD
        vgval=str2double(VgNUM{ii})/1000;
        vdval=str2double(VdNUM{ii})/1000;
        gm=getGM(vgval,vdval,gmfile);

        display(strcat('Vg =',{' '},VgNUM{ii},{' mV'}));
        display(strcat('Vd =',{' '},VdNUM{ii},{' mV'}));
        display(strcat('gm =',{' '},num2str(gm/1e-3,3),{' mS'}));

        % Open, Short, Load, and Source Calibration Files
        openfile = 'Open.s2p';

of=strcat(evalDIRA,'32nm_RF_Data/Testchip_Calibration_Structures/',ope
nfile);
        shortfile = 'Short.s2p';
```

```
sf=strcat(evalDIRA,'32nm_RF_Data/Testchip_Calibration_Structures/',sho
rtfile);
        loadfile='Load1.s2p';

lf=strcat(evalDIRA,'32nm_RF_Data/Testchip_Calibration_Structures/',loa
dfile);
        zsfile = 'Zs.csv';
        zscomplexfile = 'Zscomplex.csv';

        % Create the filenames to save the processed s-parameters
        % (deembedded) and image (not currently used)
        saveSPfile =
strcat(evalDIRA,evalDIRB,evalDIRC,'processed_','Vg',VgNUM{ii},'mvVd',V
dNUM{ii},'.s2p');
        s_plot =
strcat(evalDIRA,evalDIRB,evalDIRC,'processed_','Vg',VgNUM{ii},'mvVd',V
dNUM{ii},'.tif'); % not currently used

        % Get the raw s-parameters
        [freq, S11A, S11B, S21A, S21B, S12A, S12B, S22A, S22B] =
textread(char(readfile), '%f%f%f%f%f%f%f%f%f', 'headerlines',
startrow1); %#ok<*DTXTRD>
        [freq_o, S11A_o, S11B_o, S21A_o, S21B_o, S12A_o, S12B_o,
S22A_o, S22B_o] = textread(of, '%f%f%f%f%f%f%f%f%f', 'headerlines',
9); % assumes the open calibration file has a header
        [freq_s, S11A_s, S11B_s, S21A_s, S21B_s, S12A_s, S12B_s,
S22A_s, S22B_s] = textread(sf, '%f%f%f%f%f%f%f%f%f', 'headerlines',
9); % assumes the short calibration file has a header

        % Get the source impedance: zs is in magnitude form, zsc is in
        % complex form
        [zs] =
textread(strcat(evalDIRA,'32nm_RF_Data/Testchip_Calibration_Structures
/',zsfile), '%f', 'headerlines', 0); % assumes the source calibration
file does not have a header
        [zsc] =
csvread(strcat(evalDIRA,'32nm_RF_Data/Testchip_Calibration_Structures/
',zscomplexfile));

        % Get the characteristic impedance
        [freq2,Z0]=getZ0(of,sf,lf,3,0);

        % Error flag indicating a problem with an s-param extraction
        % (length of one or more of the files is incorrect - different
        % frequencies)
        if
length(freq)~=length(freq2)||length(freq)~=length(freq_o)||length(freq
)~=length(freq_s),
            display('There is an error in the frequency extraction of
the s-parameters');
            break;
```

```
        end

        % De-embed the s-parameters (call deembed.m)
        % Outputs:
        % S11_d: deembedded S11 in complex form
        % S11_CS: deembedded S11 magnitudes
        % S12_complex: deembedded S12 in complex form (different from
'_d' to indicate compensation from source degeneration)
        % S12_CS: deembedded S12 magnitudes
        % S21_complex: deembedded S21 in complex form (different from
'_d' to indicate compensation from source degeneration)
        % S21_CS: deembedded S21 magnitudes
        % S22_d: deembedded S22 in complex form
        % S22_CS: deembedded S22 magnitudes
        % CGval: gate capactance in F
        %
        % Inputs:
        % all vectors from s-param. files: freq, S11A, S11B, S21A,
S21B, S12A, S12B, S22A, S22B, S11A_o, S11B_o, S21A_o, S21B_o, S12A_o,
S12B_o, S22A_o, S22B_o, S11A_s, S11B_s, S21A_s, S21B_s, S12A_s,
S12B_s, S22A_s, S22B_s
        % Source impedances: zs (mag),zsc (complex)
        % Characteristic Z: Z0
        % Transconductance: gm

[S11_d,S11_CS,S12_complex,S12_CS,S21_complex,S21_CS,S22_d,S22_CS,CGval
]=deembed(freq, S11A, S11B, S21A, S21B, S12A, S12B, S22A, S22B,
S11A_o, S11B_o, S21A_o, S21B_o, S12A_o, S12B_o, S22A_o, S22B_o,
S11A_s, S11B_s, S21A_s, S21B_s, S12A_s, S12B_s, S22A_s,
S22B_s,zs,zsc,Z0,gm);

        gmVEC(ii)=gm;               % Populate a vector of GM values
        if ii==1,
            CGlow=CGval;            % Gate cap. estimate for low bias
        elseif ii==length(VgNUM)
            CGhigh=CGval;           % Gate cap. estimate for high bias
        end
        CGvec(ii)=CGval;

        % Write the de-embedded S-parameters (magnitudes) to a file
        if processSP==1,
            fid = fopen(saveSPfile, 'a');

fprintf(fid,'%s\t%s\t%s\t%s\t%s\n','Freq','S11','S12','S21','S22');
            for i=1:length(freq),

fprintf(fid,'%6.1f\t%6.1f\t%6.1f%6.1f%6.1f\n',freq(i),S11_CS(i),S12_
CS(i),S21_CS(i),S22_CS(i));
            end
            fclose(fid);
        end
```

```
        %% FT (Method 1: H21 extrapolation to 0dB)
        if processFT==1,
            % Loop through each frequency
            H11=zeros(1,length(freq));
            H21=H11;
            H12=H11;
            H22=H11;
            for index=1:length(freq),
                s_params_deembed=[S11_d(index) S12_complex(index);
S21_complex(index) S22_d(index)];
                h_params=s2h(s_params_deembed,Z0(index));   % Convert
to H-parameters (must be in complex form)
                H11(index) = h_params(1,1);
                H21(index) = h_params(2,1);
                H12(index) = h_params(1,2);
                H22(index) = h_params(2,2);
            end
            H21_d = 20*log10(abs(H21));         % Magnitude of H21
            H21_s = smooth(H21_d,401,'sgolay'); % Magnitude of H21
(smoothed)

            tmpINDEX=find(freq>=0e9,1):find(freq>=20e9,1);  % Reduce
range of frequencies to fit H21 curve to
            freqREDUCE=freq(tmpINDEX);                      % Grab
Freq values in desired range
            h21REDUCE=H21_s(tmpINDEX);                      % Grab H21
values of desired frequency range
            hfit=fit(h21REDUCE,freqREDUCE,'poly1');         % Do a
polynomial fit (linear) to the Freq. VS H21
            fT(ii)=(hfit(20)*10)/1e9;                       % Find the
point at 20 dB and multiply by 1 decade (10) to estimate the value at
0 dB
        end

        % Populate vectors of vg, vd, and S21 alues
        vgVEC(count)=VgNUM(ii);
        vdVEC(count)=VdNUM(ii);
        spVEC(count)=max(S21_CS);

        %% PLOT S-parameters
        if plotSParam==1,
            plot_title = strcat('Chip',{' '},char(dieNUM(jj)),{'
'},'Device',{' '},devID(jj),', S_{21} for V_{DS}=1 V');
            strLegend=strcat('V{_g}=',VgNUM{ii},{' mV
,V{_d}='},VdNUM{ii},{' mV, Temp='},TEMPlevel,{' C, TID='},TIDlevel,{'
krad(SiO_2)'});
            % If this is the first plot, then start a list for the
legend
            if exist('strList','var')==0,
                strList={char(strLegend)};
            % If there is already a legend list (you want to plot
multiple
```

```
            % curves together), grab the current variable at
concentate the
            % new run (add a new legend entry)
            else
                strList=[strList strLegend]; %#ok<AGROW>
            end
           % If this is the first plot, open a new figure (requires
           % variables from setFigSize.m)
            if isempty(findobj('type','figure'))==1,
                figure1 = figure('Position',...
                    [scrsz(1)+xoffset scrsz(2)+yoffset xwidth
yheight]);
                axes1 = axes('Parent',figure1);
                box(axes1,'on');
                hold(axes1,'all')
            end

            % Plot S21 vs. frequency (requires variables from
            % getMarkerProperties.m)
            p1=plot(freq,S21_CS,char(strcat('-',mvec{1})),...
                        'MarkerSize',10,'Color',cvec(1,:),...
                        'MarkerFaceColor',fvec(1,:),...
                        'LineWidth',2);
            set(gca,'FontSize',11,'XScale','log','YScale','linear',...
                    'GridLineStyle','none','XLim',[1e7
100E9],'YLim',[-30 20],...
                    'MinorGridLineStyle','none',...

'FontSize',28,'XMinorGrid','on','FontName','FixedWidth');
            xlabel('Frequency
(Hz)','FontSize',32,'FontName','FixedWidth');
            ylabel('S_{21}
(dB)','FontSize',32,'FontName','FixedWidth');
            legend(strList,'Location','EastOutside','FontSize',22);
        end
        count=count+1;
    end % End ii=1:length(VgNUM) loop

    %% fT (Methods 2 and 3: gm/Cg)
    if processFT==1,
        fTestimateLOW=gmVEC./CGlow;     % gate cap estimate at low
freqs. and low bias
        fTestimateHIGH=gmVEC./CGhigh;   % gate cap estimate at low
freqs. and high bias
        fTestimate=gmVEC./CGvec';
    end

    %% Plot fT
    if processFT==1 && plotFT==1,
        % If this is the first plot, open a new figure (requires
        % variables from setFigSize.m)
        if isempty(findobj('type','figure'))==1,
```

```
            figure2 = figure('Position',...
                [scrsz(1)+xoffset scrsz(2)+yoffset xwidth yheight]);
            axes1 = axes('Parent',figure2); %#ok<*LAXES>
            box(axes1,'on');
            hold(axes1,'all')
        end

        % Uncomment/comment the three versions of the fT estimate you
want
        % to plot

        if typeVAL==1,
            xlabel('V_{GS}
(V)','FontSize',32,'FontName','FixedWidth');
            xvec=str2num(char(VgNUM))/1000; %#ok<*ST2NM>
        else
            xlabel('|V_{GS}|
(V)','FontSize',32,'FontName','FixedWidth');
            xvec=-((str2num(char(VgNUM))/1000)-1);
        end

        p1=plot(xvec,fTestimateLOW/1e9,char(strcat('-',mvec{2})),...
                            'MarkerSize',10,'Color',cvec(2,:),...
                            'MarkerFaceColor',fvec(2,:),...
                            'LineWidth',2);
        if plotFTALL==1,
            p2=plot(xvec,fTestimateHIGH/1e9,char(strcat('-
',mvec{1})),...
                            'MarkerSize',10,'Color',cvec(1,:),...
                            'MarkerFaceColor',fvec(1,:),...
                            'LineWidth',2);
            p3=plot(xvec,fTestimateLOW/1e9,char(strcat('-
',mvec{3})),...
                'MarkerSize',10,'Color',cvec(3,:),...
                'MarkerFaceColor',fvec(3,:),...
                'LineWidth',2);
            p4=plot(xvec,fTestimate/1e9,char(strcat('-',mvec{4})),...
                'MarkerSize',10,'Color',cvec(4,:),...
                'MarkerFaceColor',fvec(4,:),...
                'LineWidth',2);
        end

        set(gca,'FontSize',11,'XScale','linear','YScale','linear',...
                'GridLineStyle','none','XLim',[0 1],'YLim',[0 200],...
                'MinorGridLineStyle','none',...

'FontSize',28,'XMinorGrid','on','FontName','FixedWidth');

        ylabel('f_{T} (GHz)','FontSize',32,'FontName','FixedWidth');

        % Build legend entries
```

```
        strLegenda=strcat({'Device '},devID,{', Chip '},dieNUM,{',
Temp='},TEMPlevel,{' C, TID='},TIDlevel,{' krad(SiO_2), f_{T}'});
        if exist('strList2a','var')==0,
            strList2a={char(strLegenda)};
        else
            strList2a=[strList2a strLegenda]; %#ok<AGROW>
        end
        strList2=strList2a;
        if exist('p1','var')==1 && exist('p2','var')==1,
            strLegendb=strcat({'Device '},devID,{', Chip '},dieNUM,{',
Temp='},TEMPlevel,{' C, TID='},TIDlevel,{' krad(SiO_2), f_{T|HIGH}'});
            if exist('strList2b','var')==0,
                strList2b={char(strLegendb)};
            else
                strList2b=[strList2b strLegendb]; %#ok<AGROW>
            end
            strList2=[strList2 strList2b]; %#ok<AGROW>
        end
        if exist('p1','var')==1 && exist('p2','var')==1 &&
exist('p3','var')==1,
            strLegendc=strcat({'Device '},devID,{', Chip '},dieNUM,{',
Temp='},TEMPlevel,{' C, TID='},TIDlevel,{' krad(SiO_2), f_{T|LOW}'});
            if exist('strList2c','var')==0,
                strList2c={char(strLegendc)};
            else
                strList2c=[strList2c strLegendc]; %#ok<AGROW>
            end
            strList2=[strList2 strList2c]; %#ok<AGROW>
        end
        if exist('p1','var')==1 && exist('p2','var')==1 &&
exist('p3','var')==1 && exist('p4','var')==1,
            strLegendd=strcat({'Device '},devID,{', Chip '},dieNUM,{',
Temp='},TEMPlevel,{' C, TID='},TIDlevel,{' krad(SiO_2), f_{T|est}'});
            if exist('strList2d','var')==0,
                strList2d={char(strLegendd)};
            else
                strList2d=[strList2d strLegendd]; %#ok<AGROW>
            end
            strList2=[strList2 strList2d]; %#ok<AGROW>
        end
        legend(strList2,'Location','Northwest','FontSize',26);
    end

     %% Plot gm
    if plotGM==1,
        % If this is the first plot, open a new figure (requires
        % variables from setFigSize.m)
        if isempty(findobj('type','figure'))==1,
            figure3 = figure('Position',...
                [scrsz(1)+xoffset scrsz(2)+yoffset xwidth yheight]);
            axes1 = axes('Parent',figure3); %#ok<*LAXES>
            box(axes1,'on');
```

```
            hold(axes1,'all')
        end

        % Uncomment/comment the three versions of the fT estimate you
want
        % to plot

        if typeVAL==1,
            xlabel('V_{GS}
(V)','FontSize',32,'FontName','FixedWidth');
            xvec=str2num(char(VgNUM))/1000; %#ok<*ST2NM>
        else
            xlabel('|V_{GS}|
(V)','FontSize',32,'FontName','FixedWidth');
            xvec=-((str2num(char(VgNUM))/1000)-1);
        end

        p1=plot(xvec,gmVEC/1e-3,char(strcat('-',mvec{2})),...
                            'MarkerSize',10,'Color',cvec(2,:),...
                            'MarkerFaceColor',fvec(2,:),...
                            'LineWidth',2);
        set(gca,'FontSize',11,'XScale','linear','YScale','linear',...
                'GridLineStyle','none','XLim',[0 1],'YLim',[0 50],...
                'MinorGridLineStyle','none',...

'FontSize',28,'XMinorGrid','on','FontName','FixedWidth');
        ylabel('g_{m} (mS)','FontSize',32,'FontName','FixedWidth');

        % Build legend entries
        strLegenda=strcat({'Device '},devID,{', Chip '},dieNUM,{',
Temp='},TEMPlevel,{' C, TID='},TIDlevel,{' krad(SiO_2), g_{m}'});
        if exist('strList2a','var')==0,
            strList2a={char(strLegenda)};
        else
            strList2a=[strList2a strLegenda]; %#ok<AGROW>
        end
        strList2=strList2a;
        legend(strList2,'Location','Northwest','FontSize',26);
    end

    %% Write compiled S21 data
    if writeCompiled==1,
        fid = fopen(strcat(evalDIRA,evalDIRB,evalDIRC,'compiled.txt'),
'a');
        fprintf(fid,'%s\t%s\t%s\t%s\n',...
            'Device','Vg (mV)','Vd (mV)','Peak S21 (dB)');
        for k=1:count-1,
            fprintf(fid,'%s\t%6.1f\t%6.1f\t%6.1f\n',...
                char(devID(jj)),[str2double(vgVEC(k))
str2double(vdVEC(k)) spVEC(k)]);
        end
        fclose(fid);
```

```
    end

    %% Write fT data
    if processFT==1 && writeFT==1,
      display(strcat({'Peak fT = '},num2str(max(fT))));
        display(strcat({'Peak fTHIGH =
'},num2str(max(fTestimateHIGH/1e9))));
        display(strcat({'Peak fTLOW =
'},num2str(max(fTestimateLOW/1e9))));
        display(strcat({'Peak fTest =
'},num2str(max(fTestimate/1e9))));
        fid = fopen(strcat(evalDIRA,evalDIRB,'fT.txt'), 'a');
        fprintf(fid,'%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\n',...
            'Die#','Device','Vg (mV)','Vd (mV)','Peak fT
(GHz)','fT(gm/Cg)|LOW (GHz)','fT(gm/Cg)|HIGH (GHz)','fT(gm/Cg)|est
(GHz)');
        for k=1:count-1,

fprintf(fid,'%s\t%s\t%6.1f\t%6.1f\t%6.1f\t%6.1f\t%6.1f\t%6.1f\n',...
                char(dieNUM(jj)),char(devID(jj)),[str2double(vgVEC(k))
str2double(vdVEC(k)) fT(k) fTestimateLOW(k)/1e9 fTestimateHIGH(k)/1e9
fTestimate(k)/1e9]);
        end
        fclose(fid);
    end
    diary off;
end
```

%%deembed.m script

```
function
[S11_d,S11_CS,S12_complex,S12_CS,S21_complex,S21_CS,S22_d,S22_CS,CGval
]=deembed(freq, S11A, S11B, S21A, S21B, S12A, S12B, S22A, S22B,
S11A_o, S11B_o, S21A_o, S21B_o, S12A_o, S12B_o, S22A_o, S22B_o,
S11A_s, S11B_s, S21A_s, S21B_s, S12A_s, S12B_s, S22A_s,
S22B_s,szs,szsc,Z0,gm)
    % % What format is the s2p file?
    % % 1 = MA (Magnitude/Angle)
    % % 2 = RI (Real/Imaginary)
    % % 3 = DB (dB/Angle)
    format = 3;

    switch (format)
    case 1
        S11 = S11A.*exp(1i*S11B*pi/180);
        S21 = S21A.*exp(1i*S21B*pi/180);
        S12 = S12A.*exp(1i*S12B*pi/180);
        S22 = S22A.*exp(1i*S22B*pi/180);
        clear S11A S11B S21A S21B S12A S12B S22A S22B;

        S11_o = S11A_o.*exp(1i*S11B_o*pi/180);
        S21_o = S21A_o.*exp(1i*S21B_o*pi/180);
        S12_o = S12A_o.*exp(1i*S12B_o*pi/180);
        S22_o = S22A_o.*exp(1i*S22B_o*pi/180);
        clear S11A_o S11B_o S21A_o S21B_o S12A_o S12B_o S22A_o S22B_o;

        S11_s = S11A_s.*exp(1i*S11B_s*pi/180);
        S21_s = S21A_s.*exp(1i*S21B_s*pi/180);
        S12_s = S12A_s.*exp(1i*S12B_s*pi/180);
        S22_s = S22A_s.*exp(1i*S22B_s*pi/180);
        clear S11A_s S11B_s S21A_s S21B_s S12A_s S12B_s S22A_s S22B_s;
    case 2
        S11 = S11A + 1i*S11B;
        S21 = S21A + 1i*S21B;
        S12 = S12A + 1i*S12B;
        S22 = S22A + 1i*S22B;
        clear S11A S11B S21A S21B S12A S12B S22A S22B;

        S11_o = S11A_o + 1i*S11B_o;
        S21_o = S21A_o + 1i*S21B_o;
        S12_o = S12A_o + 1i*S12B_o;
        S22_o = S22A_o + 1i*S22B_o;
        clear S11A_o S11B_o S21A_o S21B_o S12A_o S12B_o S22A_o S22B_o;

        S11_s = S11A_s + 1i*S11B_s;
        S21_s = S21A_s + 1i*S21B_s;
        S12_s = S12A_s + 1i*S12B_s;
        S22_s = S22A_s + 1i*S22B_s;
        clear S11A_s S11B_s S21A_s S21B_s S12A_s S12B_s S22A_s S22B_s;
    case 3
```

```
        S11 = 10.^(S11A/20).*exp(1i*S11B*pi/180);
        S21 = 10.^(S21A/20).*exp(1i*S21B*pi/180);
        S12 = 10.^(S12A/20).*exp(1i*S12B*pi/180);
        S22 = 10.^(S22A/20).*exp(1i*S22B*pi/180);
        clear S11A S11B S21A S21B S12A S12B S22A S22B;

        S11_o = 10.^(S11A_o/20).*exp(1i*S11B_o*pi/180);
        S21_o = 10.^(S21A_o/20).*exp(1i*S21B_o*pi/180);
        S12_o = 10.^(S12A_o/20).*exp(1i*S12B_o*pi/180);
        S22_o = 10.^(S22A_o/20).*exp(1i*S22B_o*pi/180);
        clear S11A_o S11B_o S21A_o S21B_o S12A_o S12B_o S22A_o S22B_o;

        S11_s = 10.^(S11A_s/20).*exp(1i*S11B_s*pi/180);
        S21_s = 10.^(S21A_s/20).*exp(1i*S21B_s*pi/180);
        S12_s = 10.^(S12A_s/20).*exp(1i*S12B_s*pi/180);
        S22_s = 10.^(S22A_s/20).*exp(1i*S22B_s*pi/180);
        clear S11A_s S11B_s S21A_s S21B_s S12A_s S12B_s S22A_s S22B_s;

    end

      Y11=zeros(1,length(freq));
    Y12=Y11;
    Y21=Y11;
    Y22=Y11;
    Y11_o=Y11;
    Y12_o=Y11;
    Y21_o=Y11;
    Y22_o=Y11;
    Y11_s=Y11;
    Y12_s=Y11;
    Y21_s=Y11;
    Y22_s=Y11;
    Y11_d=Y11;
    Y12_d=Y11;
    Y21_d=Y11;
    Y22_d=Y11;
    S11_d=Y11;
    S12_d=Y11;
    S21_d=Y11;
    S22_d=Y11;
    CG=Y11;
    for index = 1:length(freq)
        y_params = s2y([S11(index) S12(index); S21(index)
S22(index)],Z0(index));
        Y11(index) = y_params(1,1);
        Y21(index) = y_params(2,1);
        Y12(index) = y_params(1,2);
        Y22(index) = y_params(2,2);
        y_params_o = s2y([S11_o(index) S12_o(index); S21_o(index),
S22_o(index)],Z0(index));
        Y11_o(index) = y_params_o(1,1);
        Y21_o(index) = y_params_o(2,1);
```

```
        Y12_o(index) = y_params_o(1,2);
        Y22_o(index) = y_params_o(2,2);
        y_params_s = s2y([S11_s(index) S12_s(index); S21_s(index),
S22_s(index)],Z0(index));
        Y11_s(index) = y_params_s(1,1);
        Y21_s(index) = y_params_s(2,1);
        Y12_s(index) = y_params_s(1,2);
        Y22_s(index) = y_params_s(2,2);

        Y = [Y11(index) Y12(index); Y21(index) Y22(index)];
        Y_o = [Y11_o(index) Y12_o(index); Y21_o(index) Y22_o(index)];
        Y_s = [Y11_s(index) Y12_s(index); Y21_s(index) Y22_s(index)];

        %H-parameters and FT
        Y_d = inv(inv(Y-Y_o)-inv(Y_s-Y_o));

        Y_total_w_short = (Y - Y_o);
        Y_short_wo_open = (Y_s - Y_o);
        Z_total_w_short = y2z(Y_total_w_short);
        Z_short_wo_open = y2z(Y_short_wo_open);
        Z_dut = (Z_total_w_short - Z_short_wo_open);
        s_params = z2s(Z_dut);

        Y11_d(index) = Y_d(1,1);
        Y21_d(index) = Y_d(2,1);
        Y12_d(index) = Y_d(1,2);
        Y22_d(index) = Y_d(2,2);

        %s_params=y2s([Y11_d Y12_d; Y21_d, Y22_d]);
        S11_d(index)=s_params(1,1);
        S12_d(index)=s_params(1,2);
        S21_d(index)=s_params(2,1);
        S22_d(index)=s_params(2,2);

        CG(index)=(-2/(2*pi*freq(index)))*imag(Y12_d(index)); %gate
capacticance in units of fC
        %
ZINprime(index)=Z0(index)*(1+S11_d(index))/(1-S11_d(index));
        %
ZIN(index)=ZINprime(index)+szsc(index)*(1+gm*ZINprime(index));
        %
CG2(index)=(imag(ZIN(index))/(2*pi*freq(index)));
    end

    CGval=mean(CG(1:30));
%     CGval=mean(CG);
%             ftE2(index)=gm/CG2(index);

    S12_CSS = 20*log10(abs(S12_d));
    S21_CSS = 20*log10(abs(S21_d));
    S12_CS=zeros(1,length(freq));
    S12_complex=zeros(1,length(freq));
```

```
    S21_CS=zeros(1,length(freq));
    S21_complex=zeros(1,length(freq));

    %Correct for source degeneration
    for index = 1:length(freq)
        S21_CS(index) = 20*log10(szs(index)*gm)+S21_CSS(index); %in dB
        S21_complex(index)=(1+szsc(index)*gm)*S21_d(index); %in
complex form

        S12_CS(index) = -20*log10(szs(index)*gm)+S12_CSS(index); %in
dB
        S12_complex(index)=S12_d(index)/(1+szsc(index)*gm); %in
complex form
    end

    % Final De-Embedded Parameters
    S21_CS = smooth(S21_CS',401,'sgolay'); %S21
    S12_CS = smooth(S12_CS',401,'sgolay'); %S21
    S11_CSS=20*log10(abs(S11_d));
    S11_CS=smooth(S11_CSS,401,'sgolay'); %S11
    S22_CSS=20*log10(abs(S22_d));
    S22_CS=smooth(S22_CSS,401,'sgolay'); %S22
end
```

%% getDevice.m

```
function [dietmp, devtmp]=getDevice(dirlist)
     dietmp=cell(1);                                        % variable
holer for die number (type cell)
    devtmp=cell(1);                                        % variable
holer for device name (type cell)
    % Loop through folders and list the files within each. Parse the
name in
    % order to grab the die number and device name
    for i=1:length(dirlist),
        tmp=dirlist(i).name;                               % Grab the
file names
        diestart=strfind(dirlist(i).name,'krad');          % Starting
index for die number
        dieend=strfind(dirlist(i).name,'Device');          % Ending index
for die number
        dietmp{i}=tmp(diestart+8:dieend-2);                % Die numer
        devtmp{i}=tmp(dieend+6);                           % Device name
    end
end
```

%%getGM.m

```
function gm=getGM(vgval,vdval,gmfile)
    if exist(char(gmfile),'file')==2,
        gmvals=csvread(char(gmfile));
    else
        display('There has been an error');
    end
    gmcol=gmvals(2:length(gmvals),gmvals(1,:)==vdval);
    gm=gmcol(gmvals(2:length(gmvals),1)==vgval)/1000;
end
```

%%getMarkerProperties.m

```
function [cvec,fvec,mvec]=getMarkerProperties()
    setPLOT;

    cvec(1,:)=colorvec(round(rand(1)*6+1),:);       %#ok<*NODEF>
%random color (7 choices)
     fvec(1,:)=colorvec(round(rand(1)*6+1),:);       %random face
color (7 choices)
     mvec{1}=char(markerVEC(round(rand(1)*6+1)));     %random marker
(7 choices)
end
```

%%getVDVG.m

```
function [vgtmp, vdtmp]=getVGVD(filelist)
    % Loop through list of filenames and parse the filenames and grab
Vgs and Vds values
    vgtmp=cell(1);  % Variable holder (type cell)
    vdtmp=cell(1);  % Variable holder (type cell)
    for i=1:length(filelist),
        tmp=filelist(i).name;                    % Grab the file names
        vgstart=strfind(filelist(i).name,'Vg'); % Starting index for
Vg
        vgend=strfind(filelist(i).name,'mv');   % Ending index for Vg
        vgtmp{i}=tmp(vgstart+2:vgend-1);        % Vg value

        vdstart=strfind(filelist(i).name,'Vd'); % Starting index for
Vd
        vdend=strfind(filelist(i).name,'.s');   % Ending index for Vd
        vdtmp{i}=tmp(vdstart+2:vdend-1);        % Vd value
    end
end
```

%%getZ0.m

```
function [freq,Z0]=getZ0(openfile,shortfile,loadfile,format,plotZ0)
     % % What format is the s2p file?
    % % 1 = MA (Magnitude/Angle)
    % % 2 = RI (Real/Imaginary)
    % % 3 = DB (dB/Angle)

    [freq, S11A, S11B, S21A, S21B, S12A, S12B, S22A, S22B] =
textread(char(loadfile), '%f%f%f%f%f%f%f%f%f', 'headerlines', 9);

    [~, S11A_o, S11B_o, S21A_o, S21B_o, S12A_o, S12B_o, S22A_o,
S22B_o] = textread(char(openfile), '%f%f%f%f%f%f%f%f%f',
'headerlines', 9);

    [~, S11A_s, S11B_s, S21A_s, S21B_s, S12A_s, S12B_s, S22A_s,
S22B_s] = textread(char(shortfile), '%f%f%f%f%f%f%f%f%f',
'headerlines', 9);

    switch (format)
    case 1
        S11 = S11A.*exp(1i*S11B*pi/180);
        S21 = S21A.*exp(1i*S21B*pi/180);
        S12 = S12A.*exp(1i*S12B*pi/180);
        S22 = S22A.*exp(1i*S22B*pi/180);
        clear S11A S11B S21A S21B S12A S12B S22A S22B;

        S11_o = S11A_o.*exp(1i*S11B_o*pi/180);
        S21_o = S21A_o.*exp(1i*S21B_o*pi/180);
        S12_o = S12A_o.*exp(1i*S12B_o*pi/180);
        S22_o = S22A_o.*exp(1i*S22B_o*pi/180);
        clear S11A_o S11B_o S21A_o S21B_o S12A_o S12B_o S22A_o S22B_o;

        S11_s = S11A_s.*exp(1i*S11B_s*pi/180);
        S21_s = S21A_s.*exp(1i*S21B_s*pi/180);
        S12_s = S12A_s.*exp(1i*S12B_s*pi/180);
        S22_s = S22A_s.*exp(1i*S22B_s*pi/180);
        clear S11A_s S11B_s S21A_s S21B_s S12A_s S12B_s S22A_s S22B_s;
    case 2
        S11 = S11A + 1i*S11B;
        S21 = S21A + 1i*S21B;
        S12 = S12A + 1i*S12B;
        S22 = S22A + 1i*S22B;
        clear S11A S11B S21A S21B S12A S12B S22A S22B;

        S11_o = S11A_o + 1i*S11B_o;
        S21_o = S21A_o + 1i*S21B_o;
        S12_o = S12A_o + 1i*S12B_o;
        S22_o = S22A_o + 1i*S22B_o;
        clear S11A_o S11B_o S21A_o S21B_o S12A_o S12B_o S22A_o S22B_o;

        S11_s = S11A_s + 1i*S11B_s;
```

```
        S21_s = S21A_s + 1i*S21B_s;
        S12_s = S12A_s + 1i*S12B_s;
        S22_s = S22A_s + 1i*S22B_s;
        clear S11A_s S11B_s S21A_s S21B_s S12A_s S12B_s S22A_s S22B_s;
    case 3
        S11 = 10.^(S11A/20).*exp(1i*S11B*pi/180);
        S21 = 10.^(S21A/20).*exp(1i*S21B*pi/180);
        S12 = 10.^(S12A/20).*exp(1i*S12B*pi/180);
        S22 = 10.^(S22A/20).*exp(1i*S22B*pi/180);
        clear S11A S11B S21A S21B S12A S12B S22A S22B;

        S11_o = 10.^(S11A_o/20).*exp(1i*S11B_o*pi/180);
        S21_o = 10.^(S21A_o/20).*exp(1i*S21B_o*pi/180);
        S12_o = 10.^(S12A_o/20).*exp(1i*S12B_o*pi/180);
        S22_o = 10.^(S22A_o/20).*exp(1i*S22B_o*pi/180);
        clear S11A_o S11B_o S21A_o S21B_o S12A_o S12B_o S22A_o S22B_o;

        S11_s = 10.^(S11A_s/20).*exp(1i*S11B_s*pi/180);
        S21_s = 10.^(S21A_s/20).*exp(1i*S21B_s*pi/180);
        S12_s = 10.^(S12A_s/20).*exp(1i*S12B_s*pi/180);
        S22_s = 10.^(S22A_s/20).*exp(1i*S22B_s*pi/180);
        clear S11A_s S11B_s S21A_s S21B_s S12A_s S12B_s S22A_s S22B_s;

    end

    Y11=zeros(1,length(freq));
    Y12=Y11;
    Y21=Y11;
    Y22=Y11;
    Y11_o=Y11;
    Y12_o=Y11;
    Y21_o=Y11;
    Y22_o=Y11;
    Y11_s=Y11;
    Y12_s=Y11;
    Y21_s=Y11;
    Y22_s=Y11;
    Z11_d=Y11;
    Z12_d=Y11;
    Z21_d=Y11;
    Z22_d=Y11;
    S11_d=Y11;
    S12_d=Y11;
    S21_d=Y11;
    S22_d=Y11;
    Zin=Y11;
    Z0=Y11;
    for index = 1:length(freq)
        y_params = s2y([S11(index) S12(index); S21(index)
S22(index)]);
        Y11(index) = y_params(1,1);
        Y21(index) = y_params(2,1);
```

```
        Y12(index) = y_params(1,2);
        Y22(index) = y_params(2,2);
        y_params_o = s2y([S11_o(index) S12_o(index); S21_o(index),
S22_o(index)]);
        Y11_o(index) = y_params_o(1,1);
        Y21_o(index) = y_params_o(2,1);
        Y12_o(index) = y_params_o(1,2);
        Y22_o(index) = y_params_o(2,2);
        y_params_s = s2y([S11_s(index) S12_s(index); S21_s(index),
S22_s(index)]);
        Y11_s(index) = y_params_s(1,1);
        Y21_s(index) = y_params_s(2,1);
        Y12_s(index) = y_params_s(1,2);
        Y22_s(index) = y_params_s(2,2);

        Y = [Y11(index) Y12(index); Y21(index) Y22(index)];
        Y_o = [Y11_o(index) Y12_o(index); Y21_o(index) Y22_o(index)];
        Y_s = [Y11_s(index) Y12_s(index); Y21_s(index) Y22_s(index)];

        Y_total_w_short = (Y - Y_o);
        Y_short_wo_open = (Y_s - Y_o);
        Z_total_w_short = y2z(Y_total_w_short);
        Z_short_wo_open = y2z(Y_short_wo_open);
        Z_dut = (Z_total_w_short - Z_short_wo_open);
        s_params = z2s(Z_dut);

        Z11_d(index)=Z_dut(1,1);
        Z12_d(index)=Z_dut(1,2);
        Z21_d(index)=Z_dut(2,1);
        Z22_d(index)=Z_dut(2,2);

        S11_d(index)=s_params(1,1);
        S12_d(index)=s_params(1,2);
        S21_d(index)=s_params(2,1);
        S22_d(index)=s_params(2,2);

        %Assumes the load is 50 ohm resistor.
        Zin(index)=Z11_d(index)-(Z12_d(index)-
Z21_d(index))/(Z22_d(index)-50);
        Z0(index)=Zin(index)/((1+S11_d(index))/(1-S11_d(index)));
    end
    if plotZ0==1,
        setFigSize;
        setPLOT;
        figure1 = figure('Position',...
            [scrsz(1)+xoffset scrsz(2)+yoffset xwidth yheight]);
%#ok<NASGU>
        s1 = subplot(2,1,1); %#ok<NASGU>
        title('Characteristic Impedance,
Z0','FontSize',36,'FontName','FixedWidth');
        box('on');
        hold('all');
```

```
        p1=plot(freq,smooth(abs(Z0),length(Z0)/10),strcat('-
','rs'),...
            'MarkerSize',10,'Color','r',...
            'MarkerFaceColor','k',...
            'LineWidth',2); %#ok<NASGU>

        set(gca,'FontSize',11,'XScale','log','YScale','log',...
                'GridLineStyle','none','XLim',[freq(1)
freq(length(freq))],'YLim',[10 100],...
                'MinorGridLineStyle','none',...

'FontSize',28,'XMinorGrid','on','FontName','FixedWidth');
        xlabel('Frequency
(Hz)','FontSize',32,'FontName','FixedWidth');
        ylabel('|Z0| (ohms)','FontSize',32,'FontName','FixedWidth');

        s2 = subplot(2,1,2); %#ok<NASGU>
        box('on');
        hold('all');
        p2=plot(freq,smooth(angle(Z0)*180/pi,length(Z0)/10),strcat('-
','rs'),...
            'MarkerSize',10,'Color','r',...
            'MarkerFaceColor','k',...
            'LineWidth',2); %#ok<NASGU>

        set(gca,'FontSize',11,'XScale','log','YScale','log',...
                'GridLineStyle','none','XLim',[freq(1)
freq(length(freq))],'YLim',[.001 1],...
                'MinorGridLineStyle','none',...

'FontSize',28,'XMinorGrid','on','FontName','FixedWidth');
        xlabel('Frequency
(Hz)','FontSize',32,'FontName','FixedWidth');
        ylabel('<Z0 (degrees)','FontSize',32,'FontName','FixedWidth');
    end
end
```

%%plotIDVG.m

```
% ID vs VG for various VD values
function plotIDVG(fileName,typeVEC,devID,dieNUM,TEMPlevel,TIDlevel)
    setPLOT;
    setFigSize;
    strTitle=char(fileName);

      runVEC=[3 11]; %Indexes of VD to Plot

    a1=importdata(strTitle);
%     a1.data(any(isnan(a1.data),2),:)=[]; %Only necessary to
eliminate extra spaces when saving *.csv to *.dat
    if typeVEC==1,
        vg1=a1.data(:,1);
        id1=a1.data(:,3);
        vd1=a1.data(:,2);
    elseif typeVEC==2,
        vg1=-a1.data(:,1);
        id1=-a1.data(:,3);
        vd1=a1.data(:,2).*-1;
    else
        display('This has not been coded yet');
    end

    if length(vg1)==1456,
        maxstep=16;
        vg=vg1(1:length(vg1)/maxstep);           %The vectors include
16 measurements (unique VD values) - currently hardcoded
        id=vec2mat(id1,length(id1)/maxstep)';
        vd=vec2mat(vd1,length(vd1)/maxstep)';
        vd=vd(1,:);
        gm=zeros(length(id)-1,maxstep);
        gms=gm;
        ids=zeros(length(id),maxstep);
    elseif length(vg1)==726,
        maxstep=11;
        vg=vg1(1:length(vg1)/maxstep);           %The vectors include
11 measurements (unique VD values) - currently hardcoded
        id=vec2mat(id1,length(id1)/maxstep)';
        vd=vec2mat(vd1,length(vd1)/maxstep)';
        vd=vd(1,:);
        gm=zeros(length(id)-1,maxstep);
        gms=gm;
        ids=zeros(length(id),maxstep);
    else
        display('Unrecognized length');
    end
    gmpeak=zeros(1,maxstep);
    vgpeakgm=zeros(1,maxstep);
    idpeakgm=zeros(1,maxstep);
    yint=zeros(1,maxstep);
```

```
    vt=zeros(1,maxstep);
    for i=1:maxstep
        ids(:,i)=smooth(id(:,i),'lowess',5);
        gm(:,i)=diff(id(:,i))./diff(vg)./1e-3;
        gms(:,i)=smooth(gm(:,i),'lowess',5);
        gmpeak(i)=max(gms(:,i));
        vgpeakgm(i)=vg(gms(:,i)==max(gms(:,i)));
        idpeakgm(i)=id(vg==vgpeakgm(i),i);
        yint(i)=idpeakgm(i)-gmpeak(i)*vgpeakgm(i)*1e-3;
        vt(i)=-yint(i)/(gmpeak(i).*1e-3);
    end
    vtsat=vt(maxstep);
    cvec=zeros(length(vd),3);
    fvec=zeros(length(vd),3);
    mvec=cell(length(vd),1);
    for i=1:16,
        cvec(i,:)=colorvec(round(rand(1)*6+1),:);       %#ok<NODEF>
%random color (7 choices)
        fvec(i,:)=colorvec(round(rand(1)*6+1),:);       %random face
color (7 choices)
        mvec{i}=char(markerVEC(round(rand(1)*6+1)));   %random marker
(7 choices)
    end
    clear a1 vg1 id1 ig1 vd1;

    % Plot Id-Vg (Linear)
    if isempty(findobj('type','figure'))==1,
        figure3 = figure('Position',...
            [scrsz(1)+xoffset scrsz(2)+yoffset xwidth yheight]);
        axes1 = axes('Parent',figure3); %#ok<*LAXES>
        box(axes1,'on');
        hold(axes1,'all')

        set(axes1,'XScale','linear','YScale','linear',...
            'GridLineStyle','none','XLim',[-0.3 max(vd)],'YLim',[0
20],...
            'MinorGridLineStyle','none',...
            'FontSize',28,'XMinorGrid','on','FontName','FixedWidth');
        if typeVEC==1,
            xlabel('V_{GS}
(V)','FontSize',32,'FontName','FixedWidth');
            ylabel('I_{DS}
(mA)','FontSize',32,'FontName','FixedWidth');
        elseif typeVEC==2,
            xlabel('V_{SG}
(V)','FontSize',32,'FontName','FixedWidth');
            ylabel('I_{SD}
(mA)','FontSize',32,'FontName','FixedWidth');
        end
    end

    strLegend=cell(1,length(runVEC));
```

```
    count=1;
    for i=runVEC,
%     for i=1:length(vd),
        plot(vg,ids(:,i)/1e-3,strcat('-',mvec{i}),...
            'MarkerSize',10,'Color',cvec(i,:),...
            'MarkerFaceColor',fvec(i,:),...
            'LineWidth',1);
        strLegend{1,count}=strcat(num2str(vd(1,i)));
        count=count+1;
    end


hleg1=legend(strLegend,'Location','NorthEastOutside','FontSize',26,...
        'FontName','FixedWidth');
        htitle1=get(hleg1,'Title');
        if typeVEC==1,
            set(htitle1,'String','V_{DS}
(V)','FontSize',28,'FontName','FixedWidth');
        elseif typeVEC==2,
            set(htitle1,'String','V_{SD}
(V)','FontSize',28,'FontName','FixedWidth');
        end
%     % Plot Id-Vg %%and Ig-Vg (Log)
%     figure3 = figure('Position',...
%         [scrsz(1)+xoffset scrsz(2)+yoffset xwidth yheight]);
%     axes3 = axes('Parent',figure3);
%     box(axes3,'on');
%     hold(axes3,'all')
%     for i=[3 7 11],
% %     for i=1:length(vd),
%         plot(vg,ids(:,i),strcat('-',mvec{i}),...
%             'MarkerSize',10,'Color',cvec(i,:),...
%             'MarkerFaceColor',fvec(i,:),...
%             'LineWidth',1);
%     end
%
hleg3=legend(strLegend,'Location','NorthEastOutside','FontSize',26,...
%         'FontName','FixedWidth');
%     htitle3=get(hleg3,'Title');
%     set(htitle3,'String','V_{DS}
(V)','FontSize',28,'FontName','FixedWidth');
%     set(axes3,'XScale','linear','YScale','log',...
%         'GridLineStyle','none','XLim',[-0.3 1],'YLim',[1e-9 100e-
3],...
%         'MinorGridLineStyle','none',...
%         'FontSize',28,'XMinorGrid','on','FontName','FixedWidth');
%     if typeVEC==1,
%         xlabel('V_{GS} (V)','FontSize',32,'FontName','FixedWidth');
%         ylabel('I_{DS} (A)','FontSize',32,'FontName','FixedWidth');
%     elseif typeVEC==2,
%         xlabel('V_{SG} (V)','FontSize',32,'FontName','FixedWidth');
%         ylabel('I_{SD} (A)','FontSize',32,'FontName','FixedWidth');
```

```
%     end

    peakID=max(max(id))/1e-3;
    peakGM=max(max(gm));
    peakIDL1=max(id(vg==-.3,:))/1e-9;
    peakIDL2=max(id(vg==0,:))/1e-9;

    fid = fopen('compiled.txt', 'a');
%     fprintf(fid,'%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\n',...
%             'Device','TID','Temp','Peak ID (mA)','peak gm
(mS)','IDleak@-0.3 (nA)','IDleak@0 (nA)','Vtsat (V)');
    fprintf(fid,'%s\t%s\t%s\t%6.1f\t%6.1f\t%6.1f\t%6.1f\t%6.3f\n',...

char(strcat(dieNUM,devID)),char(TIDlevel),char(TEMPlevel),[peakID
peakGM peakIDL1 peakIDL2 vtsat]);
    fclose(fid);
end
```

%%setFigSiz.m

```
scrsz = get(0,'ScreenSize');
xoffset=50;
yoffset=75;
% xwidth=800;
% yheight=600;
xwidth=1400;
yheight=800;
```

%%setGM.m

```
function gmfile=setGM(dirname,dcfile,typeVEC)
    fileName=strcat(dirname,dcfile);
    strTitle=char(fileName);
    a1=importdata(strTitle);
%     a1.data(any(isnan(a1.data),2),:)=[];  %Only necessary to
eliminate extra spaces when saving *.csv to *.dat
    if typeVEC==1,
        vg1=a1.data(:,1);
        id1=a1.data(:,3);
        vd1=a1.data(:,2);
    elseif typeVEC==2,
        vg1=a1.data(:,1)+1;
        id1=a1.data(:,3);
        vd1=a1.data(:,2).*-1;
    else
        display('This has not been coded yet');
    end

    if length(vg1)==1456,
        maxstep=16;
        vg=vg1(1:length(vg1)/maxstep);           %The vectors include
16 measurements (unique VD values) - currently hardcoded
        id=vec2mat(id1,length(id1)/maxstep)';
        vd=vec2mat(vd1,length(vd1)/maxstep)';
        vd=vd(1,:);
        gm=zeros(length(id)-1,maxstep);
        gms=gm;
        ids=zeros(length(id),maxstep);
    elseif length(vg1)==726,
        maxstep=11;
        vg=vg1(1:length(vg1)/maxstep);           %The vectors include
11 measurements (unique VD values) - currently hardcoded
        id=vec2mat(id1,length(id1)/maxstep)';
        vd=vec2mat(vd1,length(vd1)/maxstep)';
        vd=vd(1,:);
        gm=zeros(length(id)-1,maxstep);
        gms=gm;
        ids=zeros(length(id),maxstep);
    else
        display('Unrecognized length');
    end
    gmpeak=zeros(1,maxstep);
    vgpeakgm=zeros(1,maxstep);
    idpeakgm=zeros(1,maxstep);
    yint=zeros(1,maxstep);
    vt=zeros(1,maxstep);
    for i=1:maxstep
        ids(:,i)=smooth(id(:,i),'lowess',5);
        gm(:,i)=diff(id(:,i))./diff(vg)./1e-3;
        gms(:,i)=smooth(gm(:,i),'lowess',5);
```

```
        gmpeak(i)=max(gms(:,i));
        vgpeakgm(i)=vg(gms(:,i)==max(gms(:,i)));
        idpeakgm(i)=id(vg==vgpeakgm(i),i);
        yint(i)=idpeakgm(i)-gmpeak(i)*vgpeakgm(i)*1e-3;
        vt(i)=-yint(i)/(gmpeak(i).*1e-3);
    end
    clear a1 vg1 id1 ig1 vd1;

    gmmat1=[vd;gm];
    gmmat=[[0;vg(2:length(vg))] gmmat1];
     gmfile=strcat(dirname,'GM.csv');
    csvwrite(char(gmfile),gmmat);
end
```

%%setPlot.m

```
%% Plotting parameters
colorvec=zeros(7,3);
colorvec(1,:)=[1 0 0];
colorvec(2,:)=[.04 .14 .42];
colorvec(3,:)=[0 .5 0];
colorvec(4,:)=[0 0 0];
colorvec(5,:)=[1 .5 0];
colorvec(6,:)=[.42 .1 0];
colorvec(7,:)=[.5 .5 .6];

markerVEC=cell(1,7);
markerVEC{1}='o';
markerVEC{2}='^';
markerVEC{3}='V';
markerVEC{4}='s';
markerVEC{5}='d';
markerVEC{6}='X';
markerVEC{7}='*';
```

4156_IdVg.py

```
#!/usr/bin/env python
import time,math,sys,os,numpy,threading
import vxi_11

import cPickle

class e5810(vxi_11.vxi_11_connection):
    def abort(self):
        pass

ip = "129.59.93.172" #ip = "129.59.73.80"
# 3. Establish comm with serial
pos_gpib=e5810(host=ip, device="gpib0,17", raise_on_err=1,
timeout=60000,device_name="HP 4156A") # if this stops working check
device = parameter
pos_gpib.write("*rst")# Sets the 4156 to its initial settings at the
time of measurement
#print "Configuration...\n"
pos_gpib.write(":FORM:DATA ASCii")# Here i define the data format as
real-manual pp. 1-35
## Here I define all SMU settings of the Channel definition page
pos_gpib.write(":PAGE:CHAN:CDEF")# sets Channel denfiniton page
pos_gpib.write(":PAGE:CHAN:MODE SWE")  #sets measurement mode as sweep
#time.sleep(4)
pos_gpib.write(":PAGE:CHAN:SMU2:VNAME 'VG'")# sets VNAME of SMU1
pos_gpib.write(":PAGE:CHAN:SMU2:FUNC VAR1")# sets SMU1 as VAR1
pos_gpib.write(":PAGE:CHAN:SMU2:INAME 'IG'")# sets INAME of SMU1
pos_gpib.write(":PAGE:CHAN:SMU2:MODE V")# sets mode of SMU1 as V

pos_gpib.write(":PAGE:CHAN:SMU1:VNAME 'VD'")# sets VNAME of SMU2
pos_gpib.write(":PAGE:CHAN:SMU1:FUNC CONS")# sets SMU2 as CONSTANT
pos_gpib.write(":PAGE:CHAN:SMU1:INAME 'ID'")# sets INAME of SMU2
pos_gpib.write(":PAGE:CHAN:SMU1:MODE V")# sets mode of SMU2 as V

pos_gpib.write(":PAGE:CHAN:SMU3:VNAME 'VS'")# sets VNAME of SMU3
pos_gpib.write(":PAGE:CHAN:SMU3:FUNC CONS")# sets SMU3 as constant
pos_gpib.write(":PAGE:CHAN:SMU3:INAME 'IS'")# sets INAME of SMU3
pos_gpib.write(":PAGE:CHAN:SMU3:MODE V")# sets mode of SMU3 as Common

pos_gpib.write(":PAGE:CHAN:SMU4:VNAME 'VSUB'") # sets VNAME of SMU4
pos_gpib.write(":PAGE:CHAN:SMU4:FUNC CONS")  # sets SMU4 as constant
pos_gpib.write(":PAGE:CHAN:SMU4:INAME 'ISUB'") # sets INAME of SMU4
pos_gpib.write(":PAGE:CHAN:SMU4:MODE V")     # sets mode of SMU4 as
Common

pos_gpib.write(":PAGE:CHAN:VSU1:DIS")# Disables VSU1
pos_gpib.write(":PAGE:CHAN:VSU2:DIS")
pos_gpib.write(":PAGE:CHAN:VMU1:DIS")
pos_gpib.write(":PAGE:CHAN:VMU2:DIS")
#pos_gpib.write(":PAGE:CHAN:SMU3:DIS")
```

```
#time.sleep(15)

#print "Define transconductance...\n"
## Here is define the user function (transconductance (gm), and
threshold voltage (Vt) in here)
#pos_gpib.write(":PAGE:CHAN:UFUN:DEF 'GM', 'S', 'DIFF(ID,VG)'")  #
Here I define the transconductance function
#pos_gpib.write(":PAGE:CHAN:UFUN:DEF 'VTH', 'V', '@L1X'")  # Threshold
voltage; got this idea from the 4156 programming manual
#time.sleep(10)

## Here I make the integration time Medium
pos_gpib.write(":PAGE:MEAS:MSET:ITIM SHORT")# Integration time medium

#time.sleep(3)
#print "Setup measurement display...\n"
## Now the Measure setup page should display
#pos_gpib.write(":PAGE:MEAS:MSET")
pos_gpib.write(":PAGE:MEAS:CONS:SMU1 0.1")
pos_gpib.write(":PAGE:MEAS:CONS:SMU1:COMP 10mA")
pos_gpib.write(":PAGE:MEAS:VAR1: MODE SING")# Here I set the sweep
mode for VAR1-manual pp. 1-232
pos_gpib.write(":PAGE:MEAS:VAR1:SPAC LIN")# Here i set the sweep type
of VAR1-manual pp. 1-235
pos_gpib.write(":PAGE:MEAS:VAR1:STAR -0.1")# Here I set the starting
value of sweep- manual pp. 1-236
pos_gpib.write(":PAGE:MEAS:VAR1:STEP 0.01")# Here I set the step size
of Vg sweep- manual pp. 1-237
pos_gpib.write(":PAGE:MEAS:VAR1:STOP 1.5")# Here I set the stop value
of Vg -manual pp. 1-238
pos_gpib.write(":PAGE:MEAS:VAR1:COMP 1mA")# Here i set the compliance
of VAR1-manual pp. 1-232
pos_gpib.write(":PAGE:MEAS:CONS:SMU3 0")# Here I set the source (SMU3)
at Vd=0V
pos_gpib.write(":PAGE:MEAS:CONS:SMU3:COMP 10mA")
pos_gpib.write(":PAGE:MEAS:CONS:SMU4 0")
pos_gpib.write(":PAGE:MEAS:CONS:SMU4:COMP 10mA")

#time.sleep(15)

#VGrange=[0.5,-1.2,-0.01]
## Now the Display page should show up
#pos_gpib.write(":PAGE:DISP:ANAL")
pos_gpib.write(":PAGE:DISP:GRAP:GRID ON")# turns the grid on
pos_gpib.write(":PAGE:DISP:GRAP:X: SCALE LIN")# This sets the x, Y1,
Y2 axis as linear, log and log respectively
pos_gpib.write(":PAGE:DISP:GRAP:Y1:SCALE LOG")
pos_gpib.write(":PAGE:DISP:GRAP:Y2:SCALE LOG")

pos_gpib.write(":PAGE:DISP:GRAP:X: NAME 'VG'")# This specifies the
name of x, Y1, Y2 axis as Vg, IS, ID respectively
```

```
pos_gpib.write(":PAGE:DISP:GRAP:Y1:NAME 'ID'")
pos_gpib.write(":PAGE:DISP:GRAP:Y2:NAME 'IG'")

pos_gpib.write(":PAGE:DISP:GRAP:X:MIN -0.1")# This specifies the
minimum value of X, Y1, Y2 axis-manual pp. 1-149
pos_gpib.write(":PAGE:DISP:GRAP:Y1:MIN 0")
pos_gpib.write(":PAGE:DISP:GRAP:Y2:MIN 0")

pos_gpib.write(":PAGE:DISP:GRAP:X:MAX 1.5")# This specifies the
maximum value of X, Y1, Y2 axis-manual pp. 1-148
pos_gpib.write(":PAGE:DISP:GRAP:Y1:MAX 100E-6")
pos_gpib.write(":PAGE:DISP:GRAP:Y2:MAX 0.0001")

pos_gpib.write(":PAGE:DISP:LIST 'VD','VG','IG','ID','IS'")# This makes
a list of the data saved pp. 1-153
#time.sleep(15)
#pos_gpib.write(":PAGE:GLIS:SCAL:AUTO ONCE")
# This autoscales the graphics page -manual pp. 1-168


#print "Start a single measurement...\n"
## Now lets start the single measurement
pos_gpib.write(":PAGE:SCON:SING")# This is equivalent to pressing the
single button on paramter analyzer-manual pp. 1-250
#time.sleep(6)
pos_gpib.write(":PAGE:GLIS:GRAP:SCAL:AUTO ONCE")
# This autoscales the graphics page -manual pp. 1-168--WHICH AXIS??

#print "Get data catalog...\n"
#time.sleep(5)
pos_gpib.write("*OPC?")
pos_gpib.write(":PAGE:GLIS:GRAP:SCAL:AUTO ONCE")# This autoscales the
graphics page -manual pp. 1-168--WHICH AXIS??
#time.sleep(40)                                  # Just making sure
that program does not exit before the data starts saving --10/17/08

#pos_gpib.write(":PAGE:MEAS:SWE:VAR1:STAR?")     # Queries for start
of measurement setup (Vg) data - 4156 prog manual-pp. 2-19

#pos_gpib.write(":DATA:TRAC:CAT?")

pos_gpib.write(":DATA? 'VG''ID''IG''IS''ISUB''VD'")
# Writes the VG data on output buffer
VG=pos_gpib.read()
ID=pos_gpib.read()
IG=pos_gpib.read()
IS=pos_gpib.read()
ISUB=pos_gpib.read()
VD=pos_gpib.read()
#VG=VG[2:]
#VG=VG[2:]
```

```
f = open("VGID-IdVg.txt","w")                        # open file for
writing
for vg1,id1,ig1,is1,isub1,vd1 in zip(VG,ID,IG,IS,ISUB,VD):
    f.write("%s %s %s %s %s %s\n" %(vg1,id1,ig1,is1,isub1,vd1))
f.close()

pos_gpib.write(":DATA? 'VD'")                         # Writes the VG data
on output buffer
VD=pos_gpib.read()                                   # Writes VG on GPIB
VD=VD[2:]
f = open("VDrain-IdVg.txt","w")
for vd1 in zip(VD):
       f.write("%2s\n" %(vd1))
       f.close()

 #f.write("%2s" %(VD))
#f.close()

pos_gpib.write(":DATA? 'IS'")
IS=pos_gpib.read()
IS=IS[2:]
f = open("ISource-IdVg.txt","w")
f.write("%2s" %(IS))
f.close()

pos_gpib.write(":DATA? 'ID'")
ID=pos_gpib.read()
ID=ID[2:]
f = open("IDrain-IdVg.txt","w")
f.write("%2s" %(ID))
f.close()

pos_gpib.write(":DATA? 'IG'")
IG=pos_gpib.read()
IG=IG[2:]
f = open("IGate-IdVg.txt", "w")
f.write("%2s" %(IG))
f.close()

pos_gpib.write(":DATA? 'ISUB'")
ISUB=pos_gpib.read()
ISUB=ISUB[2:]
f = open("ISUB-IdVg.txt", "w")
f.write("%2s" %(ISUB))
f.close()
```

pna_meas.py

```
#!/usr/bin/python


#3521

import time
import struct
import sys
import data_acquisition2.vxi_11 as vxi_11

execfile('ADDR')

class e5810(vxi_11.vxi_11_connection):
    def abort(self):
      pass

def wait(smu):
    smu.write("*OPC?")
    while True:
      val=smu.read()[2]
      try:
          if int(float(val))==1:
            break
      except:
          pass

def stress_begin(vg=0.0,vd=0.0,gate=1,drain=2,source=3,substrate=4):
    #######SETTING UP CHANNEL PAGE#######
    #delete all initial settings
    hp4156.write(":page:chan:all:dis")
    hp4156.write(":page:chan:ufun:del:all")
    hp4156.write(":page:chan:uvar:del:all")
    #set to sampling mode
    hp4156.write(":page:stress:all:disable")
    hp4156.write(":page:stress:smu%s:mode v" % gate)
    hp4156.write(":page:stress:smu%s:mode v" % drain)
    hp4156.write(":page:stress:smu%s:mode common" % source)
    hp4156.write(":page:stress:smu%s:mode common" % substrate)

    hp4156.write(":page:stress:smu%s:func sync" % gate)
    hp4156.write(":page:stress:smu%s:func sync" % drain)
    hp4156.write(":page:stress:smu%s:func nsync" % source)
    hp4156.write(":page:stress:smu%s:func nsync" % substrate)

    hp4156.write(":page:stress:setup:cons:smu%s:source %s" %
(gate,vg))
    hp4156.write(":page:stress:setup:cons:smu%s:source %s" %
(drain,vd))
    hp4156.write(":page:stress:setup:duration 0")
    hp4156.write("*opc?")
```

```
    a=hp4156.read()
    hp4156.write(":page:scon:stress")

def stress_end():
    hp4156.write(":page:scon:stop")



def idvd(gate=1,drain=2,source=3,substrate=4,vdmax=1.0,vdmin=-0.3):
    #######SETTING UP CHANNEL PAGE#######
    #delete all initial settings
    hp4156.write(":page:chan:all:dis")
    hp4156.write(":page:chan:ufun:del:all")
    hp4156.write(":page:chan:uvar:del:all")
    #set to sampling mode
    hp4156.write(":page:chan:mode swe")
    #set up SMU's
    hp4156.write("page:chan:smu%s:vname 'VG'" % gate)
    hp4156.write("page:chan:smu%s:iname 'IG'"% gate)
    hp4156.write(":page:chan:smu%s:mode v"% gate)
    hp4156.write(":page:chan:smu%s:func var2"% gate)

    hp4156.write("page:chan:smu%s:vname 'VD'"% drain)
    hp4156.write("page:chan:smu%s:iname 'ID'"% drain)
    hp4156.write(":page:chan:smu%s:mode v"% drain)
    hp4156.write(":page:chan:smu%s:func var1"% drain)

    hp4156.write("page:chan:smu%s:vname 'VS'"% source)
    hp4156.write("page:chan:smu%s:iname 'IS'"% source)
    hp4156.write(":page:chan:smu%s:mode common"% source)
    hp4156.write(":page:chan:smu%s:func cons"% source)

    hp4156.write("page:chan:smu%s:vname 'VSUB'"% substrate)
    hp4156.write("page:chan:smu%s:iname 'ISUB'"% substrate)
    hp4156.write(":page:chan:smu%s:mode v"% substrate)
    hp4156.write(":page:chan:smu%s:func cons"% substrate)

    #######SETTING UP MEASUREMENT PAGE#######

    #set sampling
    hp4156.write("page:meas:mset:itim med")

    # substrate bias
    hp4156.write(":page:meas:cons:SMU4 0.0")

    hp4156.write(":page:meas:swe:var1:mode sing")
    hp4156.write(":page:meas:swe:var1:spac lin ")
    hp4156.write(":page:meas:swe:var1:star %s " % vdmin)
    hp4156.write(":page:meas:swe:var1:step 0.02 ")
    hp4156.write(":page:meas:swe:var1:stop %s" % vdmax)
    hp4156.write(":page:meas:swe:var1:comp 0.05")
```

```
    hp4156.write(":page:meas:swe:var2:mode sing")
    hp4156.write(":page:meas:swe:var2:spac lin ")
    hp4156.write(":page:meas:swe:var2:star 0.0 ")
    hp4156.write(":page:meas:swe:var2:step 0.1 ")
    hp4156.write(":page:meas:swe:var2:poin 11")
    hp4156.write(":page:meas:swe:var2:comp 0.0001")

    #######SETTING UP DISPLAY PAGE#######

    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG:MODE fixed")
    hp4156.write(":PAGE:MEAS:MSET:SMU2:RANG:MODE limited")
    hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG:MODE limited")
    hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG:MODE limited")
    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG 1e-3")
    hp4156.write(":PAGE:MEAS:MSET:SMU2:RANG 1e-8")
    hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG 1e-8")
    hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG 1e-8")

    ###Graphics
    hp4156.write(":page:disp:set:grap:x:del")
    hp4156.write(":page:disp:set:grap:y1:del")
    hp4156.write(":page:disp:set:grap:y2:del")
    hp4156.write(":page:disp:set:dvar:del:all")
    #
    hp4156.write(":page:disp:set:grap:x:name 'VD'")
    hp4156.write(":page:disp:set:grap:x:scal lin")
    #
    hp4156.write(":page:disp:set:grap:y1:name 'ID'")
    hp4156.write(":page:disp:set:grap:y1:scal lin")
    hp4156.write(":page:disp:set:grap:y1:min -1mA")
    hp4156.write(":page:disp:set:grap:y1:max 3mA")

    ##List
    hp4156.write(":page:disp:set:list:del:all")
    #hp4156.write(":page:disp:set:list:sel
'VG','VD','ID','IG','ISUB','IS','VSUB'")
    hp4156.write(":page:disp:set:list:sel 'VG','VD','ID'")

    hp4156.write("*opc?")
    hp4156.read()

    hp4156.write(":page:scon:sing")
    wait(hp4156)
    outputs=[]

    for par in ['VG','VD','ID','VS']:
      hp4156.write('trac? %s' % par)
      val=hp4156.read()[2].strip().split(',')
      tmp=[]
      tmp.append(par)
      for i in val:
          tmp.append(i)
```

```
      outputs.append(tmp)

    # print outputs
    tabdata=""
    for j in range(len(outputs[0])):
      for i in range(len(outputs)):
          tabdata+="%s " % outputs[i][j]
      tabdata+="\n"
    return tabdata



def
idvg(gate=1,drain=2,source=3,substrate=4,dbias=0.050,vdmax=1.0,vdmin=-
0.3):
    integrationtime="short"
    #######SETTING UP CHANNEL PAGE#######
    #delete all initial settings
    hp4156.write(":page:chan:all:dis")
    hp4156.write(":page:chan:ufun:del:all")
    hp4156.write(":page:chan:uvar:del:all")

    #set to sampling mode
    hp4156.write(":page:chan:mode swe")

    #set up SMU's
    hp4156.write("page:chan:smu%s:vname 'VG'" % gate)
    hp4156.write("page:chan:smu%s:iname 'IG'" % gate)
    hp4156.write(":page:chan:smu%s:mode v" % gate)
    hp4156.write(":page:chan:smu%s:func var1" % gate)

    hp4156.write("page:chan:smu%s:vname 'VD'" % drain)
    hp4156.write("page:chan:smu%s:iname 'ID'" % drain)
    hp4156.write(":page:chan:smu%s:mode v" % drain)
    hp4156.write(":page:chan:smu%s:func var2" % drain)

    hp4156.write("page:chan:smu%s:vname 'VS'" % source)
    hp4156.write("page:chan:smu%s:iname 'IS'" % source)
    hp4156.write(":page:chan:smu%s:mode common" % source)
    hp4156.write(":page:chan:smu%s:func cons" % source)

    hp4156.write("page:chan:smu%s:vname 'VSUB'" % substrate)
    hp4156.write("page:chan:smu%s:iname 'ISUB'" % substrate)
    hp4156.write(":page:chan:smu%s:mode common" % substrate)
    hp4156.write(":page:chan:smu%s:func cons" % substrate)

    #######SETTING UP MEASUREMENT PAGE#######

    #set sampling
    hp4156.write("page:meas:mset:itim med")
```

```
    # VDS
    #hp4156.write(":page:meas:cons:SMU%s %s" % (drain,dbias))

    hp4156.write(":page:meas:swe:var1:mode sing")
    hp4156.write(":page:meas:swe:var1:spac lin ")
    hp4156.write(":page:meas:swe:var1:star %s" % vdmin)
    hp4156.write(":page:meas:swe:var1:step 0.02 ")
    hp4156.write(":page:meas:swe:var1:stop %s" % vdmax)
    hp4156.write(":page:meas:swe:var1:comp .1")

    hp4156.write(":page:meas:swe:var2:mode sing")
    hp4156.write(":page:meas:swe:var2:spac lin ")
    hp4156.write(":page:meas:swe:var2:star 0.0 ")
    hp4156.write(":page:meas:swe:var2:step 0.1 ")
    hp4156.write(":page:meas:swe:var2:poin 11")
    hp4156.write(":page:meas:swe:var2:comp 0.1")



    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG:MODE fixed")
    hp4156.write(":PAGE:MEAS:MSET:SMU2:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG 1e-3")
    #hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG 1e-8")
    #hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG 1e-8")

    #######SETTING UP DISPLAY PAGE#######

    ##Graphics
    hp4156.write(":page:disp:set:grap:x:del")
    hp4156.write(":page:disp:set:grap:y1:del")
    hp4156.write(":page:disp:set:grap:y2:del")
    hp4156.write(":page:disp:set:dvar:del:all")
    #
    hp4156.write(":page:disp:set:grap:x:name 'VG")
    hp4156.write(":page:disp:set:grap:x:scal lin")
    #
    hp4156.write(":page:disp:set:grap:y1:name 'ID'")
    hp4156.write(":page:disp:set:grap:y1:scal log")
    hp4156.write(":page:disp:set:grap:y1:min 1pA")
    hp4156.write(":page:disp:set:grap:y1:max 10mA")

    ##List

    hp4156.write(":page:disp:set:list:del:all")
    #hp4156.write(":page:disp:set:list:sel
'VG','VD','ID','IG','ISUB','IS','VSUB'")
    hp4156.write(":page:disp:set:list:sel 'VG','VD','ID'")

    hp4156.write("*opc?")
    hp4156.read()
```

```
    hp4156.write(":page:scon:sing")
    wait(hp4156)
    outputs=[]

    for par in ['VG','VD','ID','VS']:
      hp4156.write('trac? %s' % par)
      val=hp4156.read()[2].strip().split(',')
      tmp=[]
      tmp.append(par)
      for i in val:
          tmp.append(i)

      outputs.append(tmp)

    # print outputs
    tabdata=""
    for j in range(len(outputs[0])):
      for i in range(len(outputs)):
          tabdata+="%s " % outputs[i][j]
      tabdata+="\n"
    return tabdata


def
idvg_single(gate=1,drain=2,source=3,substrate=4,dbias=0.050,vdmax=1.0,
vdmin=-0.3):
    integrationtime="short"
    #######SETTING UP CHANNEL PAGE#######
    #delete all initial settings
    hp4156.write(":page:chan:all:dis")
    hp4156.write(":page:chan:ufun:del:all")
    hp4156.write(":page:chan:uvar:del:all")

    #set to sampling mode
    hp4156.write(":page:chan:mode swe")

    #set up SMU's
    hp4156.write("page:chan:smu%s:vname 'VG'" % gate)
    hp4156.write("page:chan:smu%s:iname 'IG'" % gate)
    hp4156.write(":page:chan:smu%s:mode v" % gate)
    hp4156.write(":page:chan:smu%s:func var1" % gate)

    hp4156.write("page:chan:smu%s:vname 'VD'" % drain)
    hp4156.write("page:chan:smu%s:iname 'ID'" % drain)
    hp4156.write(":page:chan:smu%s:mode v" % drain)
    hp4156.write(":page:chan:smu%s:func cons" % drain)

    hp4156.write("page:chan:smu%s:vname 'VS'" % source)
    hp4156.write("page:chan:smu%s:iname 'IS'" % source)
    hp4156.write(":page:chan:smu%s:mode common" % source)
```

```
    hp4156.write(":page:chan:smu%s:func cons" % source)

    hp4156.write("page:chan:smu%s:vname 'VSUB'" % substrate)
    hp4156.write("page:chan:smu%s:iname 'ISUB'" % substrate)
    hp4156.write(":page:chan:smu%s:mode common" % substrate)
    hp4156.write(":page:chan:smu%s:func cons" % substrate)

    #######SETTING UP MEASUREMENT PAGE#######

    #set sampling
    hp4156.write("page:meas:mset:itim med")

    # VDS
    hp4156.write(":page:meas:cons:SMU%s %s" % (drain,dbias))

    hp4156.write(":page:meas:swe:var1:mode sing")
    hp4156.write(":page:meas:swe:var1:spac lin ")
    hp4156.write(":page:meas:swe:var1:star %s" % vdmin)
    hp4156.write(":page:meas:swe:var1:step 0.02 ")
    hp4156.write(":page:meas:swe:var1:stop %s" % vdmax)
    hp4156.write(":page:meas:swe:var1:comp .1")

    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG:MODE fixed")
    hp4156.write(":PAGE:MEAS:MSET:SMU2:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG 1e-3")
    #hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG 1e-8")
    #hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG 1e-8")

    #######SETTING UP DISPLAY PAGE#######

    ##Graphics
    hp4156.write(":page:disp:set:grap:x:del")
    hp4156.write(":page:disp:set:grap:y1:del")
    hp4156.write(":page:disp:set:grap:y2:del")
    hp4156.write(":page:disp:set:dvar:del:all")
    #
    hp4156.write(":page:disp:set:grap:x:name 'VG")
    hp4156.write(":page:disp:set:grap:x:scal lin")
    #
    hp4156.write(":page:disp:set:grap:y1:name 'ID'")
    hp4156.write(":page:disp:set:grap:y1:scal log")
    hp4156.write(":page:disp:set:grap:y1:min 1pA")
    hp4156.write(":page:disp:set:grap:y1:max 10mA")

    ##List

    hp4156.write(":page:disp:set:list:del:all")
    #hp4156.write(":page:disp:set:list:sel
'VG','VD','ID','IG','ISUB','IS','VSUB'")
    hp4156.write(":page:disp:set:list:sel 'VG','VD','ID'")
```

```
    hp4156.write("*opc?")
    hp4156.read()


    hp4156.write(":page:scon:sing")
    wait(hp4156)
    outputs=[]

    for par in ['VG','VD','ID','VS']:
      hp4156.write('trac? %s' % par)
      val=hp4156.read()[2].strip().split(',')
      tmp=[]
      tmp.append(par)
      for i in val:
          tmp.append(i)

      outputs.append(tmp)

    # print outputs
    tabdata=""
    for j in range(len(outputs[0])):
      for i in range(len(outputs)):
          tabdata+="%s " % outputs[i][j]
      tabdata+="\n"
    return tabdata



hp4156=e5810(host=ADDR, device="gpib0,17", raise_on_err=0,
timeout=5000,device_name="HP4156A")
PNA=e5810(host=ADDR, device="gpib0,16", raise_on_err=0,
timeout=5000,device_name="PNA")



def read_sdata():
    x=[];s11=[];s21=[];s12=[];s22=[]

    for ch in ("CH1_S11_1","CH1_S12_2","CH1_S21_3","CH1_S22_4"):
      if ch=="CH1_S11_1":
          sparam=s11
      if ch=="CH1_S12_2":
          sparam=s12
      if ch=="CH1_S21_3":
          sparam=s21
      if ch=="CH1_S22_4":
          sparam=s22

      PNA.write(':CALC:PAR:SEL "%s"' % ch)
```

```
      PNA.write(":CALC:DATA? SDATA")
      val=PNA.read()[2].strip().split(',')


      for i in range(0,len(val),2):
          sparam.append((float(val[i]),float(val[i+1])))

    PNA.write(":CALC:X?")
    xval=PNA.read()[2].strip().split(',')
    for xp in xval:
      x.append(float(xp))

    return((x,s11,s12,s21,s22))


topdir='500krad_Die14_DeviceA_xrayoff'

a=open("%s\IDVG_pre.dat" % topdir,'w')
a.write("%s" % idvg())
a.close()

a=open("%s\IDVD_pre.dat" % topdir,'w')
a.write("%s" % idvd())
a.close()


for vd in [0.3,0.6,1.0]:
    for vg in [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0]:
      print "Gate Voltage: %s    Drain Voltage: %s" % (vg,vd)
      stress_begin(vg=vg,vd=vd)
      time.sleep(0.5)
      x,s11,s12,s21,s22=read_sdata()
      stress_end()
      time.sleep(0.1)

      #print len(x),len(s11),len(s12),len(s21),len(s22)


      filename="%s\Vg%03imvVd%03i.s2p" % (topdir,vg*1000,vd*1000)
      print filename
      outfile=open(filename,'w')
      for f,p1,p2,p3,p4 in zip(x,s11,s12,s21,s22):
          outfile.write('%s %s %s %s %s %s %s %s %s\n' %
(f,p1[0],p1[1],p2[0],p2[1],p3[0],p3[1],p4[0],p4[1]))
      outfile.close()

a=open("%s\IDVG_post.dat" % topdir,'w')
a.write("%s" % idvg())
a.close()
```

```
## pna_meas_stress.py

#!/usr/bin/python


#3521

import time
import struct
import sys
import data_acquisition2.vxi_11 as vxi_11

execfile('ADDR')

class e5810(vxi_11.vxi_11_connection):
    def abort(self):
      pass

def wait(smu):
    smu.write("*OPC?")
    while True:
      val=smu.read()[2]
      try:
          if int(float(val))==1:
            break
      except:
          pass

def stress_begin(vg=0.0,vd=0.0,gate=1,drain=2,source=3,substrate=4):
    #######SETTING UP CHANNEL PAGE#######
    #delete all initial settings
    hp4156.write(":page:chan:all:dis")
    hp4156.write(":page:chan:ufun:del:all")
    hp4156.write(":page:chan:uvar:del:all")
    #set to sampling mode
    hp4156.write(":page:stress:all:disable")
    hp4156.write(":page:stress:smu%s:mode v" % gate)
    hp4156.write(":page:stress:smu%s:mode v" % drain)
    hp4156.write(":page:stress:smu%s:mode common" % source)
    hp4156.write(":page:stress:smu%s:mode common" % substrate)

    hp4156.write(":page:stress:smu%s:func sync" % gate)
    hp4156.write(":page:stress:smu%s:func sync" % drain)
    hp4156.write(":page:stress:smu%s:func nsync" % source)
    hp4156.write(":page:stress:smu%s:func nsync" % substrate)

    hp4156.write(":page:stress:setup:cons:smu%s:source %s" %
(gate,vg))
    hp4156.write(":page:stress:setup:cons:smu%s:source %s" %
(drain,vd))
    hp4156.write(":page:stress:setup:duration 0")
    hp4156.write("*opc?")
```

```
    a=hp4156.read()
    hp4156.write(":page:scon:stress")

def stress_end():
    hp4156.write(":page:scon:stop")
    hp4156.write(":page:scon:stop")
    hp4156.write(":page:scon:stop")


def idvd(gate=1,drain=2,source=3,substrate=4,vdmax=1.0,vdmin=-0.3):
    #######SETTING UP CHANNEL PAGE#######
    #delete all initial settings
    hp4156.write(":page:chan:all:dis")
    hp4156.write(":page:chan:ufun:del:all")
    hp4156.write(":page:chan:uvar:del:all")
    #set to sampling mode
    hp4156.write(":page:chan:mode swe")
    #set up SMU's
    hp4156.write("page:chan:smu%s:vname 'VG'" % gate)
    hp4156.write("page:chan:smu%s:iname 'IG'"% gate)
    hp4156.write(":page:chan:smu%s:mode v"% gate)
    hp4156.write(":page:chan:smu%s:func var2"% gate)

    hp4156.write("page:chan:smu%s:vname 'VD'"% drain)
    hp4156.write("page:chan:smu%s:iname 'ID'"% drain)
    hp4156.write(":page:chan:smu%s:mode v"% drain)
    hp4156.write(":page:chan:smu%s:func var1"% drain)

    hp4156.write("page:chan:smu%s:vname 'VS'"% source)
    hp4156.write("page:chan:smu%s:iname 'IS'"% source)
    hp4156.write(":page:chan:smu%s:mode common"% source)
    hp4156.write(":page:chan:smu%s:func cons"% source)

    hp4156.write("page:chan:smu%s:vname 'VSUB'"% substrate)
    hp4156.write("page:chan:smu%s:iname 'ISUB'"% substrate)
    hp4156.write(":page:chan:smu%s:mode v"% substrate)
    hp4156.write(":page:chan:smu%s:func cons"% substrate)

    #######SETTING UP MEASUREMENT PAGE#######

    #set sampling
    hp4156.write("page:meas:mset:itim med")

    # substrate bias
    hp4156.write(":page:meas:cons:SMU4 0.0")

    hp4156.write(":page:meas:swe:var1:mode sing")
    hp4156.write(":page:meas:swe:var1:spac lin ")
    hp4156.write(":page:meas:swe:var1:star %s " % vdmin)
    hp4156.write(":page:meas:swe:var1:step 0.02 ")   # ALSSS 0.02
    hp4156.write(":page:meas:swe:var1:stop %s" % vdmax)
    hp4156.write(":page:meas:swe:var1:comp 0.05")
```

```
    hp4156.write(":page:meas:swe:var2:mode sing")
    hp4156.write(":page:meas:swe:var2:spac lin ")
    hp4156.write(":page:meas:swe:var2:star 0.0 ")
    hp4156.write(":page:meas:swe:var2:step 0.1 ")
    hp4156.write(":page:meas:swe:var2:poin 11")    #ALSSS
    hp4156.write(":page:meas:swe:var2:comp 0.0001")

    #######SETTING UP DISPLAY PAGE#######

    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG:MODE fixed")
    hp4156.write(":PAGE:MEAS:MSET:SMU2:RANG:MODE limited")
    hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG:MODE limited")
    hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG:MODE limited")
    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG 1e-3")
    hp4156.write(":PAGE:MEAS:MSET:SMU2:RANG 1e-8")
    hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG 1e-8")
    hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG 1e-8")

    ###Graphics
    hp4156.write(":page:disp:set:grap:x:del")
    hp4156.write(":page:disp:set:grap:y1:del")
    hp4156.write(":page:disp:set:grap:y2:del")
    hp4156.write(":page:disp:set:dvar:del:all")
    #
    hp4156.write(":page:disp:set:grap:x:name 'VD'")
    hp4156.write(":page:disp:set:grap:x:scal lin")
    #
    hp4156.write(":page:disp:set:grap:y1:name 'ID'")
    hp4156.write(":page:disp:set:grap:y1:scal lin")
    hp4156.write(":page:disp:set:grap:y1:min -1mA")
    hp4156.write(":page:disp:set:grap:y1:max 3mA")

    ##List
    hp4156.write(":page:disp:set:list:del:all")
    #hp4156.write(":page:disp:set:list:sel
'VG','VD','ID','IG','ISUB','IS','VSUB'")
    hp4156.write(":page:disp:set:list:sel 'VG','VD','ID'")

    hp4156.write("*opc?")
    hp4156.read()

    hp4156.write(":page:scon:sing")
    wait(hp4156)
    outputs=[]

    for par in ['VG','VD','ID','VS']:
      hp4156.write('trac? %s' % par)
      val=hp4156.read()[2].strip().split(',')
      tmp=[]
      tmp.append(par)
      for i in val:
```

```
            tmp.append(i)

        outputs.append(tmp)

    # print outputs
    tabdata=""
    for j in range(len(outputs[0])):
        for i in range(len(outputs)):
            tabdata+="%s " % outputs[i][j]
        tabdata+="\n"
    return tabdata



def
idvg(gate=1,drain=2,source=3,substrate=4,dbias=0.050,vdmax=1.0,vdmin=-
0.3):
    integrationtime="short"
    #######SETTING UP CHANNEL PAGE#######
    #delete all initial settings
    hp4156.write(":page:chan:all:dis")
    hp4156.write(":page:chan:ufun:del:all")
    hp4156.write(":page:chan:uvar:del:all")

    #set to sampling mode
    hp4156.write(":page:chan:mode swe")

    #set up SMU's
    hp4156.write("page:chan:smu%s:vname 'VG'" % gate)
    hp4156.write("page:chan:smu%s:iname 'IG'" % gate)
    hp4156.write(":page:chan:smu%s:mode v" % gate)
    hp4156.write(":page:chan:smu%s:func var1" % gate)

    hp4156.write("page:chan:smu%s:vname 'VD'" % drain)
    hp4156.write("page:chan:smu%s:iname 'ID'" % drain)
    hp4156.write(":page:chan:smu%s:mode v" % drain)
    hp4156.write(":page:chan:smu%s:func var2" % drain)

    hp4156.write("page:chan:smu%s:vname 'VS'" % source)
    hp4156.write("page:chan:smu%s:iname 'IS'" % source)
    hp4156.write(":page:chan:smu%s:mode common" % source)
    hp4156.write(":page:chan:smu%s:func cons" % source)

    hp4156.write("page:chan:smu%s:vname 'VSUB'" % substrate)
    hp4156.write("page:chan:smu%s:iname 'ISUB'" % substrate)
    hp4156.write(":page:chan:smu%s:mode common" % substrate)
    hp4156.write(":page:chan:smu%s:func cons" % substrate)

    #######SETTING UP MEASUREMENT PAGE#######

    #set sampling
    hp4156.write("page:meas:mset:itim med")
```

```
    # VDS
    #hp4156.write(":page:meas:cons:SMU%s %s" % (drain,dbias))

    hp4156.write(":page:meas:swe:var1:mode sing")
    hp4156.write(":page:meas:swe:var1:spac lin ")
    hp4156.write(":page:meas:swe:var1:star %s" % vdmin)
    hp4156.write(":page:meas:swe:var1:step 0.02 ")   # ALSSS 0.02
    hp4156.write(":page:meas:swe:var1:stop %s" % vdmax)
    hp4156.write(":page:meas:swe:var1:comp .1")

    hp4156.write(":page:meas:swe:var2:mode sing")
    hp4156.write(":page:meas:swe:var2:spac lin ")
    hp4156.write(":page:meas:swe:var2:star 0.0 ")
    hp4156.write(":page:meas:swe:var2:step 0.1 ")
    hp4156.write(":page:meas:swe:var2:poin 11")    # ALSSS 11
    hp4156.write(":page:meas:swe:var2:comp 0.1")



    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG:MODE fixed")
    hp4156.write(":PAGE:MEAS:MSET:SMU2:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG 1e-3")
    #hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG 1e-8")
    #hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG 1e-8")

    #######SETTING UP DISPLAY PAGE#######

    ##Graphics
    hp4156.write(":page:disp:set:grap:x:del")
    hp4156.write(":page:disp:set:grap:y1:del")
    hp4156.write(":page:disp:set:grap:y2:del")
    hp4156.write(":page:disp:set:dvar:del:all")
    #
    hp4156.write(":page:disp:set:grap:x:name 'VG")
    hp4156.write(":page:disp:set:grap:x:scal lin")
    #
    hp4156.write(":page:disp:set:grap:y1:name 'ID'")
    hp4156.write(":page:disp:set:grap:y1:scal log")
    hp4156.write(":page:disp:set:grap:y1:min 1pA")
    hp4156.write(":page:disp:set:grap:y1:max 10mA")

    ##List

    hp4156.write(":page:disp:set:list:del:all")
    #hp4156.write(":page:disp:set:list:sel
'VG','VD','ID','IG','ISUB','IS','VSUB'")
    hp4156.write(":page:disp:set:list:sel 'VG','VD','ID'")

    hp4156.write("*opc?")
```

```
    hp4156.read()


    hp4156.write(":page:scon:sing")
    wait(hp4156)
    outputs=[]

    for par in ['VG','VD','ID','VS']:
      hp4156.write('trac? %s' % par)
      val=hp4156.read()[2].strip().split(',')
      tmp=[]
      tmp.append(par)
      for i in val:
          tmp.append(i)

      outputs.append(tmp)

    # print outputs
    tabdata=""
    for j in range(len(outputs[0])):
      for i in range(len(outputs)):
          tabdata+="%s " % outputs[i][j]
      tabdata+="\n"
    return tabdata


def
idvg_single(gate=1,drain=2,source=3,substrate=4,dbias=0.050,vdmax=1.0,
vdmin=-0.3):
    integrationtime="short"
    #######SETTING UP CHANNEL PAGE#######
    #delete all initial settings
    hp4156.write(":page:chan:all:dis")
    hp4156.write(":page:chan:ufun:del:all")
    hp4156.write(":page:chan:uvar:del:all")

    #set to sampling mode
    hp4156.write(":page:chan:mode swe")

    #set up SMU's
    hp4156.write("page:chan:smu%s:vname 'VG'" % gate)
    hp4156.write("page:chan:smu%s:iname 'IG'" % gate)
    hp4156.write(":page:chan:smu%s:mode v" % gate)
    hp4156.write(":page:chan:smu%s:func var1" % gate)

    hp4156.write("page:chan:smu%s:vname 'VD'" % drain)
    hp4156.write("page:chan:smu%s:iname 'ID'" % drain)
    hp4156.write(":page:chan:smu%s:mode v" % drain)
    hp4156.write(":page:chan:smu%s:func cons" % drain)

    hp4156.write("page:chan:smu%s:vname 'VS'" % source)
    hp4156.write("page:chan:smu%s:iname 'IS'" % source)
```

```
    hp4156.write(":page:chan:smu%s:mode common" % source)
    hp4156.write(":page:chan:smu%s:func cons" % source)

    hp4156.write("page:chan:smu%s:vname 'VSUB'" % substrate)
    hp4156.write("page:chan:smu%s:iname 'ISUB'" % substrate)
    hp4156.write(":page:chan:smu%s:mode common" % substrate)
    hp4156.write(":page:chan:smu%s:func cons" % substrate)

    #######SETTING UP MEASUREMENT PAGE#######

    #set sampling
    hp4156.write("page:meas:mset:itim med")

    # VDS
    hp4156.write(":page:meas:cons:SMU%s %s" % (drain,dbias))

    hp4156.write(":page:meas:swe:var1:mode sing")
    hp4156.write(":page:meas:swe:var1:spac lin ")
    hp4156.write(":page:meas:swe:var1:star %s" % vdmin)
    hp4156.write(":page:meas:swe:var1:step 0.02 ")
    hp4156.write(":page:meas:swe:var1:stop %s" % vdmax)
    hp4156.write(":page:meas:swe:var1:comp .1")

    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG:MODE fixed")
    hp4156.write(":PAGE:MEAS:MSET:SMU2:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG:MODE auto")
    hp4156.write(":PAGE:MEAS:MSET:SMU1:RANG 1e-3")
    #hp4156.write(":PAGE:MEAS:MSET:SMU3:RANG 1e-8")
    #hp4156.write(":PAGE:MEAS:MSET:SMU4:RANG 1e-8")

    #######SETTING UP DISPLAY PAGE#######

    ##Graphics
    hp4156.write(":page:disp:set:grap:x:del")
    hp4156.write(":page:disp:set:grap:y1:del")
    hp4156.write(":page:disp:set:grap:y2:del")
    hp4156.write(":page:disp:set:dvar:del:all")
    #
    hp4156.write(":page:disp:set:grap:x:name 'VG")
    hp4156.write(":page:disp:set:grap:x:scal lin")
    #
    hp4156.write(":page:disp:set:grap:y1:name 'ID'")
    hp4156.write(":page:disp:set:grap:y1:scal log")
    hp4156.write(":page:disp:set:grap:y1:min 1pA")
    hp4156.write(":page:disp:set:grap:y1:max 10mA")

    ##List

    hp4156.write(":page:disp:set:list:del:all")
    #hp4156.write(":page:disp:set:list:sel
'VG','VD','ID','IG','ISUB','IS','VSUB'")
```

```
    hp4156.write(":page:disp:set:list:sel 'VG','VD','ID'")

    hp4156.write("*opc?")
    hp4156.read()


    hp4156.write(":page:scon:sing")
    wait(hp4156)
    outputs=[]

    for par in ['VG','VD','ID','VS']:
      hp4156.write('trac? %s' % par)
      val=hp4156.read()[2].strip().split(',')
      tmp=[]
      tmp.append(par)
      for i in val:
          tmp.append(i)

      outputs.append(tmp)

    # print outputs
    tabdata=""
    for j in range(len(outputs[0])):
      for i in range(len(outputs)):
          tabdata+="%s " % outputs[i][j]
      tabdata+="\n"
    return tabdata



hp4156=e5810(host=ADDR, device="gpib0,17", raise_on_err=0,
timeout=5000,device_name="HP4156A")
PNA=e5810(host=ADDR, device="gpib0,16", raise_on_err=0,
timeout=5000,device_name="PNA")



def read_sdata():
    x=[];s11=[];s21=[];s12=[];s22=[]

    for ch in ("CH1_S11_1","CH1_S12_2","CH1_S21_3","CH1_S22_4"):
      if ch=="CH1_S11_1":
          sparam=s11
      if ch=="CH1_S12_2":
          sparam=s12
      if ch=="CH1_S21_3":
          sparam=s21
      if ch=="CH1_S22_4":
          sparam=s22
```

```
      PNA.write(':CALC:PAR:SEL "%s"' % ch)
      PNA.write(":CALC:DATA? SDATA")
      val=PNA.read()[2].strip().split(',')


      for i in range(0,len(val),2):
          sparam.append((float(val[i]),float(val[i+1])))

    PNA.write(":CALC:X?")
    xval=PNA.read()[2].strip().split(',')
    for xp in xval:
      x.append(float(xp))

    return((x,s11,s12,s21,s22))

def set_RF_power_dbm(power):
    PNA.write(':SOURCE1:POWER2 %s' % power)


topdir='0krad_Die9_DeviceA_stress_1'

#set_RF_power_dbm(-10)
#sys.exit()
set_RF_power_dbm(-25)

a=open("%s/IDVG_pre.dat" % topdir,'w')
a.write("%s" % idvg())
a.close()

a=open("%s/IDVD_pre.dat" % topdir,'w')
a.write("%s" % idvd())
a.close()

minute=60.
hour=minute*60.

vg_stress=0.6
vd_stress=1.0
#for iteration,stress_min in list(enumerate([0,10,10])):
#for stress_min,iteration in list(enumerate([0,1*minute, 1*minute,
1*minute, 1*minute, 1*minute])):
for iteration,stress_min in list(enumerate([5*minute, 20*minute,
1*hour, 3*hour, 5*hour, 8*hour, 8*hour])):
    if stress_min>0:
      print "begin RF and DC stress",iteration,stress_min
      sys.stdout.flush()
      stress_begin(vg=vg_stress,vd=vd_stress)

      set_RF_power_dbm(0)
      time.sleep(stress_min)
      stress_end()
      set_RF_power_dbm(-25)
```

```
      print "ending RF and DC stress"
      sys.stdout.flush()

    for vd in [0.3,0.6,1.0]:
    #for vd in [0.6,]:
      for vg in [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0]:
      #for vg in [0.2,0.4,0.6,0.8,1.0]:
          print "Gate Voltage: %s    Drain Voltage: %s" % (vg,vd)
          stress_begin(vg=vg,vd=vd)
          time.sleep(0.2)
          x,s11,s12,s21,s22=read_sdata()
          stress_end()
          time.sleep(0.1)

          filename="%s/Vg%04iVd%04i_%s_%s.s2p" %
(topdir,vg*1000,vd*1000,iteration,stress_min)
          print filename
          outfile=open(filename,'w')
          for f,p1,p2,p3,p4 in zip(x,s11,s12,s21,s22):
            outfile.write('%s %s %s %s %s %s %s %s %s\n' %
(f,p1[0],p1[1],p2[0],p2[1],p3[0],p3[1],p4[0],p4[1]))
          outfile.close()

    set_RF_power_dbm(-25)
    a=open("%s/IDVG_post_%s_%s.dat" %
(topdir,iteration,stress_min),'w')
    a.write("%s" % idvg())
    a.close()
```