EXPERIMENTAL STUDY AND MODELING OF THE G_M-I DEPENDENCE OF LONG-CHANNEL MOSFETS

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

by

Michael Fong Cheng

March 2019

COMMITTEE MEMBERSHIP

TITLE:	Experimental Study and Modeling of the g_m-I Dependence of Long-Channel MOSFETs
AUTHOR:	Michael Fong Cheng
DATE SUBMITTED:	March 2019
COMMITTEE CHAIR:	Vladimir Prodanov, Ph.D. Associate Professor of Electrical Engineering
COMMITTEE MEMBER:	Tina Smilkstein, Ph.D. Associate Professor of Electrical Engineering
COMMITTEE MEMBER:	David Braun, Ph.D. Professor of Electrical Engineering

ABSTRACT

Experimental Study and Modeling of the g_m-I Dependence of Long-Channel MOSFETs

Michael Fong Cheng

This thesis describes an experimental study and modeling of the current-transconductance dependence of the ALD1106, ALD1107, and CD4007 arrays. The study tests the hypothesis that the I-g_m dependence of these 7.8 µm to 10 µm MOSFETs conforms to the Advanced Compact Model (ACM). Results from performed measurements, however, do not support this expectation. Despite the relatively large length, both ALD1106 and ALD1107 show sufficiently pronounced 'short-channel' effects to render the ACM inadequate. As a byproduct of this effort, we confirmed the modified ACM equation. With an m factor of approximately 0.6, it captures the I-g_m dependence with sub-28% maximum error and sub-10% average error. The paper also introduces several formulas and procedures for I-g_m model extraction and tuning. These are not specific to the ALD transistor family and can apply to MOSFETs with different physical size and electrical performance.

Keywords: analog, MOSFET, transconductance, model, ACM

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

In electronic design, transistors find use in multiple circuits. Digital circuits typically use transistors as switches, while analog circuits use them as amplifiers. As an amplifier, the voltage across the gate-source terminals of a MOSFET device changes the current flowing through its source-drain terminals. Biasing the transistor at a particular DC operating point maximizes potential small-signal gain, a parameter proportional to the transconductance.

1.1 MOSFET Transconductance

The slope of the transfer characteristic of a transistor evaluated at the operating point provides the transconductance, g_m. The following shows the transconductance equation for a MOSFET.

$$g_m = \left.\frac{di_D}{dv_{GS}}\right|_{o.p.} \tag{1.1}$$

For a given g_m, the product of g_m and the output resistance yields the voltage gain of the circuit.

The transconductance of a MOSFET depends on the device's length, width, oxide thickness, and bias current I_D. The Advanced Compact Model (ACM) models the I-g_m relations of a MOSFET and applies to long-channel devices. It provides a continuous function between sub-threshold and strong inversion operating regions. According to the ACM [1], the I-g_m expression for a MOSFET in saturation is:

$$g_m = \frac{2}{1+\sqrt{1+I_D/I_S}} * \frac{I_D}{nU_T} \tag{1.2}$$

The normalization current, I_S, defined in (1.3), where U_T equals the thermal voltage

kT/q. The rest of the quantities have their usual meaning.

$$I_S = \frac{1}{2} n \mu C_{ox} \frac{W}{L} U_T^2 \tag{1.3}$$

When I_D is much smaller than I_S, the device operates in sub-threshold or weak inversion, where the gate transconductance takes the form:

$$g_m = \frac{I_D}{nU_T} \tag{1.4}$$

When I_D is much larger than I_S, the device operates in strong inversion, where the transconductance is proportional to the square-root of the drain current. However, short-channel MOSFETs behave differently. When the channel length of the transistor reduces, the channel carrier velocity reaches a limit due to high lateral electric field values [2]. Expression (1.5) shows how electric fields affect carrier mobility, where v_d is drift velocity [3].

$$\mu = \frac{v_d}{|E|} \tag{1.5}$$

Since $E = \frac{V}{L}$ in a constant field, the shorter-channel length for a given V_{DS} yields a larger field than a longer-channel length. The following equation details the effective mobility, where (1.6) takes the place of μ in (1.3).

$$\mu_{eff} = \frac{\mu}{1 + \frac{\mu}{v_{max}} \frac{V_{DS}}{L}} \tag{1.6}$$

This effect only impacts higher drain-source voltages in saturation and manifests itself in a weaker than square-root g_m-I dependence [3].

According to [4], replacing the square root in (1.2) with a larger-value exponent m models the impact of strong electric fields. Equation (1.7) shows the modification, which previously lacked experimental verification.

$$g_m \cong \frac{2}{1 + (1 + I_D/I_{norm})^m} \times \frac{I_D}{V_{norm}}, \quad m \geq 0.5 \tag{1.7}$$

Here V_{norm} equals nU_T, and I_{norm} takes the place of I_S. The modified ACM's simplicity and capability of accounting for various levels of 'high-field' effects make it attractive.

1.2 Thesis Organization

This work studies the behavior of two MOSFETs intended for analog applications, ALD1106 and ALD1107, with the goal of demonstrating their I-g_m characteristics conform to (1.2). Expecting a good agreement with the ACM, we learned that capturing the I-g_m relation needed the modified ACM instead. We applied the modified ACM to a chip intended for digital circuits, the CD4007, and found it performed better than the original ACM.

Chapter 2 introduces the approach to collecting voltage data and the circuit setup that facilitates measurement. Chapter 3 discusses the algorithmic mechanics of applying the modified ACM to experimental data. Chapter 4 covers migration from breadboard to PCB test fixture. Chapter 5 reports and analyzes PCB test fixture data for the ALD1106 and ALD1107, and chapter 6 covers CD4007 results. Chapter 7 proposes expansions and improvements to the model with chapter 8 concluding the report altogether.

Chapter 2

I-V DATA COLLECTION AND PREPROCESSING

We wish to find whether the I-g_m dependence of ALD1106 and ALD1107 obeys (1.2). This stems from both transistors having a channel length of 7.8 µm, implying a 'long-channel' device [2],[5]. The first step of the process obtains reliable I-V data that covers the operating range of the device. This includes weak, moderate, and strong inversion.

2.1 Data Collection Methodology

Our experimental data consists of more than one hundred fifty I-V points per transistor studied. As shown in Table 2.1, the current spans approximately seven decades, and for the n-channel devices (ALD1106), it ranges from 1 nA to 19 mA. The range for the p-channel devices (ALD1107) is smaller: 1 nA to 6.5 mA. Different maximum currents ensure the gate-source voltage of the FET under test does not exceed the 10.6 V limit defined by the manufacturer [6],[7].

Pseudo-logarithmically spacing I-V points in the current domain yields ten uniformly distributed points within each decade. An exception in the 1-to-100 µA range, where we took 100 points as opposed to 20, better captures transistor transition from weak to strong inversion.

Table 2.1: NUMBER OF I-V POINTS PER TRANSISTOR AND THEIR DISTRIBUTION

		1-10 nA	**10-100** nA	**0.1-1** µA	**1-100** µA	**0.1-1** mA	**Beyond 1** mA
Number of Points	ALD1106	10	10	10	100	10	19 (1 mA step)
	ALD1107	10	10	10	100	10	13 (0.5 mA step)

2.2 Circuit Setup

The ALD1106 and ALD1107 arrays come in a plastic dual in-line package (PDIP). This allows for easy breadboard testing. With these chips mounted on the breadboard, solid-core copper wiring made the appropriate connections. We minimized board wiring lengths to reduce resistive loss and pickup of electromagnetic interference that would corrupt measurements.

All data uses a diode-connected transistor configuration where the Keithley-2400 serves as a current source to bias the transistor under test. Fig. 2.1 and 2.2 show the test setup. The Keithley supplies the drain current and simultaneously measures the V_{GS} of the device, eliminating the need for additional test equipment.

The attractiveness of extracting I-V data from a diode-connected device stems from a simple setup that keeps the MOSFET in saturation. The disadvantage of the 'diode-connection' is that the change in I_D stems from changes in both V_{GS} and V_{DS}. This means that the derivative of the drain current with respect to V_{GS} yields the sum of g_m and g_{ds}. This paper assumes that the transconductance g_m represents the gate transconductance referenced to the source electrode. The following equation

details drain transconductance referenced to the source electrode, g_{ds}.

$$g_{ds} = \left.\frac{di_D}{dv_{DS}}\right|_{o.p.} \tag{2.1}$$

Attributing the change to g_m alone, results in an error. The error, however, stays relatively small because the g_m is typically much larger than g_{ds} [1]. Additionally, operating the transistor in saturation prevents mobility reduction from large gate voltages at small drain voltages, which could skew high lateral field effect characterization [3].

At least two phenomena could compromise the validity of the data gathered: thermal heating and noise. An automated data collection scheme addressed thermal heating, a concern at high currents. A LabVIEW program controlled the pseudo-logarithmic current sweep (see Table 2.1) and collected gate-source voltage measurements, reducing the time the MOSFET spends in a high-current regime. Minimizing heating proves essential because thermal effects manifest in a manner similar to high-field effects: I_D decreases as temperature increases [7], [6]. This translates to a lower g_m, which implies a reduction in carrier mobility.

Noise and electromagnetic interference create concerns at lower drain currents, where the device becomes very resistive having $1/g_m$ in the hundreds of kiloohms. Averaging samples potentially solves this issue because white noise represents a zero-mean stochastic process. This means that the averaging of more samples should converge to zero. The median should also be zero. Subsequently, each data point takes ten samples for preprocessing for currents below 1 µA. The preprocessing, implemented in MATLAB, calculates either the median or mean of these ten samples. The difference between median and mean preprocessing is small. When preprocessing data using the median of the ten current points, the overall error between experimental and model data was smaller than when processed with the mean. While white noise is a zero-mean stochastic process, the attempt at reducing its impact through an

ensemble average is not very effective. This could stem from taking only ten samples. With more samples, the mean preprocessing might work better than the median.

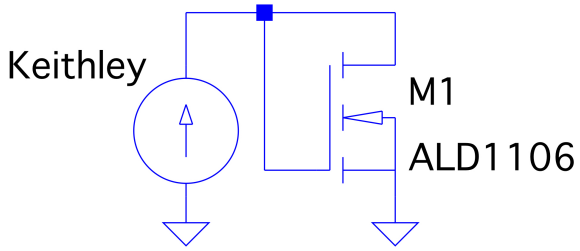

Figure 2.1: NMOS Circuit Topology used for Measuring of MOSFET I-V Dependence

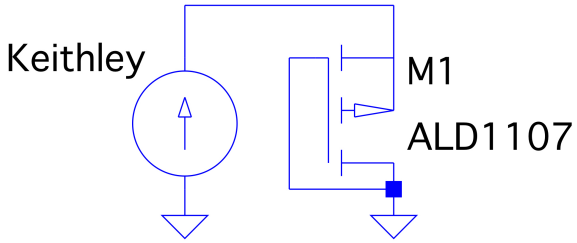

Figure 2.2: PMOS Circuit Topology used for Measuring of MOSFET I-V Dependence

Chapter 3

TUNING OF THE MODIFIED ACM EXPRESSION

This section deals with selecting V_{norm}, I_{norm}, and m in (1.7) to optimize the fit with experimental data.

The model's I-g_m domain complicates the tuning task through measurements in the I-V domain. Differentiating I-V measurements produce experimental I-g_m values. This compares experimental and model transconductance values. Fig. 3.1 clarifies this approach. Equation (3.1) in Fig. 3.1 appears in the next section. We show the full derivation in Appendix A. Alternatively, as depicted in Fig. 3.2, one could convert the model-generated I-g_m data to I-V domain and perform a comparison of voltage values. Equation (3.4) in Fig. 3.2 also appears in the next section. We also show the full derivation in Appendix A. We use both approaches in this paper, because we want to show that (1.7)'s robustness, where its parameters lack sensitivity to the tuning strategy used.

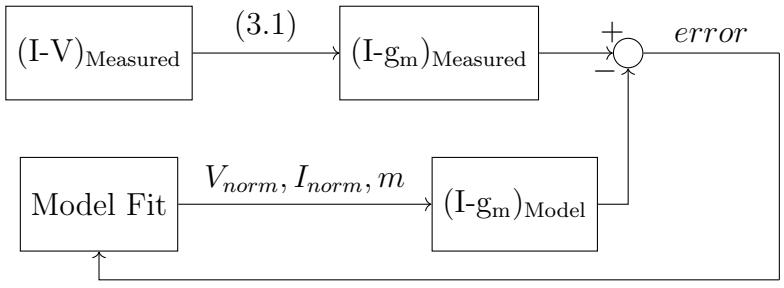

Figure 3.1: Fitting Algorithm that Reduces Error in I-g_m Domain

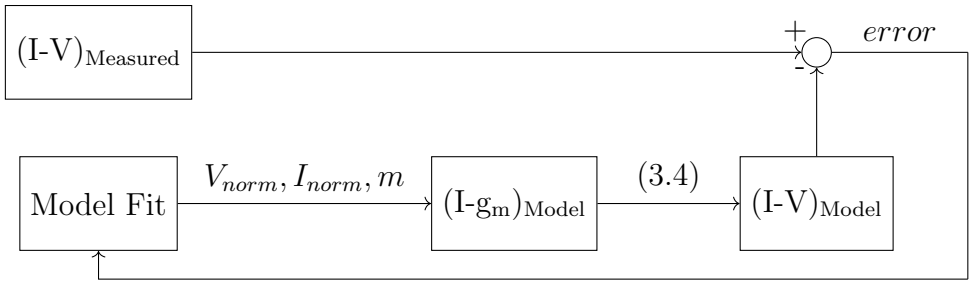

Figure 3.2: Fitting Algorithm that Reduces Error in I-V Domain

3.1 Model Tuning by Comparing Transconductance Values

Extracting and verifying an I-g_m model from measured I-V data requires differentiation typically approximated with forward or backward differences.

Because this study relies on the accuracy of the first derivative, we instead used expression (3.1), a formula based on quadratic spline interpolation of the data. The quadratic spline interpolation takes three points and assumes a quadratic function passes through the three points [8]. We use the derivative of this quadratic at the center point as the first derivative of the nonlinear function. Following this idea, we derived (3.1).

$$g_m \cong \frac{I_{D(k)} - I_{D(k-1)}}{V_{GS(k)} - V_{GS(k-1)}} + \frac{I_{D(k+1)} - I_{D(k)}}{V_{GS(k+1)} - V_{GS(k)}} - \frac{I_{D(k+1)} - I_{D(k-1)}}{V_{GS(k+1)} - V_{GS(k-1)}} \tag{3.1}$$

Expression (3.1) generalizes the simpler two-point differences. It uses all three differences one could form between three points and guarantees zero error for both linear and quadratic dependencies. Appendix A lists the complete derivation of (3.1).

When taking discrete derivatives, the spacing between points severely impacts the accuracy and potentially introduces severe errors. These errors significantly grow if the differentiated function changes rapidly. Of the mathematical functions the drain

current can follow, the weak inversion current behavior described in (3.2) follows an exponential form, which has the most rapid slope change.

$$I = I_s e^{\frac{V}{V_{norm}}} \tag{3.2}$$

To test the accuracy of our quadratic spline interpolation, a MATLAB script compared the discrete differentiation results with theoretical values. We started by limiting the voltage values from 0 to 10.5 V, since both the ALD1106 and ALD1107 have a 10.6 V limit [7], [6]. The following formula details theoretical derivative of (3.2).

$$\frac{dI}{dV} = \frac{I_s}{V_{norm}} e^{\frac{V}{V_{norm}}} \tag{3.3}$$

The number of points between 0 and 10.5 V varied from 300 to 1000. These points passed through (3.2) to generate the appropriate data for (3.1). To generate the fastest growing exponential possible, V_{norm} is 26 mV to shrink the denominator in the exponential argument, and I_s is 10 µA for a large scaling value. Fig. 3.3 shows that 1000 linearly spaced points between 0 and 10.5 V ensures a maximum error of 2.746%. The 1000 linearly spaced points translates to an inter-point granularity of 10.5 mV, and the experimental inter-point granularity averaged around 10.875 mV. This implies that the derivative error should not exceed 5%.

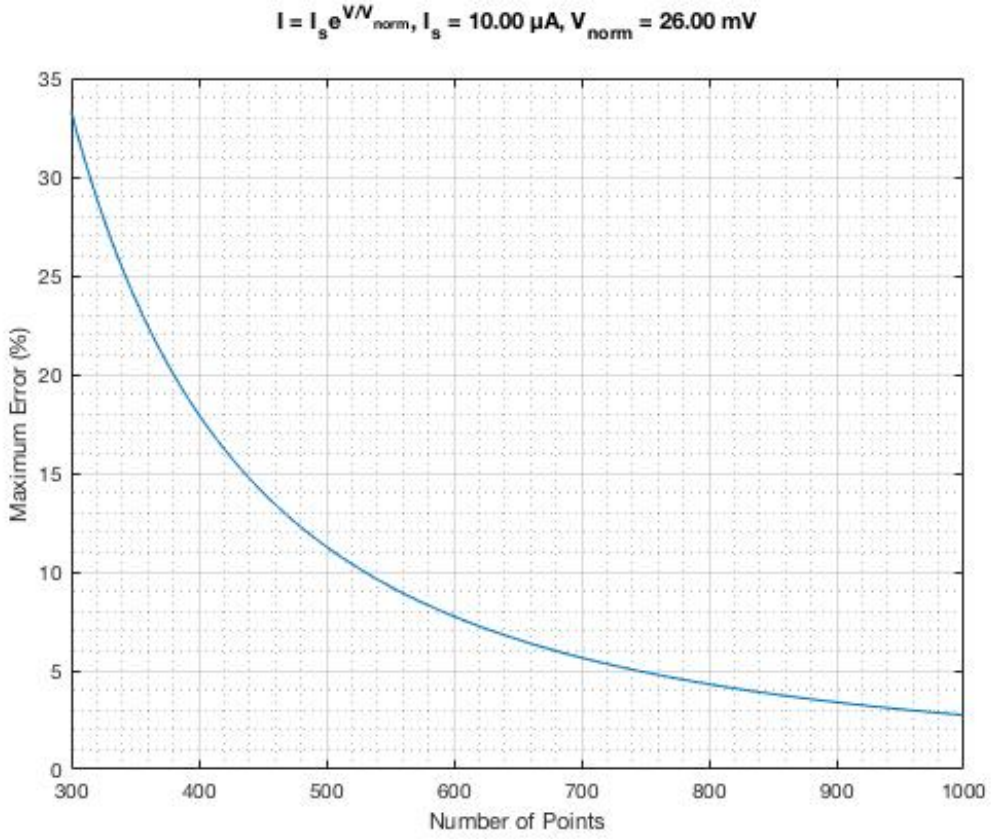

Figure 3.3: Derivative Error for Functions Similar to Equation (3.2)

Fig. 3.4 and 3.5 show that differentiating quadratic and linear polynomials create errors in the 10^{-12}% range. This approaches the inherent machine error from performing base 10 math on a base 2 processor. The quadratic function chosen has similar scaling constants to the exponential test function for analyzing potential errors. Fig. 3.5 has more error than the quadratic function, but Fig. 3.6 helps explain the discrepancy. Since computers have fixed binary precision, MATLAB's 64-bit double-precision floating point variables experience round-off errors for numbers with fractional precision. Fig. 3.6 shows errors from polynomials with integer scaling constants, as opposed to small fractional numbers. This lowers the error from decimal math approximation on a binary processor. The error significantly drops for the larger scaling constants. This implies introduction of floating-point precision errors when

differentiating functions with small constant coefficients. Fig 3.5 shows 468, 664, and 868 points produces a lower 1.292% maximum error. To verify these floating-point errors, the number of points varied from 250 to 1050 with a range between 0 and 1 V. Fig 3.6 exhibits the same phenomenon, where the error drops to zero for 257, 513, and 1025 points between 0 and 1 V. Considering the number of points include 0 V, all the errorless number of points are powers of two. This verifies that the lack of error stems from not approximating a base 10 calculation with a base 2 one.

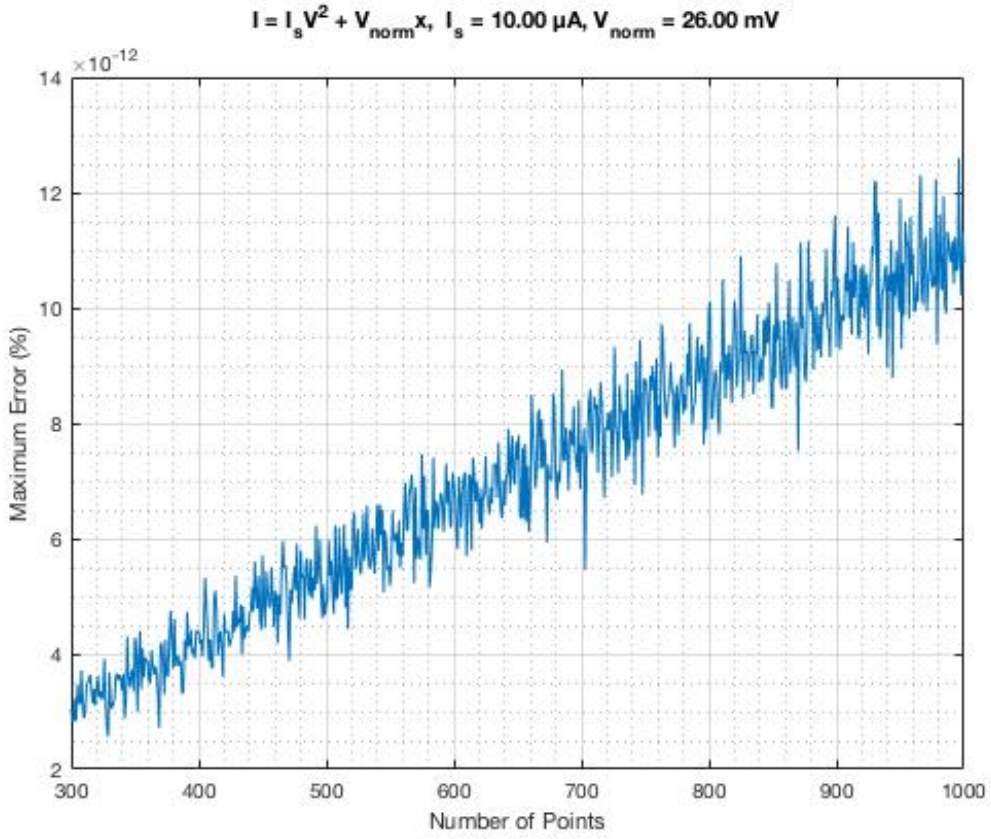

Figure 3.4: Derivative Error for Functions Similar to Current of Saturated MOSFET in Strong Inversion

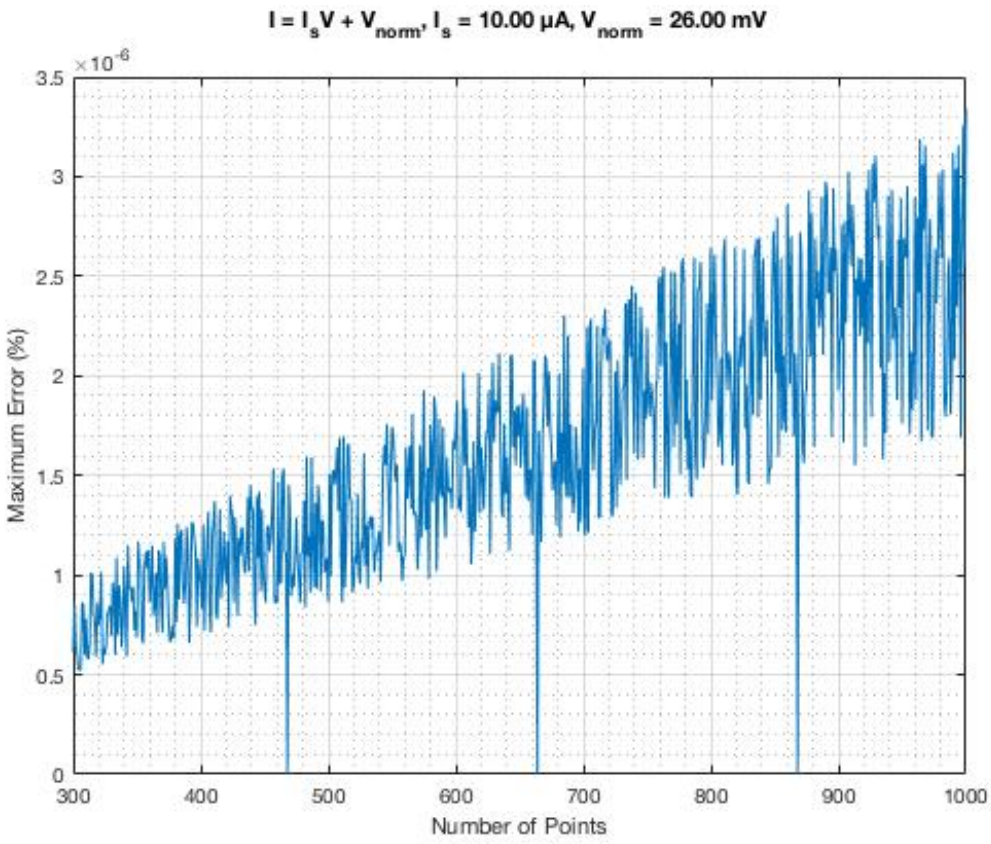

Figure 3.5: Derivative Error for Linear Functions

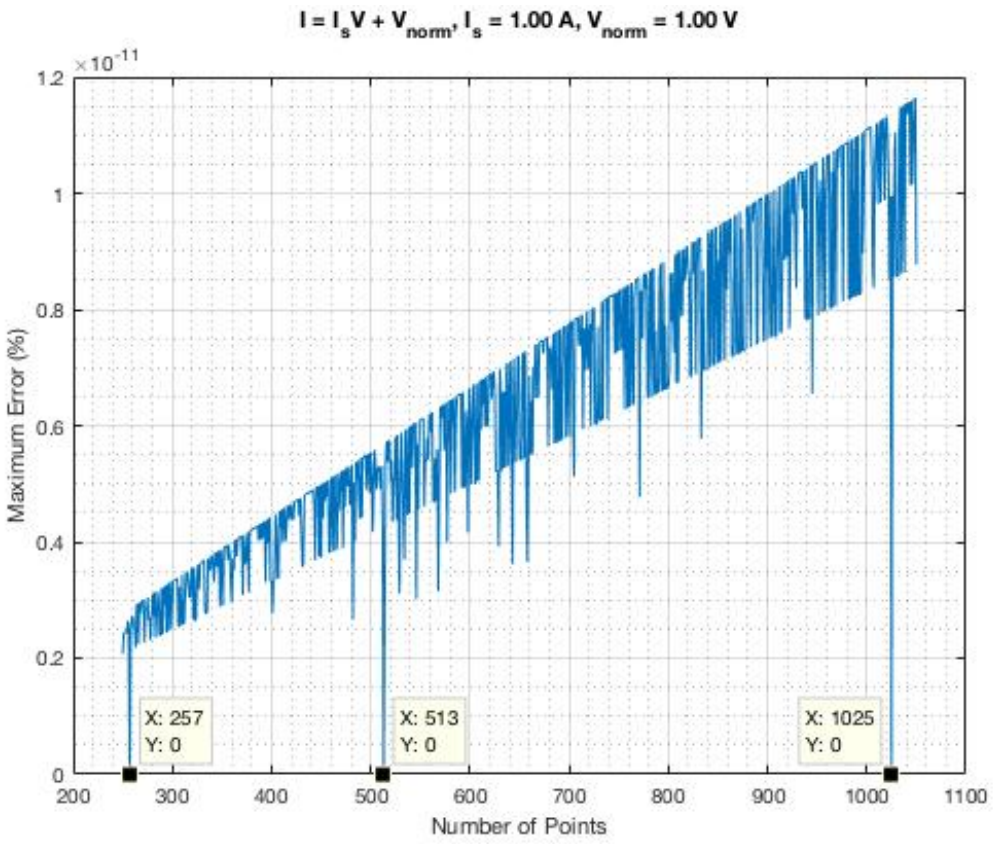

Figure 3.6: Derivative Error for Large Value Linear Functions

3.2 Model Tuning by Comparing V_{GS} Values

Equation (3.1) generates I-g_m points from the experimental I-V set. As such, it does not offer a direct comparison to measured data. Also, the derivative operator (3.1) may enhance errors present in the measured data. To circumvent this issue, we also consider the tuning strategy of Fig. 3.2 - an approach based upon expression (3.4). Expression (3.4) derives from (3.1) and allows us to produce an I-V set from an I-g_m one. It approximates an integral operation, because integrating the transconductance yields the gate-source voltage.

$$V_{GS(k+1)} \cong V_{GS(k)} + 2\frac{I_{D(k+1)} - I_{D(k)}}{g_{m(k+1)} + g_{m(k)}} \tag{3.4}$$

The tuning strategy in Fig. 3.2 provides direct comparison to measured data and offers more intuition on how accurately the model matches I-V measurements.

3.3 Selecting a Cost Function to Minimize

In this section, we discuss the algorithm to curve-fit the modified ACM to the experimental data. This requires minimization of error between experimental data points and model-generated values. We calculate the error by taking the absolute value of the difference between each experimental and model value, normalizing it by the experimental value. Depending on the tuning strategy used, the error calculation involves either g_m values or V_{GS} values, but the normalization ensures the error is always a dimensionless quantity.

Our measured and model-produced data sets consist of N points (N > 150). Hence, the error could represent an N-dimensional vector **e** and the tuning process as an effort to minimize the 'length' of the vector. According to [9], the length of an N-dimensional vector in R_N space is its norm (3.5).

$$||\mathbf{e}||_p = \left(\sum_{i=1}^{N} e_i^p \right)^{\frac{1}{p}}, \quad p \geq 1 \tag{3.5}$$

The most common norms are the grid norm (p=1), the Euclidean norm (p=2), and the infinity norm (p=∞). In this study, we choose to work with l_1 and l_∞, not only because they represent the two ends of the spectrum, but also because they have meaningful interpretations. As seen in (3.6) and (3.7), the average error relates to l_1, whereas the maximum error equals l_∞.

$$e_{ave} = \frac{e_1 + e_2 + ... + e_n}{n} \equiv \frac{||\mathbf{e}||_1}{N} \tag{3.6}$$

$$e_{max} = max\{e_1, e_2, ..., e_N\} \equiv ||\mathbf{e}||_\infty \tag{3.7}$$

Minimizing the Euclidean (l_2) norm, not pursued here, should perform between the 1-norm and the infinity-norm. Any intermediate p-norm performs between 1-norm and infinity-norm.

3.4 Defining the Search Space

To find the optimal model, we test various combinations of V_{norm}, I_{norm}, and m against experimental data. Reducing the number of cases tried requires establishing a meaningful range for each parameter. The consideration details follow.

Because I_D/V_{norm} represents the transconductance in deep sub-threshold, finding the g_m at low currents and taking the ratio I_D/g_m yields an approximate value for V_{norm}. In this study, one hundred possible normalization voltages generate from a ten percent window around the calculated approximate value of V_{norm}.

The normalization current, I_{norm}, delineates weak inversion from strong inversion. Examination of the experimental data for the ALD1106 and ALD1107 reveals that this transition occurs at current levels below 10 µA. Thus, we took 100 evenly spaced current values between 0 and 10 µA.

Ideally, the optimum parameter search returns an m value of 0.5, validating the ACM expression (1.2), but larger values for m are also likely. As discussed in [4], m values in the range of 0.7 to 1 are common for sub-0.5 µm MOSFETs. This justifies a search range of 0.5 to 1. We again take one hundred m values evenly spaced between 0.5 and 1.

With each parameter having 100 possible values, the tuning algorithm examines one million possible solutions. The lowest error between (3.6) and (3.7) passes as

the model fit output. For each transistor, we have four 'best' models corresponding to four possible combinations of two tuning schemes, (3.1) and (3.4), and two cost functions, (3.6) and (3.7). Chapters 5 and 6 discusses these differences.

Chapter 4 first discusses the transition from breadboard to PCB. Chapter 7 and 8 discuss future work and summarize results.

Chapter 4

PCB ASSEMBLY AND TESTING

4.1 Designing the PCB Test Fixture

This chapter describes two experimental strategies employed to measure transistor current versus voltage characteristics. We collect several data sets. The first set mounts transistor arrays on a breadboard. The long added trace length and poor electrical contact from the spring loaded pins make breadboards susceptible to noise effects. Furthermore, the connection between the Keithley 2400 current source and the breadboard consists of two banana-to-grabber connectors. These grabber connectors latches onto loose 22-gauge solid-core copper wires. This also potentially injects more noise into the tested circuit. For the first data set from devices mounted on the breadboard, the current proves accurate despite low current data measurements. We fabricate a printed circuit board to verify the accuracy of breadboard data. We did not solder short wires directly to the chips because of potential repeatability issues. Once populated, the circuit board's chip sockets allow for changing of ICs without re-soldering. Re-soldering subjects the transistor arrays to intense temperature changes and does not guarantee the consistency of the solder joints between tests. The circuit board includes jumpers to easily reconfigure transistors under test without re-soldering. Potential configurations include disconnecting the transistor's drain and gate or introducing substrate biasing. The breadboard suffers from particularly long and thin traces, which potentially adds parasitic impedance, so the PCB has 40 mil traces to ensure low impedance paths between pins. BNC jacks connects the current source to the test fixtures to ensure the use of coaxial cable between the Keithley 2400 and PCB. While the Keithley 2400 has only banana jacks, a banana-

to-BNC adapter mounted directly on the supply minimizes potential noise impacts. The coaxial cable improves noise suppression. Fig. 4.1 and 4.2 show the complete test fixture schematic and layout.

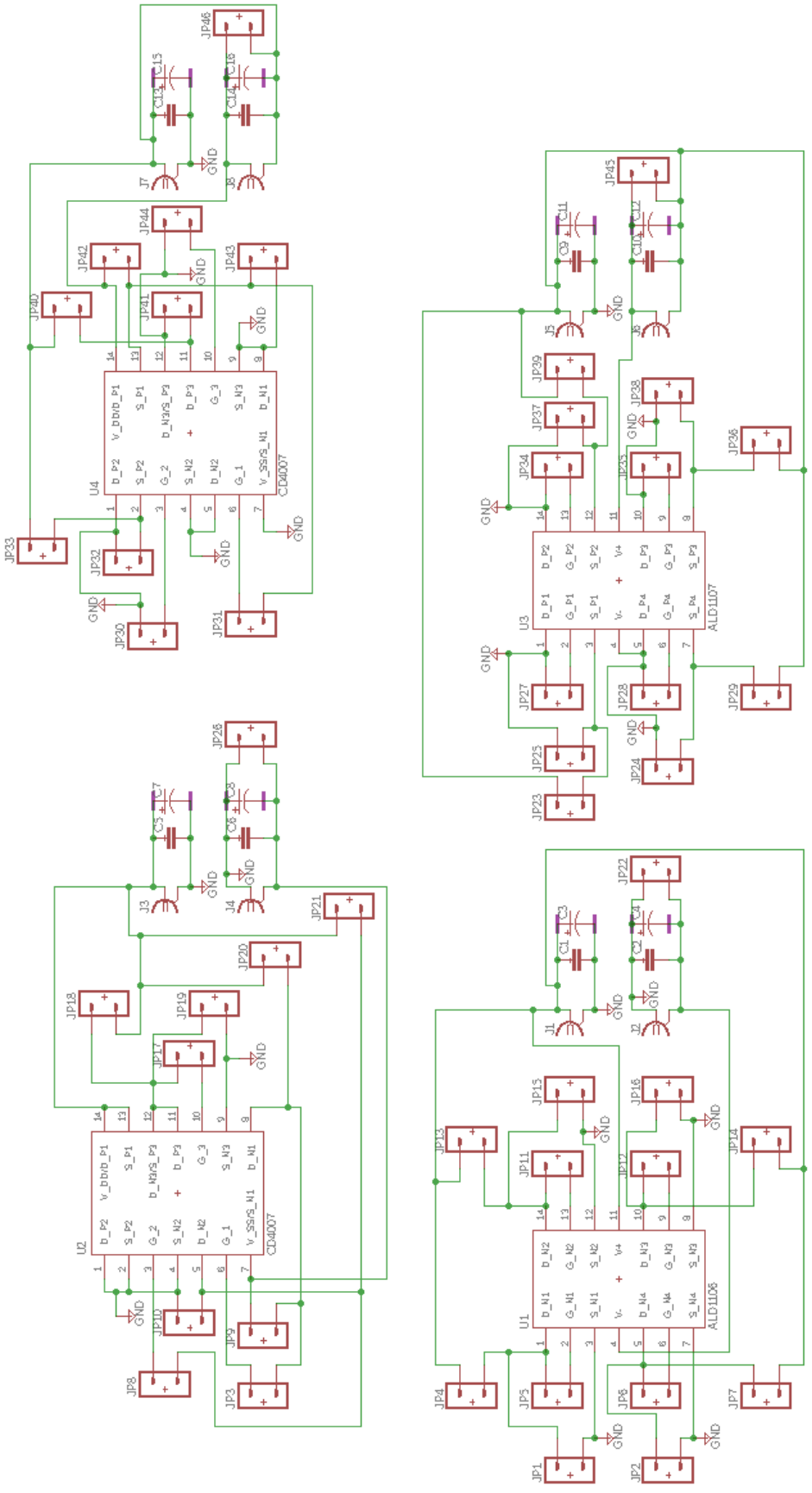

Figure 4.1: PCB Schematic

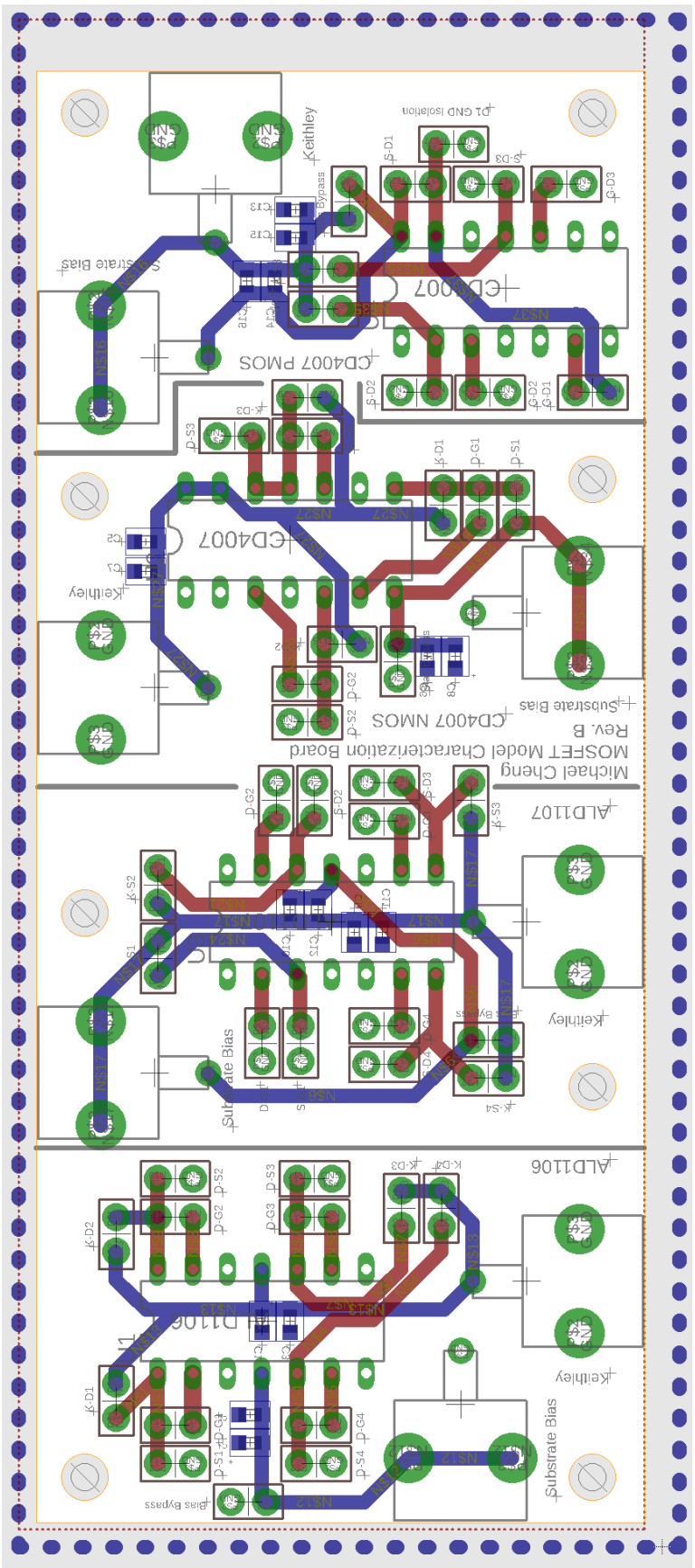

Figure 4.2: PCB Layout

OSH Park's two sided, 1 oz. copper option fabricates this board and appears in Fig. 4.3 and 4.4. Fig. 4.5 and 4.6 show the completed and cleaned boards with M3 screws and standoffs.

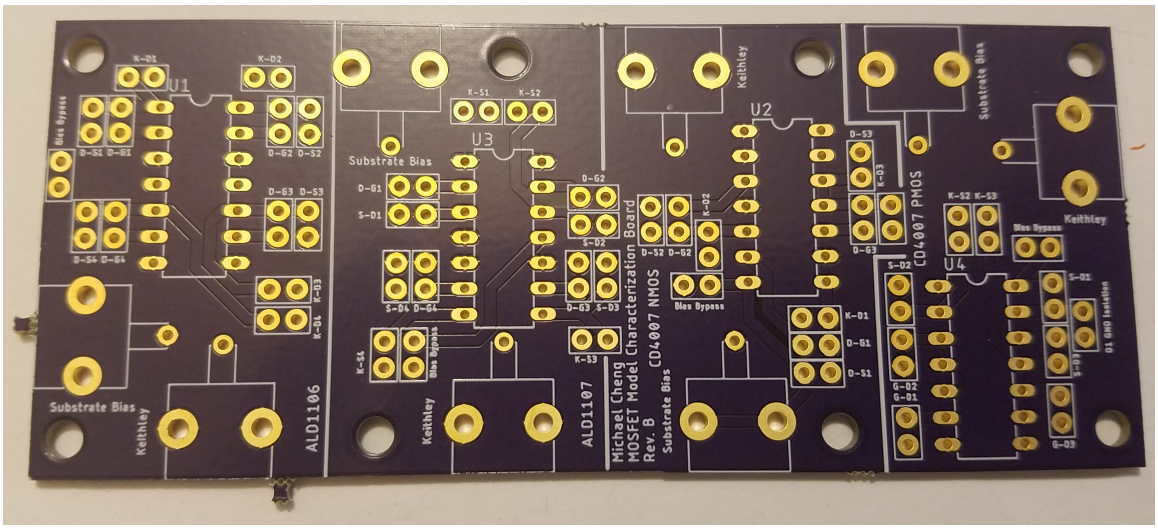

Figure 4.3: Top Side of Unpopulated PCB

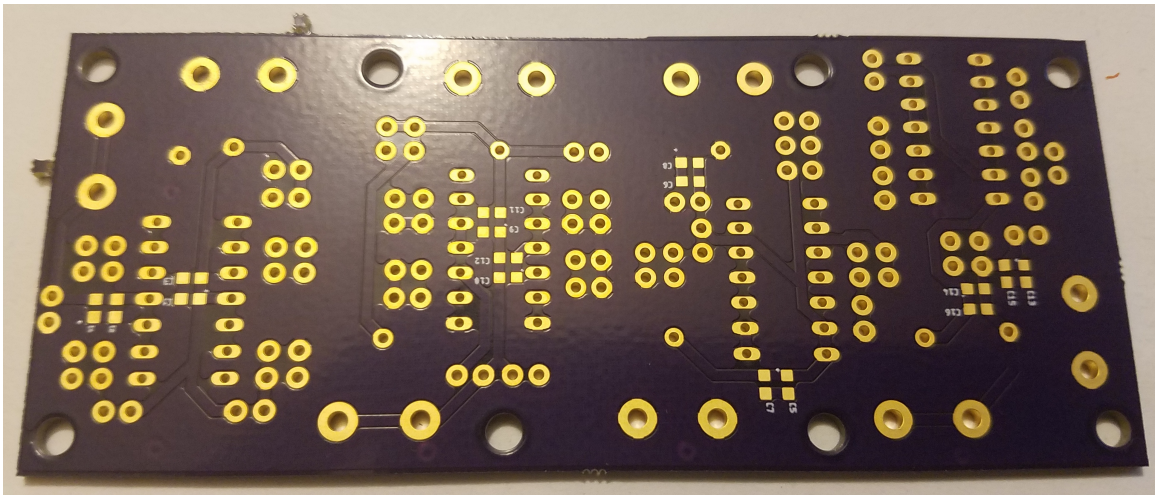

Figure 4.4: Bottom Side of Unpopulated PCB

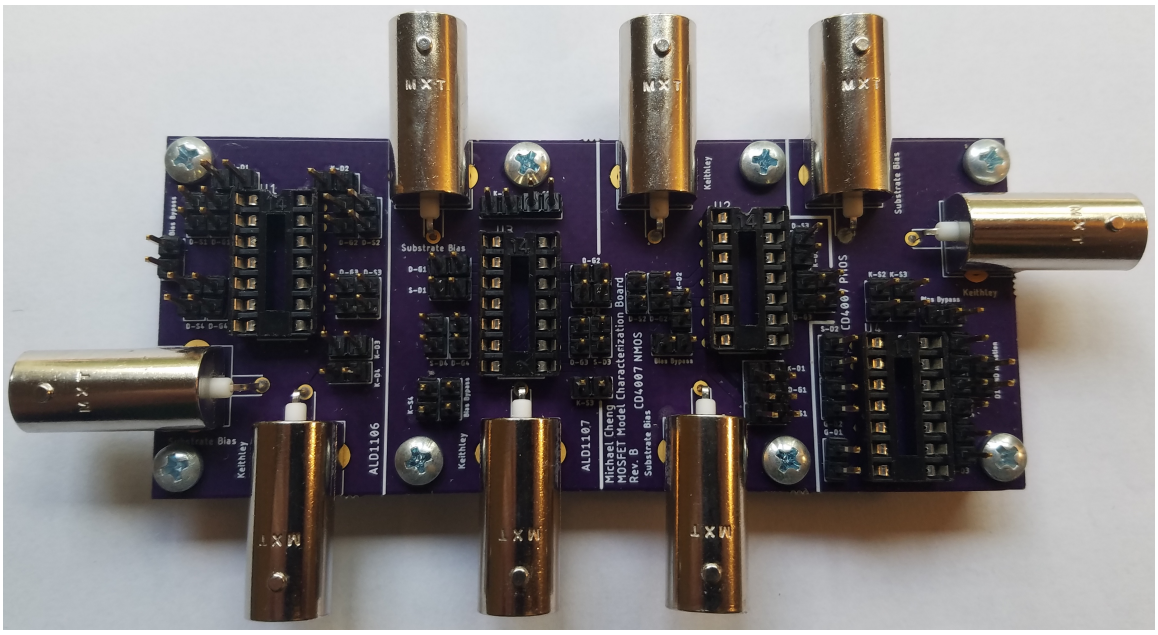

Figure 4.5: Top Side of Populated PCB with Arrays and Jumpers Removed

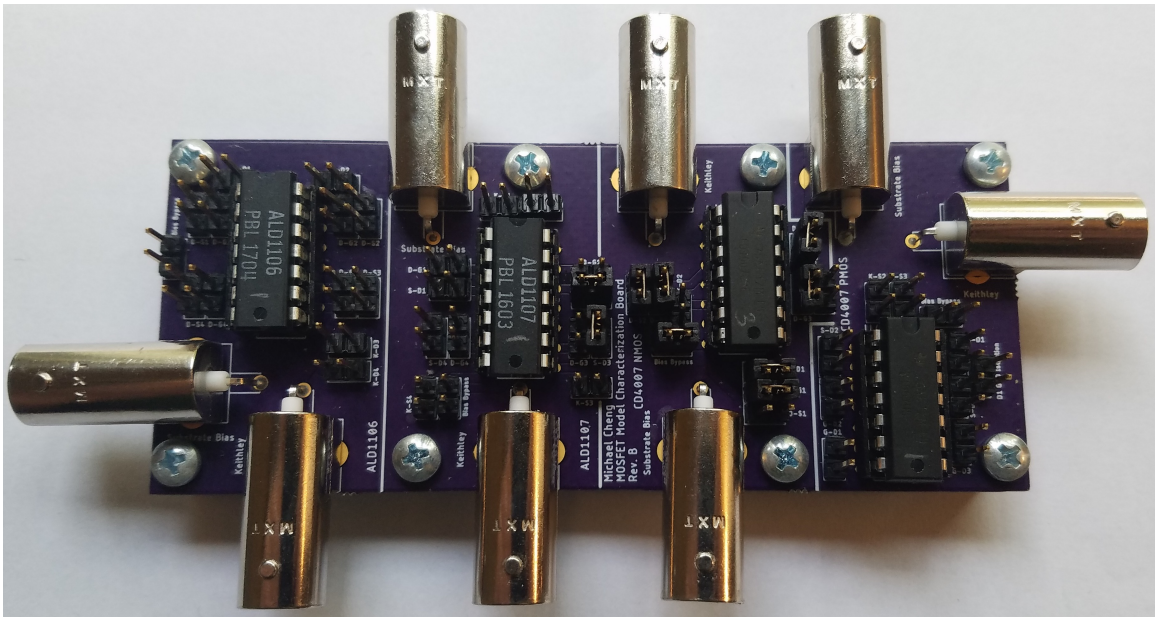

Figure 4.6: Top Side of Populated PCB with Arrays and Jumpers

4.2 Current Leakage due to Flux Residue

Initial results from the fabricated and populated boards differed greatly from breadboard results. Initially, this sparked confusion, as PCB results are typically more accurate than breadboard results because of reduced parasitic resistance, capacitance, and inductance. However, the breadboard results had low-error model parameters after the curve fitting algorithm. With a drain current of 100 pA, the breadboard's drain-source voltage measurement is 195.34 mV, but the PCB's voltage measurement is 25.89 mV. This severe drop in voltage implies current leakage somewhere in the circuit.

The first suspected cause of altered measurements were the added decoupling capacitors intended to mitigate voltage spikes from the supply. Fig. 4.7 and 4.8 show a modified 2.1 and 2.2 circuit.

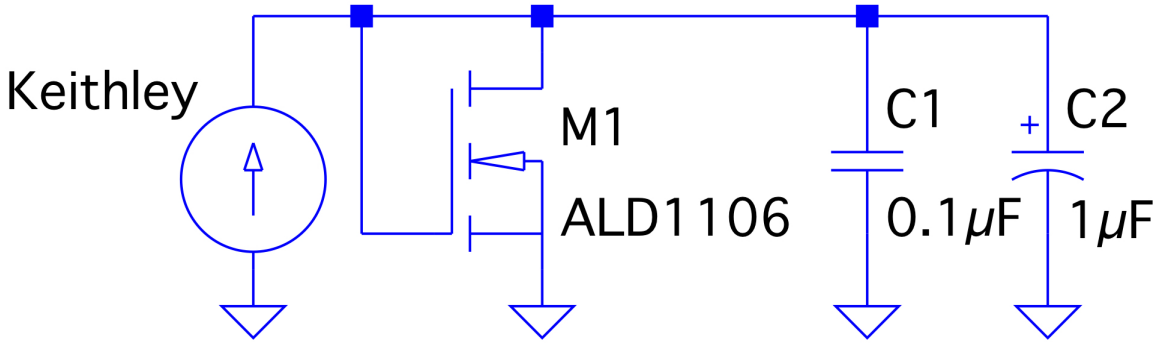

Figure 4.7: Modified NMOS Circuit Topology used for Measuring of MOSFET I-V Dependence

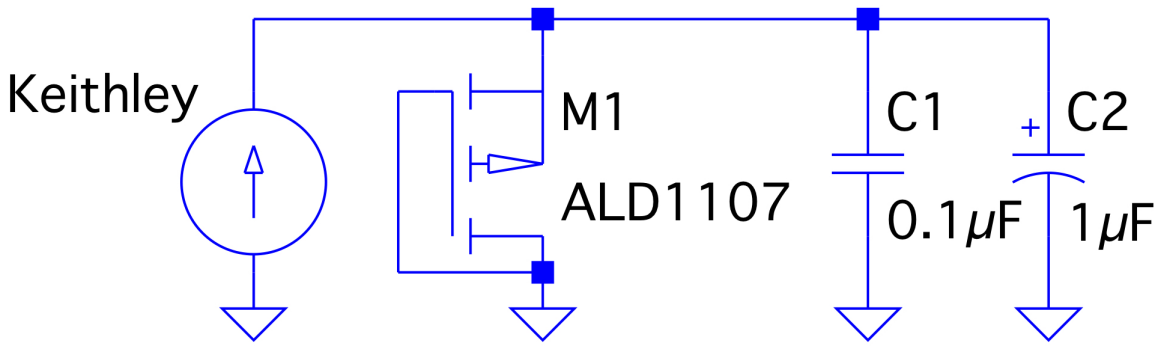

Figure 4.8: Modified PMOS Circuit Topology used for Measuring of MOSFET I-V Dependence

The decoupling capacitors across the supply consisted of ceramic and tantalum capacitors. However, the leakage current of a tantalum electrolytic capacitor potentially affected current measurements below 1 µA due to the capacitor series having leakage currents in the microamp range [10]. We removed tantalum capacitors to prevent any potential leakage currents. This did not fix the problem, so we collected more data with the tantalum capacitors removed.

Fig. 4.9 shows a voltage comparison between the PCB without tantalum capacitors and the breadboard test fixtures. Each voltage point comparison occurred at the same test current. In theory, the plot should be linear with a 1 to 1 slope.

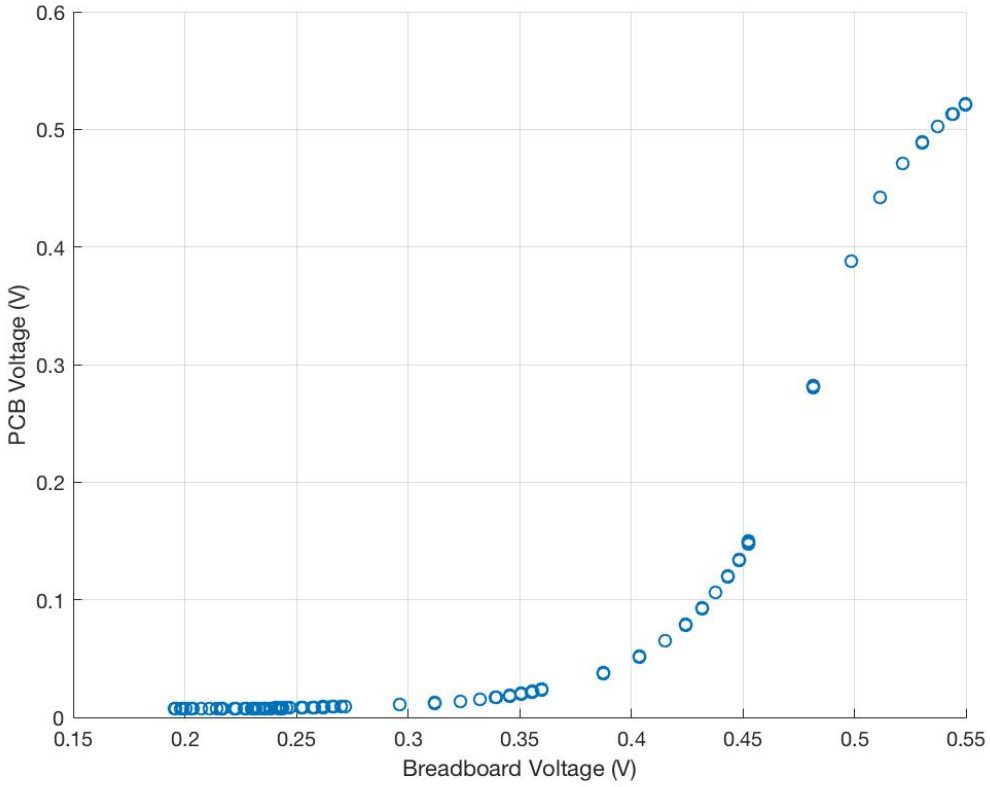

Figure 4.9: Comparing Voltage Measurements between Breadboard and PCB without Tantalum Capacitors for Currents between 100 pA **and** 900 nA

The rosin flux residue was the second suspected cause of altered measurements. The solder used to populate the board was 60/40 Kester 44 flux-cored wire, which uses a rosin-activated flux. Typically, ignoring flux residue resistance does not impact circuit performance, because the flux has a high resistance. The channel resistance, V_{DS}/I_D on the breadboard at 100 pA is 1.68 GΩ, but the measured resistance of the device on the PCB is 221.89 MΩ. According to the surface insulation resistivity test of Kester 44 flux-cored wire, the rosin activated flux residue has a nominal resistance of 220 MΩ [11]. The surface insulation resistivity test measures potential leakage paths between conductors [12]. The flux leakage causes the measured resistance to drop to 221.89 MΩ, because the flux impedance effectively connects in parallel to the channel resistance. This reduces the measured output voltage for a fixed test current.

After learning flux introduced current leakage, we removed the ceramic decoupling capacitors because of the possibility of flux getting trapped underneath the package. The ceramic capacitors slow the current source from reaching the specified target at lower currents, as $\frac{dv}{dt} = \frac{I}{C}$. With all the transistor arrays removed from the board, the LabVIEW acquisition program attempted to force currents ranging from 100 pA to 900 nA through the board with a 10.5 V voltage compliance limit. In theory, no current should pass through, but current still flowed. The current source had a voltage compliance limit set to 10.5 V. This means that the Keithley SourceMeter should have outputted 0 A at an output voltage of 10.5 V. Fig. 4.10 shows this behavior for an unpopulated PCB that never touched flux. In contrast, Fig. 4.11 shows the characteristics for a populated PCB with all transistor arrays and jumpers removed. The lower currents do not cause the supply to reach the compliance limit, implying current leakage somewhere on the board.

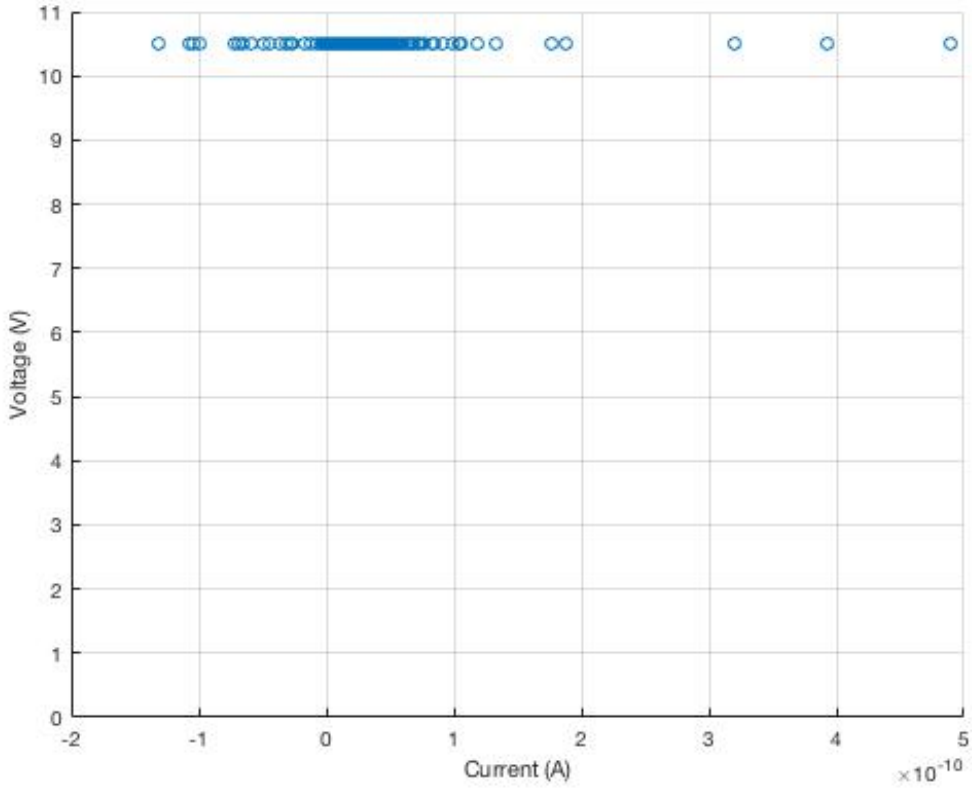

Figure 4.10: Low Current Measurements for Unpopulated PCB

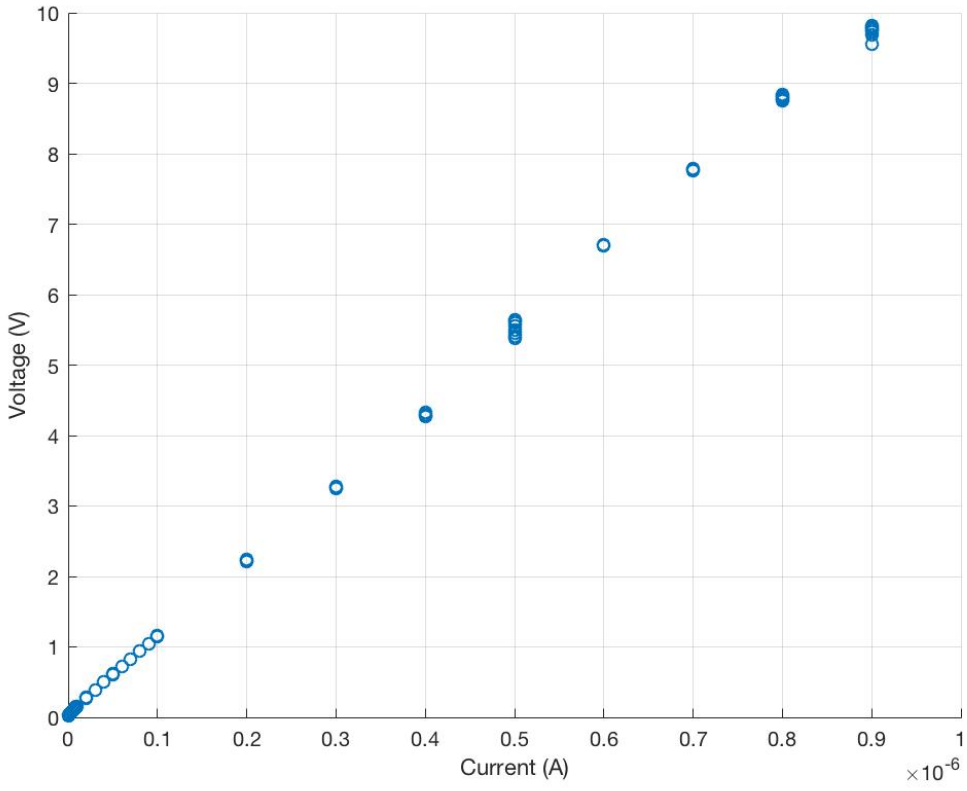

Figure 4.11: Low Current Measurements for Populated PCB with ICs Removed from Sockets

To combat this residue, we cleaned the board with 91% isopropyl alcohol. A Kimwipe, dipped in alcohol, scrubbed the board to take off the residue. Fig. 4.12 shows the board after the initial cleaning process. The board still had a haze after intense scrubbing with the alcohol-laced Kimwipe. It also had concentrations of flux residue around each solder joint. Removing transistor arrays for low current testing yields Fig. 4.13. This shows some improvement, as the supply now reaches the 10.5 V compliance limit at some low currents.

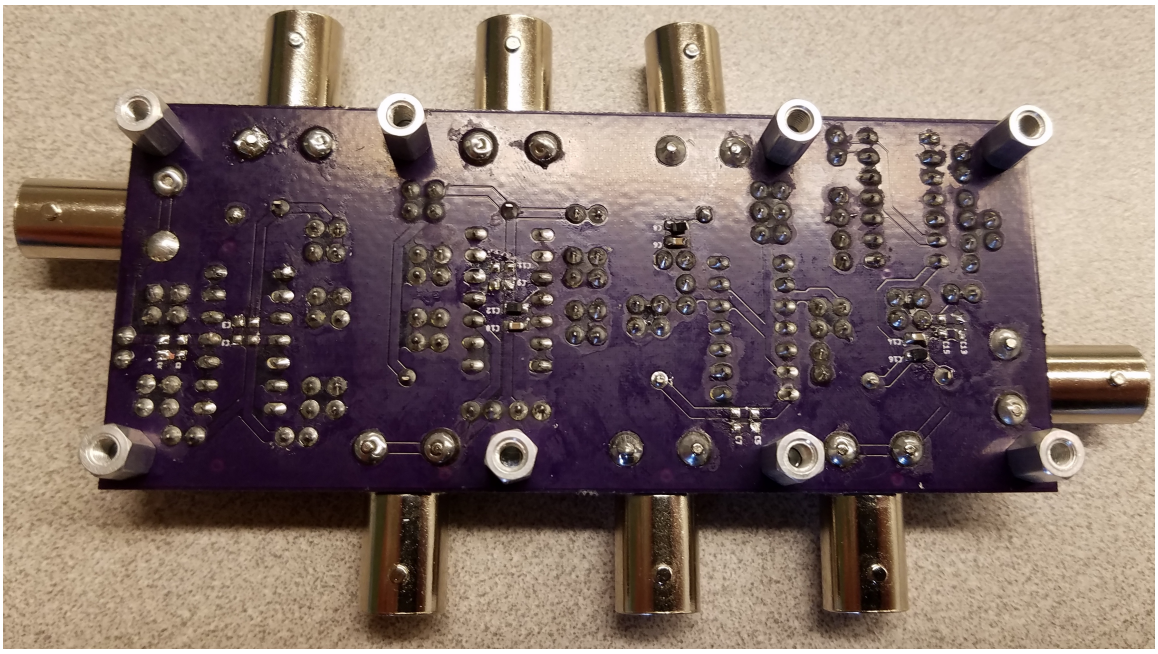

Figure 4.12: Dull Areas between Solder Joints Show Flux Residue after Initial Isopropyl Cleaning

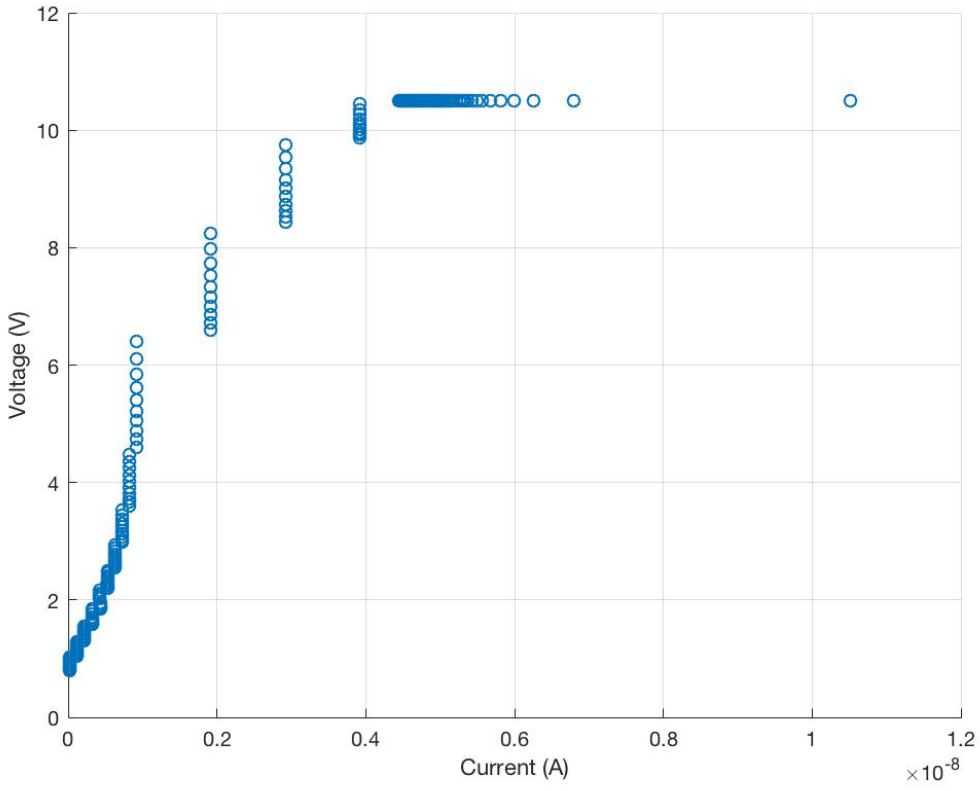

Figure 4.13: Low Current Measurements After Initial Isopropyl Cleaning Suggests Leakage Resistance of Approximately $2.5\,\mathrm{G}\Omega$

Another attempt to remove flux residue involved cleaning the board with M.G. Chemicals 4140 Flux Remover, a mixture of ethanol, isopropanol, and ethyl acetate. It cleaned the board without leaving as much residue on the board, as evidenced by the lessened haze. Fig. 4.14 shows the board after cleaning with flux remover. Another low current test, detailed in Fig. 4.15, reveals that the flux cleaner improves performance by reaching the voltage compliance limit at even lower currents. However, the flux cleaner failed to improve performance to that of the breadboard for currents below $1\,\mathrm{nA}$. This could stem from flux trapped underneath soldered components like header pins or chip sockets.

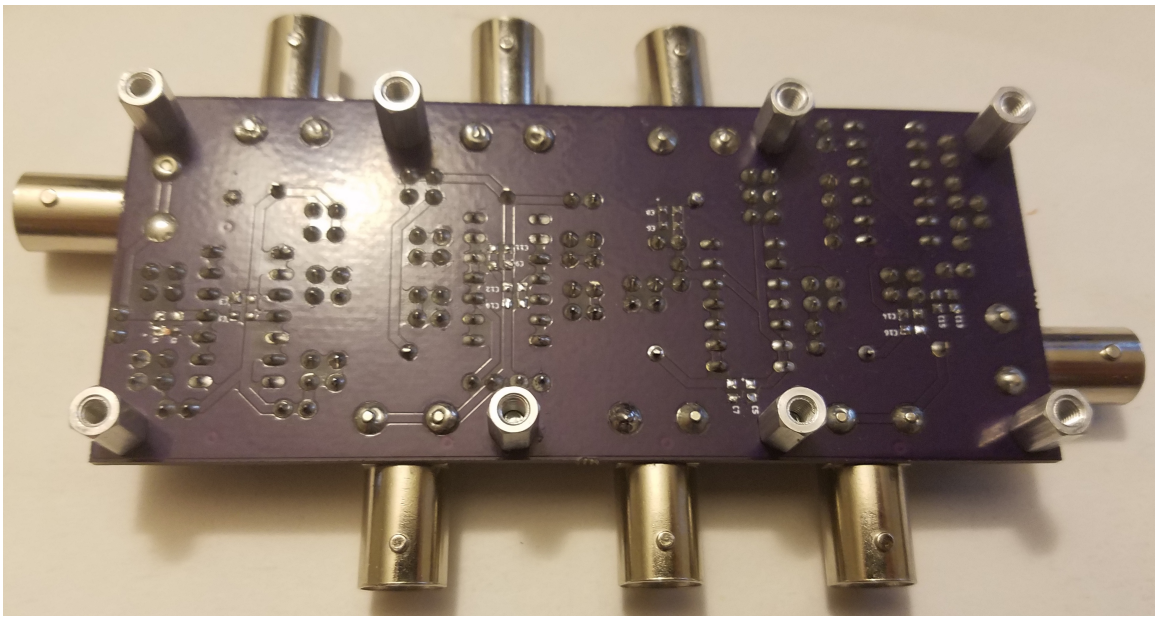

Figure 4.14: Flux Residue Reduced After Flux Remover Cleaning

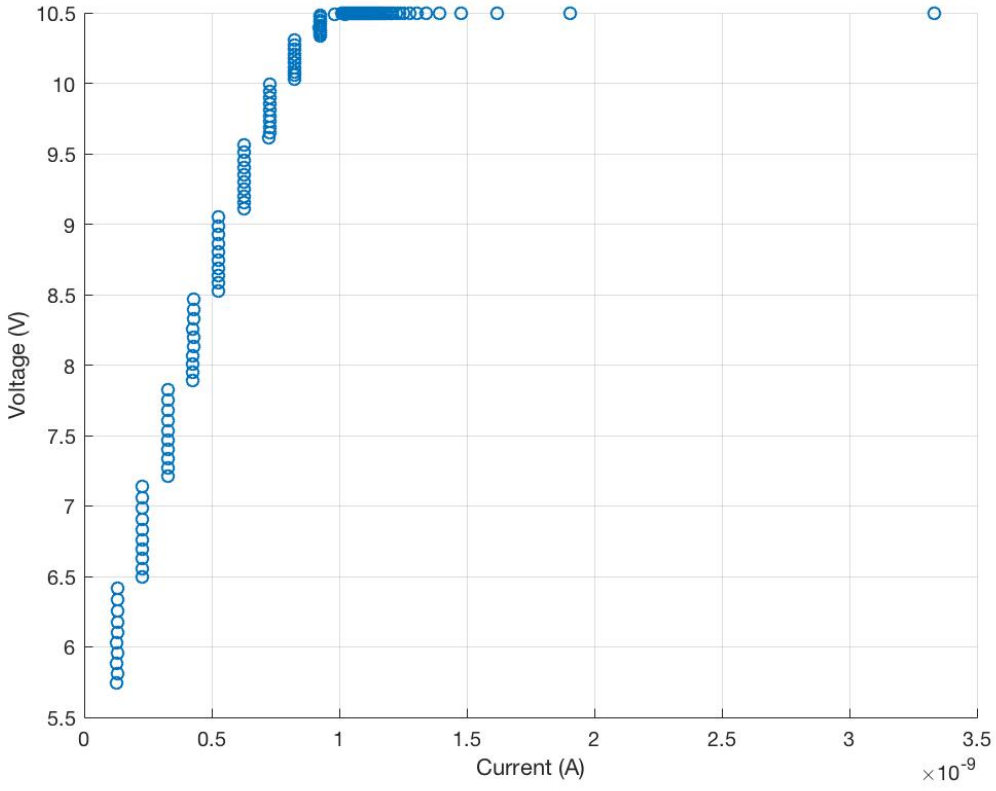

Figure 4.15: Low Current Measurements After Flux Remover Cleaning Suggests Leakage Resistance in Excess of $10\,\mathrm{G}\Omega$

Since rosin-activated flux leaves significant residue, we explored using solder without any solder flux. If populating the PCB required no flux, then the leakage effects cease being a problem. Attempting to populate the board with fluxless solder ended poorly with no good solder joints. The solder stuck to the solder mask instead of the metal pad. This showed that fluxless solder was not a viable option for populating the PCB and cannot bypass flux leakage current issues.

Chapter 5

RESULTS AND DISCUSSION

5.1 Performance of Original and Modified ACM

As seen in Table 5.1 and 5.2, we achieved the best fit between experimental values and modified ACM with m values of approximately 0.6. This holds true for both the n-channel and the p-channel devices. The exponent value changes little with change in tuning strategy and cost function.

Table 5.1: MODEL PARAMETERS AND PERFORMANCE OF MODIFIED ACM EXTRACTED FROM ALD1106 MEASUREMENTS

Tuning Scheme		**V_{norm}**	**I_{norm}**	**m**	**g_m Error, %**		**V_{GS} Error, %**	
Domain	Cost Func.	[mV]	[µA]	-	max	average	max	average
g_m	l_∞	42.0	1.85	0.611	12.1	5.4	3.53	0.17
g_m	l_1	39.7	1.27	0.581	24.8	3.9	2.37	0.14
V	l_∞	40.6	1.85	0.601	26.4	9.7	0.79	0.14
V	l_1	40.1	1.47	0.586	27.9	5.5	1.39	0.13

Table 5.2: MODEL PARAMETERS AND PERFORMANCE OF MODIFIED ACM EXTRACTED FROM ALD1107 MEASUREMENTS

Tuning Scheme		$\mathbf{V_{norm}}$	$\mathbf{I_{norm}}$	**m**	$\mathbf{g_m}$ **Error, %**		$\mathbf{V_{GS}}$ **Error, %**	
Domain	Cost Func.	[mV]	[µA]	-	max	average	max	average
g_m	l_∞	33.0	0.307	0.586	10.6	7.8	2.43	0.15
g_m	l_1	35.0	0.408	0.586	19.1	3.0	0.83	0.10
V	l_∞	38.1	0.509	0.596	22.2	3.7	0.59	0.09
V	l_1	36.8	0.509	0.601	19.4	3.3	0.72	0.08

Fig. 5.1 and 5.2 show that a model with m=0.5 does not fit for large currents. This finding proves surprising, considering the devices have a long channel length of 7.8 µm. The poor fit for m=0.5 and good modified ACM fit verify the behavior in (1.6), which states the carrier mobility decreases for larger drain-source voltages.

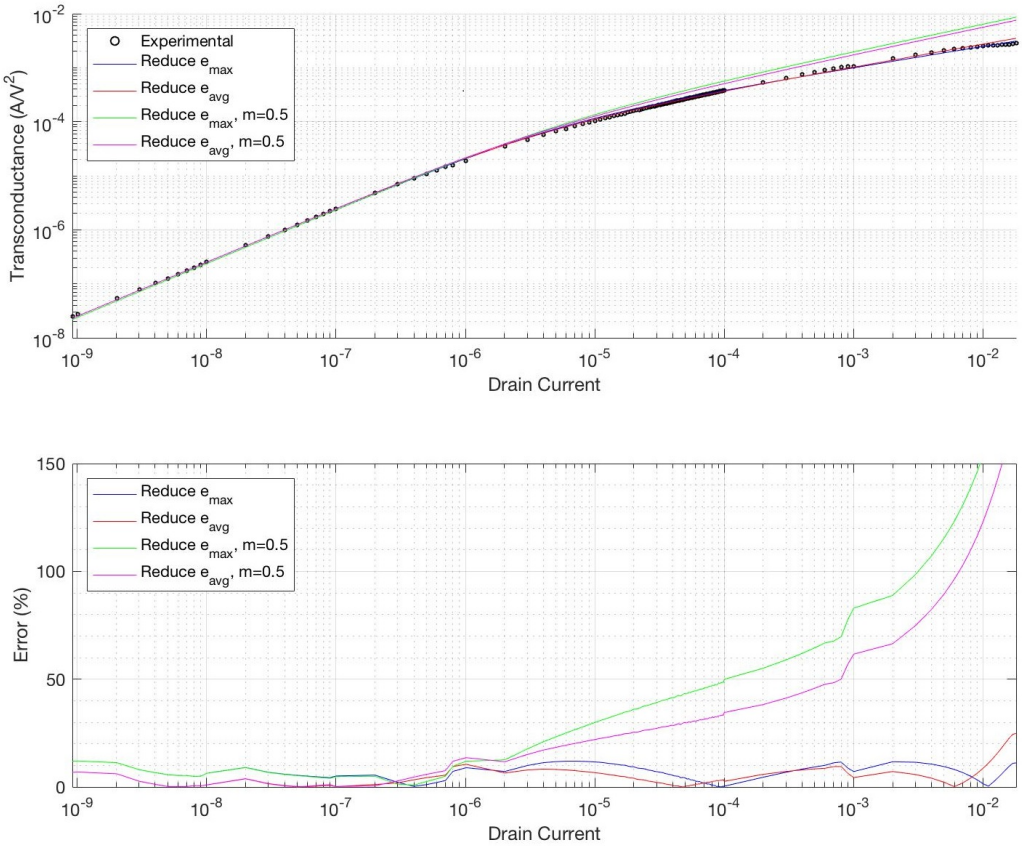

Figure 5.1: ALD1106 experimental transconductance compared to modified ACM extracted using g_m-domain tuning and two different cost functions. The graphs also show the poor fit provided by the ACM (m=0.5).

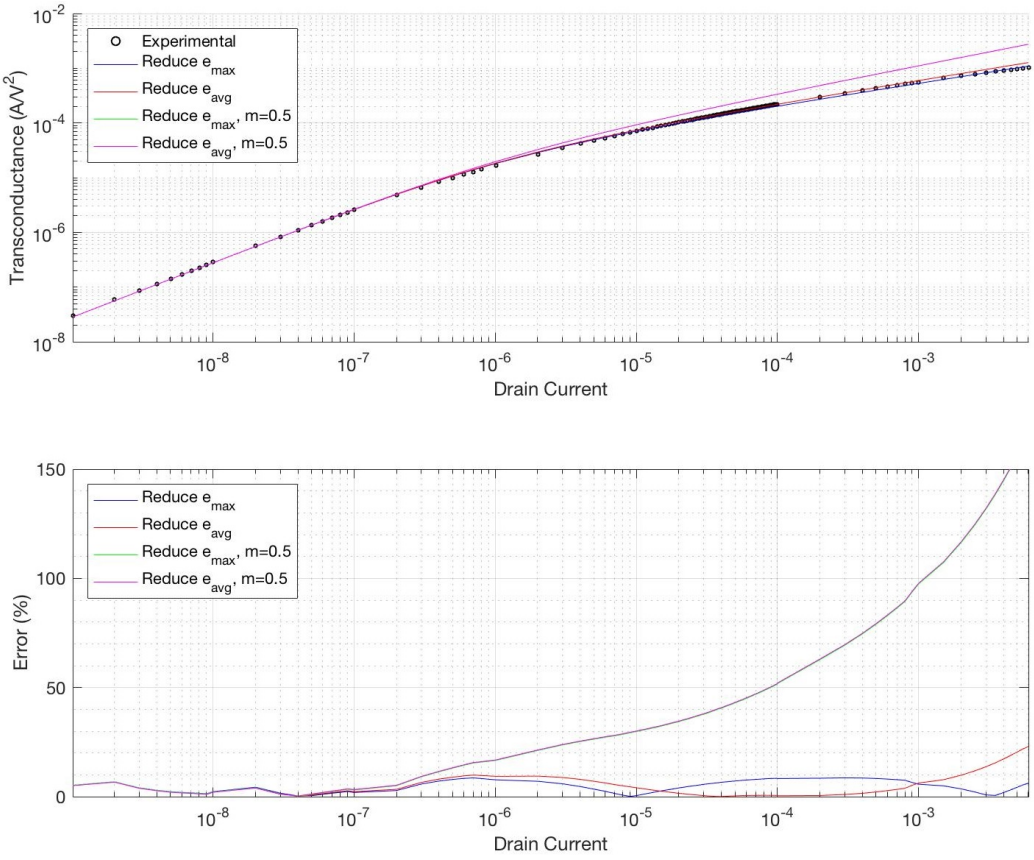

Figure 5.2: ALD1107 experimental transconductance compared to modified ACM extracted using g_m-domain tuning and two different cost functions. The graphs also show the poor fit provided by the ACM (m=0.5).

5.2 Impact of Cost Function

Fig. 5.1 and 5.2 show that minimizing the mean error, i.e. using an l_1-based cost function, provides a better fit for lower current data points. This stems from a higher concentration of data points at lower currents that shifts the model fit to optimize lower currents. The higher current points suffer more error. Conversely, reducing the maximum error (l_∞) provides a better fit for higher current data points at the expense of lower current points. In absence of outliers, the discrepancies between the two cost functions should diminish, if the density of the points becomes more uniform.

5.3 Impact of Tuning Domain

Table 5.1 and 5.2 show the tuning domain (see Fig. 3.1 and Fig. 3.2) in this particular study has little impact upon optimum models found. General trends are intuitive and summarize as follows: optimizing in one domain sacrifices performance in the other domain. However, the differences are insignificant. We note that irrespective of tuning domain used, the V_{GS} error is small. Maximum error never exceeds 3.6%, and average error stays below 0.2%. The match between experimental g_m and modified ACM model typically has maximum error of 20-25% and an average of sub-10%. This performance proves superior to the performance of the original ACM, where the maximum error exceeds 150%.

5.4 Differences Between N and P-Channel Devices

Tables 5.1 and 5.2 show differences between NMOS and PMOS devices. The devices have similar normalization voltages and m parameters, but their normalization currents are vastly different. ALD1106 has much larger I_{norm}. This larger I_{norm} does not stem from a wider n-channel transistor, as both devices have 138 µm widths [5]. Referring to (1.3), the higher mobility of electrons and the slightly larger n factor of ALD1106 explains the larger normalization current of ALD1106. The larger n follows from the larger V_{norm}.

5.5 Repeatability of Model Parameters

Each tested transistor has a specific generated $\{m, V_{norm}, I_{norm}\}$ set. This begs the question of whether the modified ACM applies to every transistor array within a certain product category. We performed the model fitting process to each transistor on three ALD1106 and three ALD1107 chips, a total of 24 transistors. For the ALD1106

and ALD1107 chips, their datasheets index each device with a number [7], [6]. Tables 5.3 and 5.4 show statistics for devices one through four, as well as statistics for the aggregation of all devices. For example, the statistics for transistor one provides the mean and standard deviation derived from the transistor one of each chip.

The statistical analysis shows that the model's repeatability, as the standard deviation never exceeds 12% of the mean. The standard deviation typically stays under 5% of the mean for most parameters. The normalization current has the highest deviation from the mean around 10%, while the normalization voltage and m exponent typically stays below 1% of the mean. The modified ACM proves applicable to entire product families of integrated circuits.

Table 5.3: ALD1106 Repeatability Results

	V_{norm} [mV]		I_{norm} [µA]		m	
	μ	σ	μ	σ	μ	σ
N1	39.53	0.231	1.27	0.058	0.577	0.0029
N2	39.53	0.577	1.31	0.058	0.583	0.0058
N3	39.53	0.577	1.27	0.058	0.579	0.0058
N4	39.83	0.289	1.30	0.055	0.579	0.0029
All N	39.61	0.211	1.29	0.052	0.580	0.0044

Table 5.4: ALD1107 Repeatability Results

	V_{norm} [mV]		I_{norm} [µA]		m	
	μ	σ	μ	σ	μ	σ
P1	35.93	0.351	0.425	0.031	0.584	0.0077
P2	35.77	0.153	0.422	0.026	0.584	0.0077
P3	36.00	0.520	0.437	0.052	0.586	0.0101
P4	36.07	0.551	0.437	0.052	0.586	0.0101
All P	35.94	0.380	0.431	0.036	0.585	0.0077

Chapter 6

APPLYING THE MODIFIED ACM TO DIGITAL-OPTIMIZED DEVICES

6.1 Reasons for Experimenting on Digitally Optimized Integrated Circuits

Where transconductance proves vital for analog designs, the modified ACM provides accurate characterization for digital domain optimized MOSFETs. The CD4007 find applications in many digital circuits as CMOS inverters but are also appropriate for high input impedance amplifiers [13]. It also has an estimated channel length of 10 µm, which is similar in length to the ALD1106 and ALD1107 [14], [5]. As stated in the introduction, the transconductance is important for finding appropriate amplifier gain and biasing. We explored whether the modified ACM would apply to MOSFETs optimized for digital use.

6.2 CD4007 Input Protection Network

While the ALD1106 and ALD1107 are standalone transistors, the CD4007 has input protection circuitry. However, the input protection network would never see inappropriate voltages through the voltage compliance limits on the Keithley, as well as the diode connecting of the transistor [13].

6.3 Results

Fig. 6.1 and 6.2 show that the modified ACM improves on the original ACM for digital optimized chips as well. The normalization voltages change due to digital integrated circuits optimizing the diode ideality factor differently, as designers attempt to limit

subthreshold currents for lower power consumption. The normalization current also changes. While the input protection network never conducts in the test configuration, the presence of additional passive components might skew the current where the device tranisitions from weak to strong inversion. However, the normalization voltage stays in the millivolt range, and the normalization current stays in the microampere range.

Similarly, the ACM and modified ACM both characterize lower current operation well, but when drain current increases, the modified ACM characterizes lateral field effect behavior better. Furthermore, the m exponent stays around 0.6, which is similar to the analog-optimized ALD1106 and ALD1107's m exponents. This stems from both transistor types having similar channel lengths around 10 µm, reflecting the m exponent's indication of high-field effects as a function of channel length.

Table 6.1: MODEL PARAMETERS AND PERFORMANCE OF MODIFIED ACM EXTRACTED FROM CD4007 NMOS MEASUREMENTS

Tuning Scheme		$\mathbf{V_{norm}}$	$\mathbf{I_{norm}}$	**m**	$\mathbf{g_m}$ **Error, %**		$\mathbf{V_{GS}}$ **Error, %**	
Domain	Cost Func.	[mV]	[µA]	-	max	average	max	average
g_m	l_∞	62.4	4.38	0.586	7.06	2.53	0.703	0.0529
g_m	l_1	64	4.76	0.586	9.48	2.21	0.459	0.043
V	l_∞	65.6	6.22	0.606	11.6	4.87	0.215	0.039
V	l_1	65.3	6.12	0.606	11.25	4.65	0.263	0.038

Table 6.2: MODEL PARAMETERS AND PERFORMANCE OF MODIFIED ACM EXTRACTED FROM CD4007 PMOS MEASUREMENTS

Tuning Scheme		$\mathbf{V_{norm}}$	$\mathbf{I_{norm}}$	**m**	$\mathbf{g_m}$ **Error, %**		$\mathbf{V_{GS}}$ **Error, %**	
Domain	Cost Func.	[mV]	[µA]	-	max	average	max	average
g_m	l_∞	36.1	1.22	0.571	6.06	2.69	0.481	0.036
g_m	l_1	35.9	1.32	0.576	6.92	1.55	0.234	0.030
V	l_∞	37.7	1.72	0.591	9.97	3.64	0.135	0.026
V	l_1	36.8	1.62	0.591	7.77	2.97	0.264	0.025

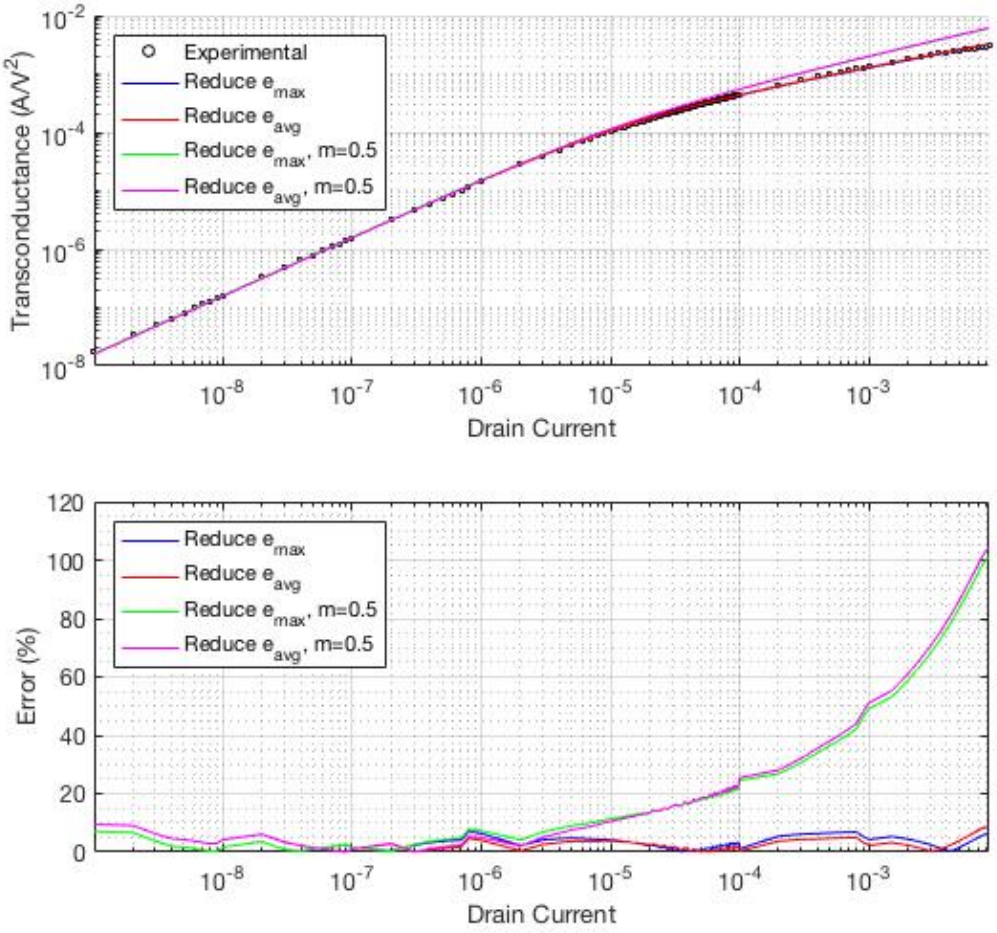

Figure 6.1: CD4007 NMOS experimental transconductance compared to modified ACM extracted using g_m-domain tuning and two different cost functions. The graphs also show the poor fit provided by the ACM (m=0.5).

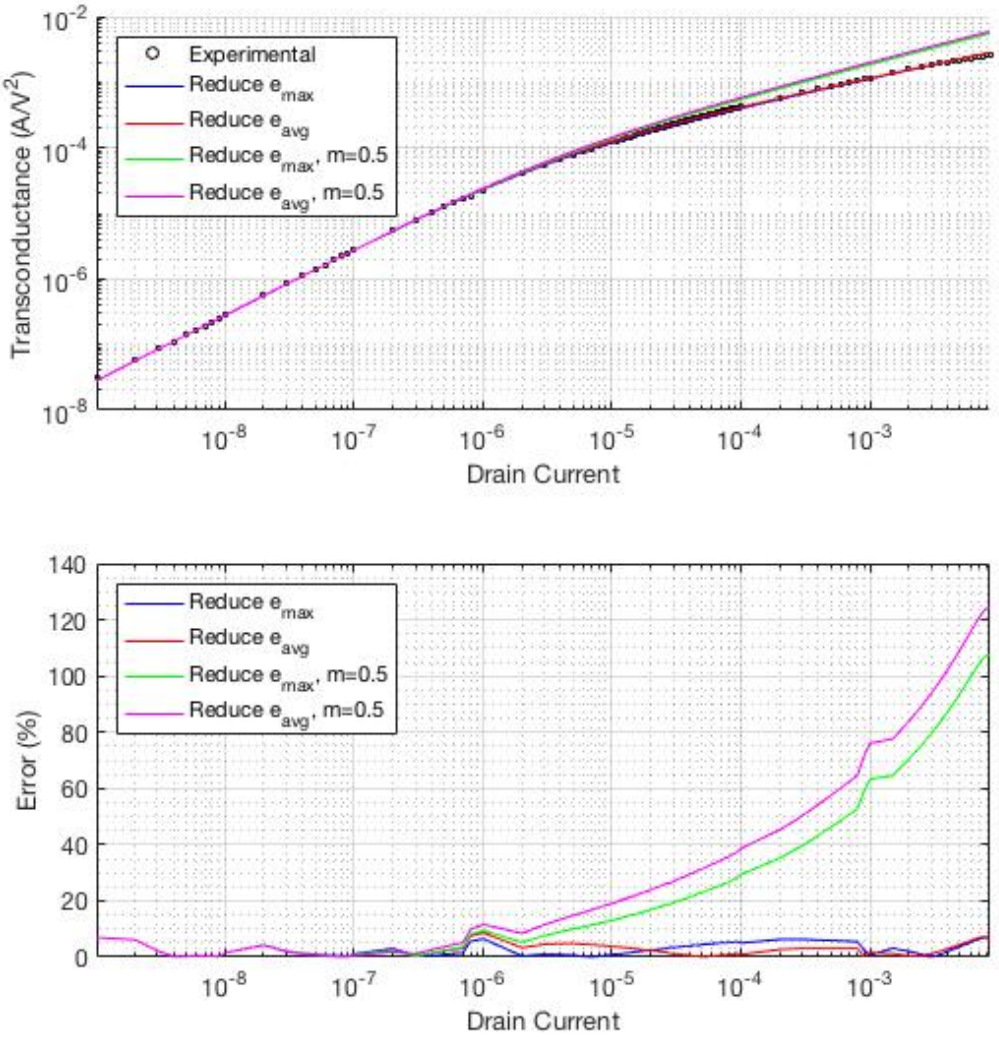

Figure 6.2: CD4007 PMOS experimental transconductance compared to modified ACM extracted using g_m-domain tuning and two different cost functions. The graphs also show the poor fit provided by the ACM (m=0.5).

6.4 Repeatability

The statistical analysis shows that the model is repeatable for digitally optimized circuitry, as the standard deviation never exceeds 10% of the mean. The standard deviation typically stays under 5% of the mean for most parameters. The normaliza-

tion current has the highest deviation from the mean around 10% of the mean, while the normalization voltage and m exponent typically stays below 1% of the mean.

Table 6.3: CD4007 Repeatability Results

	V_{norm} [mV]		I_{norm} [µA]		m	
	μ	σ	μ	σ	μ	σ
N1	63.47	0.289	5.13	0.463	0.589	0.0058
N2	61.87	0.577	5.16	0.437	0.608	0.0029
N3	62.43	0.839	5.12	0.475	0.601	0.0133
All N	62.59	0.831	5.14	0.397	0.599	0.0110
P1	35.57	0.231	1.39	0.059	0.579	0.0029
P2	35.67	0.115	1.42	0.00098	0.586	0.0000
P3	35.43	0.153	1.42	0.00085	0.586	0.0000
All P	35.56	0.181	1.41	0.034	0.584	0.0037

Chapter 7

FUTURE WORK

7.1 No-Clean Flux Solder as an Alternative

Since the populating the board needs solder with flux, a no-clean flux solder alternative could work. The Kester 275 flux-cored wire has a better surface resistance rating, but the problem remains that even no-clean flux residue generates leakage currents [15], [16]. Testing this in the future could reveal if 100 pA range measurements are viable.

7.2 Including Substrate Biasing in the Modified ACM

The modification to the ACM only applies to transistors without substrate biasing. However, certain applications necessitate substrate biasing between the MOSFET's body and source terminals. Preliminary testing implies the substrate biasing shifts the current where the transistor enters strong inversion. It also seems to affect the rate at which the transistor transitions from weak to strong inversion. The modified model could expand to include a substrate biasing term.

Chapter 8

CONCLUSION

This study shows the 'deviation from square-law' applies to more than sub-micrometer MOSFETs. As seen with the ALD1106, ALD1107, and CD4007, a device with lengths between 7.8 µm and 10 µm could exhibit 'high-field' effects sufficient to invalidate the ACM model. Furthermore, device optimization for either analog or digital applications does not affect the relationship between high-field effects and channel length. More importantly, the experimental results corroborate the argument presented in [4]. Namely, a simple modification to the ACM captures the I-g_m dependence of practical MOSFETs. The introduced exponent m, a 'catch-all' parameter, accounts for many phenomena that shape the I-g_m dependence at high currents. These are strong vertical and lateral fields and finite source-side contact resistance among others. As such, the modified ACM offers analog designers an insight into the overall behavior of the device and facilitates biasing decisions.

BIBLIOGRAPHY

[1] A. I. A. Cunha, M. C. Schneider, and C. Galup-Montoro, "An MOS transistor model for analog circuit design," *IEEE J. Solid-State Circuits*, vol. 33, no. 10, pp. 1510–1519, October 1998.

[2] P. R. Gray, P. J. Hurst, S. H. Lewis, and R. G. Meyer, *Analysis and Design of Analog Integrated Circuits*, 4th ed. New York, New York: John Wiley & Sons, 2001, p. 59.

[3] S. Dimitrijev, *Principles of Semiconductor Devices*, 2nd ed. New York, New York: Oxford University Press, Inc., 2012, pp. 124, 323–324.

[4] V. I. Prodanov, "Empirical model for the transconductance-current dependence of short-channel MOSFETs," in *2012 IEEE 55th International Midwest Symposium on Circuits and Systems (MWSCAS)*, Boise, Idaho, 2012, pp. 290–293.

[5] Advanced Linear Devices, "ALD 11xx MOSFET PSPICE MODEL," Feb. 2004. [Online]. Available: http://aldinc.com/pdf/Model_ALDMOSFET.zip

[6] ——, "QUAD/DUAL P-CHANNEL MATCHED PAIR MOSFET ARRAY," 2012, ALD 1107 datasheet. [Online]. Available: http://www.aldinc.com/pdf/ALD1117.pdf

[7] ——, "QUAD/DUAL N-CHANNEL MATCHED PAIR MOSFET ARRAY," 2012, ALD 1106 datasheet. [Online]. Available: http://www.aldinc.com/pdf/ALD1116.pdf

[8] J. Qiu, "Interpolation by Splines," April 2013, quadratic spline interpolation

background. [Online]. Available:
https://www.math.uh.edu/~jingqiu/math4364/spline.pdf

[9] D. C. Lay, S. R. Lay, and J. J. McDonald, *Linear Algebra and Its Applications*, 5th ed. Pearson Education, Inc., 2016, p. 379.

[10] AVX Corporation, "F98 Series," January 2018, tantalum capacitor datasheet. [Online]. Available: http://datasheets.avx.com/F98.pdf

[11] Kester, "44 flux-cored wire," kester 44 Flux-Cored Wire datasheet. [Online]. Available: https://www.kester.com/DesktopModules/Bring2mind/DMX/Download.aspx?Command=Core_Download&EntryId=1072&language=en-US&PortalId=0&TabId=96

[12] P. Kinner, "The principle of surface insulation resistance (sir) testing and its role in establishing the electrochemical reliability of a printed circuit board," in *Proceedings of 2004 International Conference on the Business of Electronic Product Reliability and Liability (IEEE Cat. No.04EX809)*, Apr 2004, pp. 3–8.

[13] Texas Instruments, "CD4007UB Types, CMOS Dual Complementary Pair Plus Inverter," September 2003, CD4007 datasheet. [Online]. Available: http://www.ti.com/lit/ds/symlink/cd4007ub.pdf

[14] L. Fuller, "SPICE Model for NMOS and PMOS FETs in the CD4007 Chip," August 2015, CD4007 size analysis. [Online]. Available: https://people.rit.edu/lffeee/CD4007_SPICE_MODEL.pdf

[15] Kester, "275 flux-cored wire," kester 275 Flux-Cored Wire datasheet. [Online]. Available: https://www.kester.com/DesktopModules/Bring2mind/DMX/Download.aspx?Command=Core_Download&EntryId=978&language=en-US&PortalId=0&TabId=96

[16] M. Bixenman, M. McMeen, and B. Tolla, "WHY CLEAN A NO-CLEAN FLUX," in *Proceedings of the International Conference on Soldering and Reliability (ICSR) 2016*. SMTA, 2016.

[17] M.G. Chemicals, "Flux Remover for PC Boards 4140 Technical Data Sheet," March 2017, Flux Remover datasheet. [Online]. Available: https://www.mgchemicals.com/downloads/tds/tds-4140-l.pdf

APPENDICES

Appendix A

QUADRATIC SPLINE DIFFERENTIATION DERIVATION

A.1 Derivation

Fig. A.1 plots three data points (x_1,y_1), (x_2,y_2), and (x_3,y_3).

These data points follow some function, f(x). This f(x) could decompose into a sum of two functions.

$$f(x) = m(x) + b(x) \tag{A.1}$$

The line segment between (x_1,y_1) and (x_3,y_3) describes the first function, m(x).

$$m(x) = y_1 + \frac{y_3 - y_1}{x_3 - x_1}x \tag{A.2}$$

The second function, b(x), corresponds to the difference between the actual and line segment value at a given x. This function takes the form:

$$b(x) = a(x - x_1)(x - x_3) \tag{A.3}$$

This ensures a difference contribution of zero for points on the original function.

Since a derivative operation serves as a linear operation, the derivative of any point on the original function should be:

$$\frac{df(x)}{dx} = \frac{dm(x)}{dx} + \frac{db(x)}{dx} \tag{A.4}$$

$$\frac{dm(x)}{dx} = \frac{y_3 - y_1}{x_3 - x_1} \tag{A.5}$$

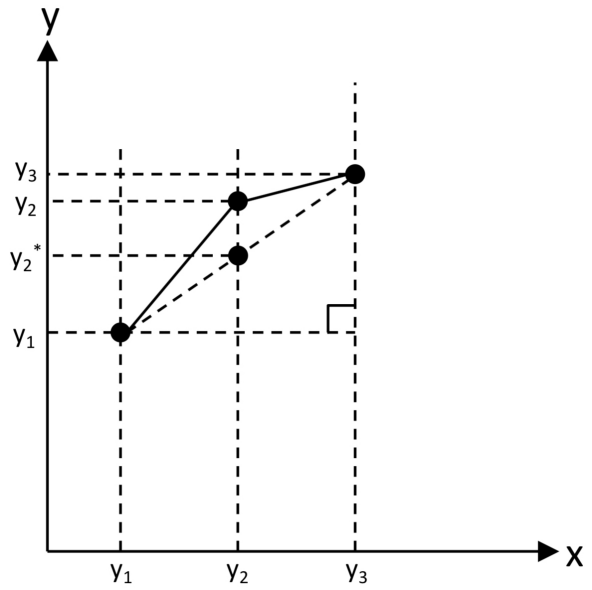

Figure A.1: Three Arbitrary Data Point Example

$$\frac{db(x)}{dx} = a(2x - (x_1 + x_3)) \tag{A.6}$$

In order to write an equation for the derivative of the original function, we need to solve for a. Since a is a part of the equation for b(x), we can solve for a by leveraging this relationship:

$$b(x_2) = y_2 - y_2^* \tag{A.7}$$

This implies that a positive $b(x_2)$ means that the actual point is greater than the main function point and vice versa.

Using the similar triangle theorem, we can find an equation for y_2^*:

$$\frac{y_3 - y_1}{x_3 - x_1} = \frac{y_2^* - y1}{x_2 - x_1}$$

$$y_2^* = \frac{(y_3 - y_1)(x_2 - x_1)}{x_3 - x_1}$$

$$a(x_2 - x_1)(x_2 - x_3) = (y_2 - y_1) - \frac{(y_3 - y_1)(x_2 - x_1)}{x_3 - x_1}$$

$$a = \frac{y_2 - y_1}{(x_2 - x_1)(x_2 - x_3)} - \frac{y_3 - y_1}{(x_3 - x_1)(x_2 - x_3)}$$

Substituting that equation into (A.3) and (A.7) yields an equation for a:

$$b(x_2) = (y_2 - y_1) - \frac{(y_3 - y_1)(x_2 - x_1)}{x_3 - x_1}$$

$$a(x_2 - x_1)(x_2 - x_3) = (y_2 - y_1) - \frac{(y_3 - y_1)(x_2 - x_1)}{x_3 - x_1}$$

$$a = \frac{y_2 - y_1}{(x_2 - x_1)(x_2 - x_3)} - \frac{y_3 - y_1}{(x_3 - x_1)(x_2 - x_3)}$$

This yields a complete equation for the derivative:

$$\frac{df(x)}{dx} = \frac{y_3 - y_1}{x_3 - x_1} + \left(\frac{y_2 - y_1}{(x_2 - x_1)(x_2 - x_3)} - \frac{y_3 - y_1}{(x_3 - x_1)(x_2 - x_3)} (2x - (x_1 + x_3)) \right) \tag{A.8}$$

Since we only care about the derivative at x_2,

$$\frac{df(x_2)}{dx} = \frac{y_3 - y_1}{x_3 - x_1} + \left(\frac{y_2 - y_1}{(x_2 - x_1)(x_2 - x_3)} - \frac{y_3 - y_1}{(x_3 - x_1)(x_2 - x_3)} (2x_2 - (x_1 + x_3)) \right) \tag{A.9}$$

This derivative solution works for all sets of three data points. These data points do not have to monotonically increase or decrease. The second point could be much lower or higher than the first and last points. This stems from y_2^* determining the difference contribution calculation, which is always on the linear line segment between the first and last points.

A.2 Verification

Assuming the points (1,1), (2.2,2), and (3.3) exist in a 2D Cartesian plane, we get the following from (A.5) and (A.6).

$$\frac{dm(x)}{dx} = 1, \frac{db(x)}{dx} = 0.083$$

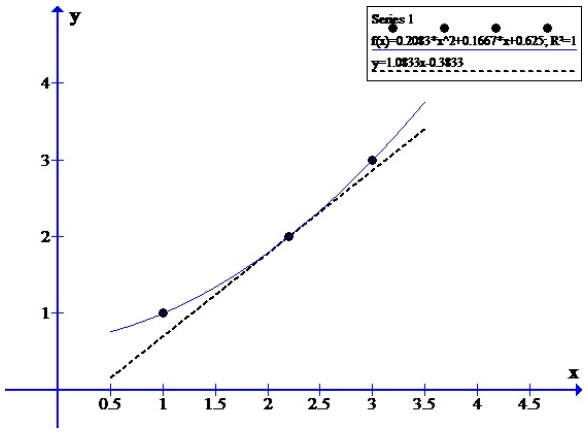

Figure A.2: Three Arbitrary Data Point Example with Midpoint Below Linear Segment

The calculation agrees with data obtained from the Graph plotting program. Graph allows you to input data points, curve-fit, and produce a tangent line to a smooth curve.

Moving x_2 away from the middle increases the slope irrespective of the direction of the shift. In other words, the "extra slope" is always positive, as seen below.

Assuming (1,1), (1.8,2), and (3,3), we get exactly the same values:

$$\frac{dm(x)}{dx} = 1, \frac{db(x)}{dx} = 0.083$$

Graph confirms this property.

A.3 Simplification

The derived equation in (A.9) distills down to a linear combination of the finite differences between the points. Rewriting the equation yields the following.

$$\begin{aligned}
\frac{df(x_2)}{dx} &= \frac{y_3 - y_1}{x_3 - x_1} - \frac{x_3 - x_2 + x_1 - x_2}{x_3 - x_2}\left(\frac{y_3 - y_1}{x_3 - x_1} - \frac{y_2 - y_1}{x_2 - x_1}\right) \\
&= \frac{y_3 - y_1}{x_3 - x_1} - \left(1 + \frac{x_1 - x_2}{x_3 - x_2}\right)\left(\frac{y_3 - y_1}{x_3 - x_1} - \frac{y_2 - y_1}{x_2 - x_1}\right) \\
&= \frac{y_2 - y_1}{x_2 - x_1} - \frac{(x_1 - x_2)(y_3 - y_1)}{(x_3 - x_2)(x_3 - x_1)} + \frac{(x_1 - x_2)(y_2 - y_1)}{(x_3 - x_2)(x_2 - x_1)}
\end{aligned}$$

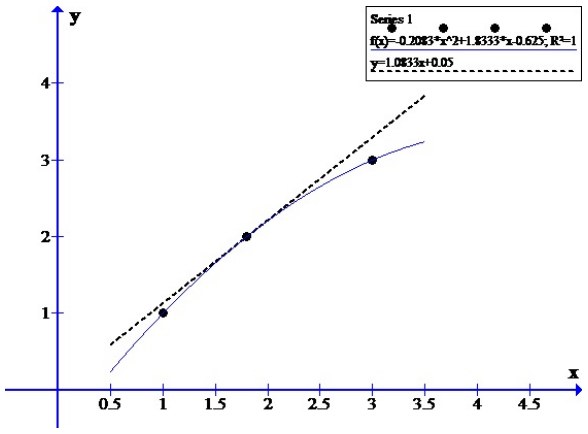

Figure A.3: Three Arbitrary Data Point Example with Midpoint Above Linear Segment

Through partial fraction expansion:

$$\frac{(x_1 - x_2)(y_3 - y_1)}{(x_3 - x_2)(x_3 - x_1)} = \frac{A}{x_3 - x_2} + \frac{B}{x_3 - x_1}$$

$$(x_1 - x_2)(y_3 - y_1) = A(x_3 - x_1) + B(x_3 - x_2)$$

$$x_1(y_3 - y_1) - x_2(y_3 - y_1) = Ax_3 - Ax_1 + Bx_3 - Bx_2$$

$$A + B = 0, A = y_3 - y_1, B = y_3 - y_1$$

$$\frac{(x_1 - x_2)(y_2 - y_1)}{(x_3 - x_2)(x_2 - x_1)} = \frac{A}{(x_3 - x_2)} + \frac{B}{(x_2 - x_1)}$$

$$x_1(y_2 - y_1) - x_2(y_2 - y_1) = A(x_2 - x_1) + B(x_3 - x_2)$$

$$B = 0, A = -(y_2 - y_1)$$

Using the results of the partial fraction expansion,

$$\begin{aligned}\frac{df(x_2)}{dx} &= \frac{y_2 - y_1}{x_2 - x_1} + \frac{y_3 - y_1}{x_3 - x_2} - \frac{y_3 - y_1}{x_3 - x_1} - \frac{y_2 - y_1}{x_3 - x_2} \\ &= \frac{y_2 - y_1}{x_2 - x_1} + \frac{y_3 - y_1 - y_2 + y_1}{x_3 - x_2} - \frac{y_3 - y_1}{x_3 - x_1} \\ &= \frac{y_2 - y_1}{x_2 - x_1} + \frac{y_3 - y_2}{x_3 - x_2} - \frac{y_3 - y_1}{x_3 - x_1}\end{aligned} \tag{A.10}$$

For finding the transconductance as described in 1.1, the following equation yields

$$g_m = \frac{I_2 - I_1}{V_2 - V_1} + \frac{I_3 - I_2}{V_3 - V2} - \frac{I_3 - I_1}{V_3 - V_1} \tag{A.11}$$

This shows that the quadratic spline differentiation first adds the forward and backward differences. Then, it subtracts the finite difference between the outer points from the sum. The higher order derivative consists of a linear combination of first order differences. This differentiation scheme yields zero error for linear or quadratic points.

A.4 Obtaining Voltage from Transconductance

A.4.1 Deriving the Voltage Expression

Given a set of g_m-I points, we wanted to obtain all other voltage values in the V-I set.

A quadratic that passes through the first point (V_1,I_1) has the following general expression:

$$I = I_1 + a_1(V - V_1) + a_2(V - V_1)^2$$

The derivative of A.4.1, g_m, is:

$$g_m = a_1 + 2a_2(V - V_1) \tag{A.12}$$

Evaluating A.12 at V_1 and V_2, we have:

$$g_{m1} = a_1$$

$$g_{m2} = a_1 + 2a_2(V_2 - V_1)$$

These two expressions allow us to relate a_1 and a_2 to g_{m1}, g_{m2}, and the difference $V_2 - V_1$.

$$a_1 = g_{m1}$$

$$a_2 = \frac{1}{2}\frac{g_{m2} - g_{m1}}{V_2 - V_1}$$

Substituting in A.4.1, we get:

$$I = I_1 + g_{m1}(V - V_1) + \frac{1}{2}\frac{g_{m2} - g_{m1}}{V_2 - V_1}(V - V_1)^2 \tag{A.13}$$

Since A.13 must also pass through the second point, V_2 and I_2 must relate as:

$$I_2 = I_1 + g_{m1}(V_2 - V_1) + \frac{g_{m2} - g_{m1}}{2(V_2 - V_1)}(V_2 - V_1)^2$$

This leads to the following simple expression for V_2.

$$V_2 = 2\frac{I_2 - I_1}{g_{m2} + g_{m1}} + V_1$$

The last equation generalizes to:

$$V_{k+1} = 2\frac{I_{k+1} - I_k}{g_{m(k+1)} + g_{m(k)}} \tag{A.14}$$

A.4.2 Proving the Voltage Expression Relates to Transconductance Equation

Using A.14, consider the points from Fig. A.1. The x-axis shows voltage, and the y-axis shows current. The derivatives at these points yield g_{m1}, g_{m2}, and g_{m3}. If all three points belong to the same quadratic curve, the following must be true.

$$g_{m2} + g_{m1} = 2\frac{I_2 - I_1}{V_2 - V_1} \tag{A.15}$$

$$g_{m3} + g_{m2} = 2\frac{I_3 - I_2}{V_3 - V_2} \tag{A.16}$$

$$g_{m3} + g_{m1} = 2\frac{I_3 - I_1}{V_3 - V_1} \tag{A.17}$$

Summing A.15 and A.16, as well as subtracting A.17, produces the following expression for the transconductance at the mid-point, the same expression as A.10. This shows the direct relationship between A.10 and A.14.

Appendix B

MATLAB ALGORITHM DESIGN

B.1 Model Fitting Algorithm

```
function [ Inorm_opt,Vnorm_opt,m_opt,e_max_gm,e_avg_gm,e_max_v,e_avg_v] = ...
    gm_model_fit( I,V,lowILim,minVt,m_min,m_max,Inorm_max,n,Vnorm_tuneFactor,figNum,prefix,postType )
% Tuning Parameters
% minVt = minimum threshold voltage for MOSFET,
%         useful for finding subthreshold region for finding Vnorm
% m_min = long-channel limit
% m_max = degenerated BJT limit
% Inorm_max = arbitrarily large limit
% n = number of points between min and max (resolution)
% Vnorm_tuneFactor = search window around nominal Vnorm
%
% Outputs
% Inorm_opt - a four element row array that contains the optimal
%             normalization currents for different fitting algorithms. The
%             first element should be reducing

%% Initial Conditions

if (strcmp(postType,'Median'))
    [I,V]=lowIMedianFilter(I,V,lowILim);
elseif (strcmp(postType,'Mean'))
    [I,V]=lowIMeanFilter(I,V,lowILim);
else
```

```
    [I,V]=lowIMeanFilter(I,V,lowILim);
end

[gm,d_I,d_V] = derivative_3pt(I,V);
index = find(d_V < minVt);
[d_gm, ~, ~] = derivative_3pt(gm(index),d_I(index));
d_gm=d_gm(find(d_gm(:) <= 1/0.026 & d_gm(:)>=1/(0.026*3)));
Vnorm_min = (1-Vnorm_tuneFactor)*1/(median(d_gm(:)));
Vnorm_max = (1+Vnorm_tuneFactor)*1/(median(d_gm(:)));
Inorm_min = d_I(max(index)+1);

gm(1:8)=[];
d_I(1:8 )=[];
d_V(1:8)=[];


%% Non-Linear Regression Fit
m = linspace(m_min,m_max,n);
Inorm = linspace(Inorm_min,Inorm_max,n);
Vnorm = linspace(Vnorm_min,Vnorm_max,n);

pMat = combvec(Vnorm,Inorm,m); % parameter matrix
VComb = pMat(1,:);
IComb = pMat(2,:);
mComb = pMat(3,:);

numSol = size(pMat,2);
e_Max_gm = zeros(1,numSol);
e_Avg_gm = zeros(1,numSol);
e_Max_V = zeros(1,numSol);
e_Avg_V = zeros(1,numSol);
e_Max_Avg = zeros(1,numSol);
```

```
e_Avg_Avg = zeros(1,numSol);


L=length(d_I);
d2_V_Lo = d_V(1:L-1);
d2_V_Hi = d_V(2:L);
d2_I_Lo = d_I(1:L-1);
d2_I_Hi = d_I(2:L);
tic
parfor i=1:size(pMat,2)
    gmexp = 2./(1+(1+d_I./IComb(i)).^mComb(i)).*d_I./VComb(i);
    Vexp = 2.*(d2_I_Hi - d2_I_Lo)./(gmexp(2:L)+gmexp(1:L-1)) + d2_V_Lo;
    % gm model fitting
    e_gm = abs(gmexp-gm)./gm;
    e_Max_gm(i) = max(e_gm); % error should not exceed certain limit
    e_Avg_gm(i) = mean(e_gm);

    % I-V model fitting
    e_V = abs(Vexp-d2_V_Hi)./d2_V_Hi;
    e_Max_V(i) = max(e_V); % error should not exceed certain limit
    e_Avg_V(i) = mean(e_V);

    e_Max_Avg(i) = mean([e_Max_gm(i) e_Max_V(i)],2);
    e_Avg_Avg(i) = mean([e_Avg_gm(i) e_Avg_V(i)],2);

end

[~,index_gmMax] = min(e_Max_gm);
[~,index_gmAvg] = min(e_Avg_gm);
[~,index_VMax] = min(e_Max_V);
[~,index_VAvg] = min(e_Avg_V);
[~,index_AvgMax] = min(e_Max_Avg);
```

```
[~,index_AvgAvg] = min(e_Avg_Avg);
m_AvgMaxOpt = mComb(index_AvgMax);
I_AvgMaxOpt = IComb(index_AvgMax);
V_AvgMaxOpt = VComb(index_AvgMax);
m_AvgAvgOpt = mComb(index_AvgAvg);
I_AvgAvgOpt = IComb(index_AvgAvg);
V_AvgAvgOpt = VComb(index_AvgAvg);
m_gmMaxOpt = mComb(index_gmMax);
I_gmMaxOpt = IComb(index_gmMax);
V_gmMaxOpt = VComb(index_gmMax);
m_gmAvgOpt = mComb(index_gmAvg);
I_gmAvgOpt = IComb(index_gmAvg);
V_gmAvgOpt = VComb(index_gmAvg);
m_VMaxOpt = mComb(index_VMax);
I_VMaxOpt = IComb(index_VMax);
V_VMaxOpt = VComb(index_VMax);
m_VAvgOpt = mComb(index_VAvg);
I_VAvgOpt = IComb(index_VAvg);
V_VAvgOpt = VComb(index_VAvg);
toc

Inorm_opt = [I_gmMaxOpt I_gmAvgOpt I_VMaxOpt I_VAvgOpt I_AvgMaxOpt
    I_AvgAvgOpt];
Vnorm_opt = [V_gmMaxOpt V_gmAvgOpt V_VMaxOpt V_VAvgOpt V_AvgMaxOpt
    V_AvgAvgOpt];
m_opt = [m_gmMaxOpt m_gmAvgOpt m_VMaxOpt m_VAvgOpt m_AvgMaxOpt
    m_AvgAvgOpt];
e_max_gm = zeros(1,6);
e_avg_gm = zeros(1,6);
e_max_v = zeros(1,6);
e_avg_v = zeros(1,6);
%% Visual Analysis Plot
```

```
h = [];
gm1exp = 2./(1+(1+d_I./I_gmMaxOpt).^m_gmMaxOpt).*d_I./V_gmMaxOpt;
Vexp1 = 2.*(d_I(2:L) - d_I(1:L-1))./(gm1exp(2:L)+gm1exp(1:L-1)) + d_V(1:
    L-1);
gm2exp = 2./(1+(1+d_I./I_gmAvgOpt).^m_gmAvgOpt).*d_I./V_gmAvgOpt;
Vexp2 = 2.*(d_I(2:L) - d_I(1:L-1))./(gm2exp(2:L)+gm2exp(1:L-1)) + d_V(1:
    L-1);
e_max_gm(1) = max(abs(gm1exp-gm)./gm*100);
e_avg_gm(1) = mean(abs(gm1exp-gm)./gm*100);
e_max_gm(2) = max(abs(gm2exp-gm)./gm*100);
e_avg_gm(2) = mean(abs(gm2exp-gm)./gm*100);
e_max_v(1) = max(abs(Vexp1-d_V(2:L))./d_V(2:L)*100);
e_avg_v(1) = mean(abs(Vexp1-d_V(2:L))./d_V(2:L)*100);
e_max_v(2) = max(abs(Vexp2-d_V(2:L))./d_V(2:L)*100);
e_avg_v(2) = mean(abs(Vexp2-d_V(2:L))./d_V(2:L)*100);

gm1expQuad = 2./(1+(1+d_I./I_gmMaxOpt).^0.5).*d_I./V_gmMaxOpt;
Vexp1Quad = 2.*(d_I(2:L) - d_I(1:L-1))./(gm1expQuad(2:L)+gm1expQuad(1:L
    -1)) + d_V(1:L-1);
gm2expQuad = 2./(1+(1+d_I./I_gmAvgOpt).^0.5).*d_I./V_gmAvgOpt;
Vexp2Quad = 2.*(d_I(2:L) - d_I(1:L-1))./(gm2expQuad(2:L)+gm2expQuad(1:L
    -1)) + d_V(1:L-1);

h(1)=figure(figNum); set(h(1),'Visible','on'); clf;
subplot(2,1,1);
scatter(d_I,gm,'k');
%title('g_{m} Model Fit');
hold on;
plot(d_I,gm1exp,'b');
plot(d_I,gm2exp,'r');
plot(d_I,gm1expQuad,'g');
plot(d_I,gm2expQuad,'m');
```

```
145 grid on; grid minor;
146
147 l=legend('Experimental','Reduce e_{max}','Reduce e_{avg}','Reduce e_{max
    }, m=0.5','Reduce e_{avg}, m=0.5','Location','northwest');
148 set(l,'FontSize',11);
149 set(gca,'FontSize',12);
150 xlabel('Drain Current','FontSize',13);
151 ylabel('Transconductance (A/V^2)','FontSize',13);
152 xlim([min(d_I) max(d_I)]);
153 subplot(2,1,2);
154 plot(d_I,abs(gm1exp-gm)./gm*100,'b');
155 hold on; grid on; grid minor;
156 plot(d_I,abs(gm2exp-gm)./gm*100,'r');
157 plot(d_I,abs(gm1expQuad-gm)./gm*100,'g');
158 plot(d_I,abs(gm2expQuad-gm)./gm*100,'m');
159 l=legend('Reduce e_{max}','Reduce e_{avg}','Reduce e_{max}, m=0.5','
    Reduce e_{avg}, m=0.5','Location','northwest');
160 set(l,'FontSize',11);
161 set(gca,'FontSize',12);
162 xlabel('Drain Current','FontSize',13);
163 ylabel('Error (%)','FontSize',13);
164 xlim([min(d_I) max(d_I)]);
165
166 h(2)=figure(figNum+1);set(h(2),'Visible','on');clf;
167 subplot(2,1,1);
168 scatter(d_I(2:L),d_V(2:L),'k');
169 hold on; grid on; grid minor;
170 plot(d_I(2:L),Vexp1,'b');
171 plot(d_I(2:L),Vexp2,'r');
172 title('gm Model Fit');
173 legend('Experimental','Max Model','Mean Model');
174 xlabel('Drain Current (A)');
```

```
ylabel('Gate Voltage (V)');
subplot(2,1,2);
plot(d_I(2:L),abs(Vexp1-d_V(2:L))./d_V(2:L)*100,'b');
hold on; grid on; grid minor;
plot(d_I(2:L),abs(Vexp2-d_V(2:L))./d_V(2:L)*100,'r');
legend('Max Model','Mean Model');
xlabel('Drain Current');
ylabel('Error (%)');
xlim([min(d_I) max(d_I)]);

gm3exp = 2./(1+(1+d_I./I_VMaxOpt).^m_VMaxOpt).*d_I./V_VMaxOpt;
Vexp3 = 2.*(d_I(2:L) - d_I(1:L-1))./(gm3exp(2:L)+gm3exp(1:L-1)) + d_V(1:
    L-1);
gm4exp = 2./(1+(1+d_I./I_VAvgOpt).^m_VAvgOpt).*d_I./V_VAvgOpt;
Vexp4 = 2.*(d_I(2:L) - d_I(1:L-1))./(gm4exp(2:L)+gm4exp(1:L-1)) + d_V(1:
    L-1);
e_max_gm(3) = max(abs(gm3exp-gm)./gm*100);
e_avg_gm(3) = mean(abs(gm3exp-gm)./gm*100);
e_max_gm(4) = max(abs(gm4exp-gm)./gm*100);
e_avg_gm(4) = mean(abs(gm4exp-gm)./gm*100);
e_max_v(3) = max(abs(Vexp3-d_V(2:L))./d_V(2:L)*100);
e_avg_v(3) = mean(abs(Vexp3-d_V(2:L))./d_V(2:L)*100);
e_max_v(4) = max(abs(Vexp4-d_V(2:L))./d_V(2:L)*100);
e_avg_v(4) = mean(abs(Vexp4-d_V(2:L))./d_V(2:L)*100);

h(3)=figure(figNum+2);set(h(3),'Visible','on');clf;
subplot(2,1,1);
scatter(d_I,gm,'k');
title('Voltage Model Fit');
hold on;
plot(d_I,gm3exp,'b');
plot(d_I,gm4exp,'r');
```

```
grid on; grid minor;
legend('Experimental','Max Model','Mean Model');
xlabel('Drain Current');
ylabel('Transconductance (A/V^2)');
xlim([min(d_I) max(d_I)]);
subplot(2,1,2);
plot(d_I,abs(gm3exp-gm)./gm*100,'b');
hold on; grid on; grid minor;
plot(d_I,abs(gm4exp-gm)./gm*100,'r');
legend('Max Model','Mean Model');
xlabel('Drain Current');
ylabel('Error (%)');
xlim([min(d_I) max(d_I)]);

h(4)=figure(figNum+3);set(h(4),'Visible','on');clf;
subplot(2,1,1);
scatter(d_I(2:L),d_V(2:L),'k');
hold on; grid on; grid minor;
plot(d_I(2:L),Vexp3,'b');
plot(d_I(2:L),Vexp4,'r');
title('Voltage Model Fit');
legend('Experimental','Max Model','Mean Model');
xlabel('Drain Current (A)');
ylabel('Gate Voltage (V)');
subplot(2,1,2);
plot(d_I(2:L),abs(Vexp3-d_V(2:L))./d_V(2:L)*100,'b');
hold on; grid on; grid minor;
plot(d_I(2:L),abs(Vexp4-d_V(2:L))./d_V(2:L)*100,'r');
legend('Max Model','Mean Model');
xlabel('Drain Current');
ylabel('Error (%)');
xlim([min(d_I) max(d_I)]);
```

```
237
238 gm5exp = 2./(1+(1+d_I./I_AvgMaxOpt).^m_AvgMaxOpt).*d_I./V_AvgMaxOpt;
239 Vexp5 = 2.*(d_I(2:L) - d_I(1:L-1))./(gm5exp(2:L)+gm5exp(1:L-1)) + d_V(1:
    L-1);
240 gm6exp = 2./(1+(1+d_I./I_AvgAvgOpt).^m_AvgAvgOpt).*d_I./V_AvgAvgOpt;
241 Vexp6 = 2.*(d_I(2:L) - d_I(1:L-1))./(gm6exp(2:L)+gm6exp(1:L-1)) + d_V(1:
    L-1);
242 e_max_gm(5) = max(abs(gm5exp-gm)./gm*100);
243 e_avg_gm(5) = mean(abs(gm5exp-gm)./gm*100);
244 e_max_gm(6) = max(abs(gm6exp-gm)./gm*100);
245 e_avg_gm(6) = mean(abs(gm6exp-gm)./gm*100);
246 e_max_v(5) = max(abs(Vexp5-d_V(2:L))./d_V(2:L)*100);
247 e_avg_v(5) = mean(abs(Vexp5-d_V(2:L))./d_V(2:L)*100);
248 e_max_v(6) = max(abs(Vexp6-d_V(2:L))./d_V(2:L)*100);
249 e_avg_v(6) = mean(abs(Vexp6-d_V(2:L))./d_V(2:L)*100);
250
251 h(5)=figure(figNum+4); set(h(5),'Visible','on');clf;
252 subplot(2,1,1);
253 scatter(d_I,gm,'k');
254 title('Avg Model Fit');
255 hold on;
256 plot(d_I,gm5exp,'b');
257 plot(d_I,gm6exp,'r');
258 grid on; grid minor;
259 legend('Experimental','Max Model','Mean Model');
260 xlabel('Drain Current');
261 ylabel('Transconductance (A/V^2)');
262 xlim([min(d_I) max(d_I)]);
263 subplot(2,1,2);
264 plot(d_I,abs(gm5exp-gm)./gm*100,'b');
265 hold on; grid on; grid minor;
266 plot(d_I,abs(gm6exp-gm)./gm*100,'r');
```

```
legend('Max Model','Mean Model');
xlabel('Drain Current');
ylabel('Error (%)');
xlim([min(d_I) max(d_I)]);

h(6)=figure(figNum+5);set(h(6),'Visible','on');clf;
subplot(2,1,1);
scatter(d_I(2:L),d_V(2:L),'k');
hold on; grid on; grid minor;
plot(d_I(2:L),Vexp5,'b');
plot(d_I(2:L),Vexp6,'r');
title('Avg Model Fit');
legend('Experimental','Max Model','Mean Model');
xlabel('Drain Current (A)');
ylabel('Gate Voltage (V)');
subplot(2,1,2);
plot(d_I(2:L),abs(Vexp5-d_V(2:L))./d_V(2:L)*100,'b');
hold on; grid on; grid minor;
plot(d_I(2:L),abs(Vexp6-d_V(2:L))./d_V(2:L)*100,'r');
legend('Max Model','Mean Model');
xlabel('Drain Current');
ylabel('Error (%)');
xlim([min(d_I) max(d_I)]);
savefig(h,strcat(prefix,'_',postType),'compact');

end
```

Listing B.1: Model Fitting Algorithm

B.2 Three Point Derivative

```
function [ dY_dX, Y_in2, X_in2 ] = derivative_3pt( Y_in,X_in )
%  Input arguments are:
```

```
3 %   Y_in - y-axis variable / drain current array
4 %   X_in - x-axis variable / gate-source voltage array
5 %  Output value:
6 %   dY_dX - derivative / transconductance output array
7 %   Y_out - corresponding y-axis points
8 %   X_out - corresponding x-axis points
9 %
10 % Description:
11 % This function calculates a derivative from three data points using the
12 % formula dY/dX = (Y2-Y1)/(X2-X1) + (Y3-Y2)/(X3-X2) - (Y3-Y1)/(X3-X1)
13 %
14 % Example:
15 % This function calculates the transconductance from the drain current
    and
16 % gate-source voltage data.  It uses three points to generate a
    derivative.
17 % The derivative exists at the center voltage point of any three
    voltages.
18 % The formula for the derivative is:
19 %   gm2 = (I2-I1)/(V2-V1) + (I3-I2)/(V3-V2) - (I3-I1)/(V3-V1)
20 % This function vectorizes all the difference and division calculations
21 % using MATLAB's element operations.  The output array will always be
    two
22 % points smaller than the input arrays due to the edge points having
23 % insufficient data for the derivative.
24
25 if (length(Y_in) ~= length(X_in))
26     fprintf(1,'X and Y Arrays not the same size\n');
27 end
28
29 Y_in1 = Y_in(1:length(Y_in)-2);
30 Y_in2 = Y_in(2:length(Y_in)-1);
```

```
Y_in3 = Y_in(3:length(Y_in));
X_in1 = X_in(1:length(X_in)-2);
X_in2 = X_in(2:length(X_in)-1);
X_in3 = X_in(3:length(X_in));

dY_dX = (Y_in2 - Y_in1)./(X_in2 - X_in1) + ...
    (Y_in3 - Y_in2)./(X_in3 - X_in2) - ...
    (Y_in3 - Y_in1)./(X_in3 - X_in1);

end
```

Listing B.2: Quadratic Spline Differentiation Algorithm

B.3 Low Current Filtering

```
function [ Iout,Vout ] = lowIMeanFilter( I,V,limit )
% I - input current array
% V - input voltage array
% limit - maximum current used for low current processing
%
% Iout - output current array
% Vout - output voltage array
%
% Function takes all currents below the limit and performs an average of
% all samples within +/- 5% of each other.

idxLowI=find(I<limit,1,'last');
idxHiI = idxLowI+1;

flag = I(1);
Vbuffer = [];
Ibuffer = [];
Iout=[];
```

```
Vout=[];
for i=1:idxLowI
    if (I(i) > 0.95*flag) && (I(i) < 1.05*flag)
       Vbuffer = [Vbuffer; V(i)];
       Ibuffer = [Ibuffer; I(i)];
    else
        Vout=[Vout;mean(Vbuffer)];
        idxI = find(I>=mean(Ibuffer),1,'first');
        Iout=[Iout;I(idxI)];
        Vbuffer = [];
        Ibuffer = [];
        flag = I(i+1);
    end
end

Iout = [Iout;I(idxHiI:length(I))];
Vout = [Vout;V(idxHiI:length(I))];
end
```

Listing B.3: Low Current Mean Filter Algorithm

```
function [ Iout,Vout ] = lowIMedianFilter( I,V,limit )
% I - input current array
% V - input voltage array
% limit - maximum current used for low current processing
%
% Iout - output current array
% Vout - output voltage array
%
% Function takes all currents below the limit and takes the median of
% all samples within +/- 5% of each other.

idxLowI=find(I<limit,1,'last');
idxHiI = idxLowI+1;
```

```
14
15 flag = I(1);
16 buffer = [];
17 Iout=[];
18 Vout=[];
19 for i=1:idxLowI
20     if (I(i) > 0.95*flag) && (I(i) < 1.05*flag)
21        buffer = [buffer; V(i)];
22     else
23         Vout=[Vout;median(buffer)];
24         idxI = find(V>=median(buffer),1,'first');
25         Iout=[Iout;I(idxI)];
26         buffer = [];
27         flag = I(i+1);
28     end
29 end
30
31 Iout = [Iout;I(idxHiI:length(I))];
32 Vout = [Vout;V(idxHiI:length(I))];
33 end
```

Listing B.4: Low Current Median Filter Algorithm

Appendix C

LABVIEW DATA ACQUISITION PROGRAM

The Keithley 2400 SourceMeter functions as a current source in the test circuit. A LabVIEW program easily controls the Keithley 2400 SourceMeter. Keithley wrote drivers to communicate with the instrument, including an example called "Keithley 24XX Output List and Acquire.vi". This file displayed the measurements on a front panel graph but did not provide an easy way to glean the data in a csv or tab delimited form. Modifying the example program to include reading desired current points from a configuration file and writing the results to a tab-delimited output file solved this problem. The modification yielded the new program titled "Keithley 24XX Output List and Acquire with File IO.vi". We used the program on a Keithley 2400 LV, as well as a regular Keithley 2400 with good results. Fig. C.1 and C.2 show the program with file input and output. We sourced the drivers from `http://sine.ni.com/apps/utf8/niid_web_display.download_page?p_id_guid=25B255F3AA83660EE0440003BA7CCD71`.

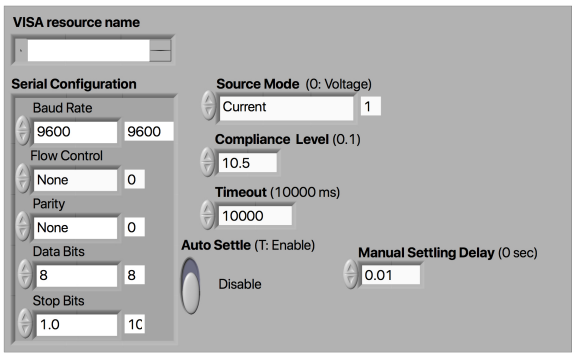

Figure C.1: Keithley 2400 Data Acquisition Front Panel

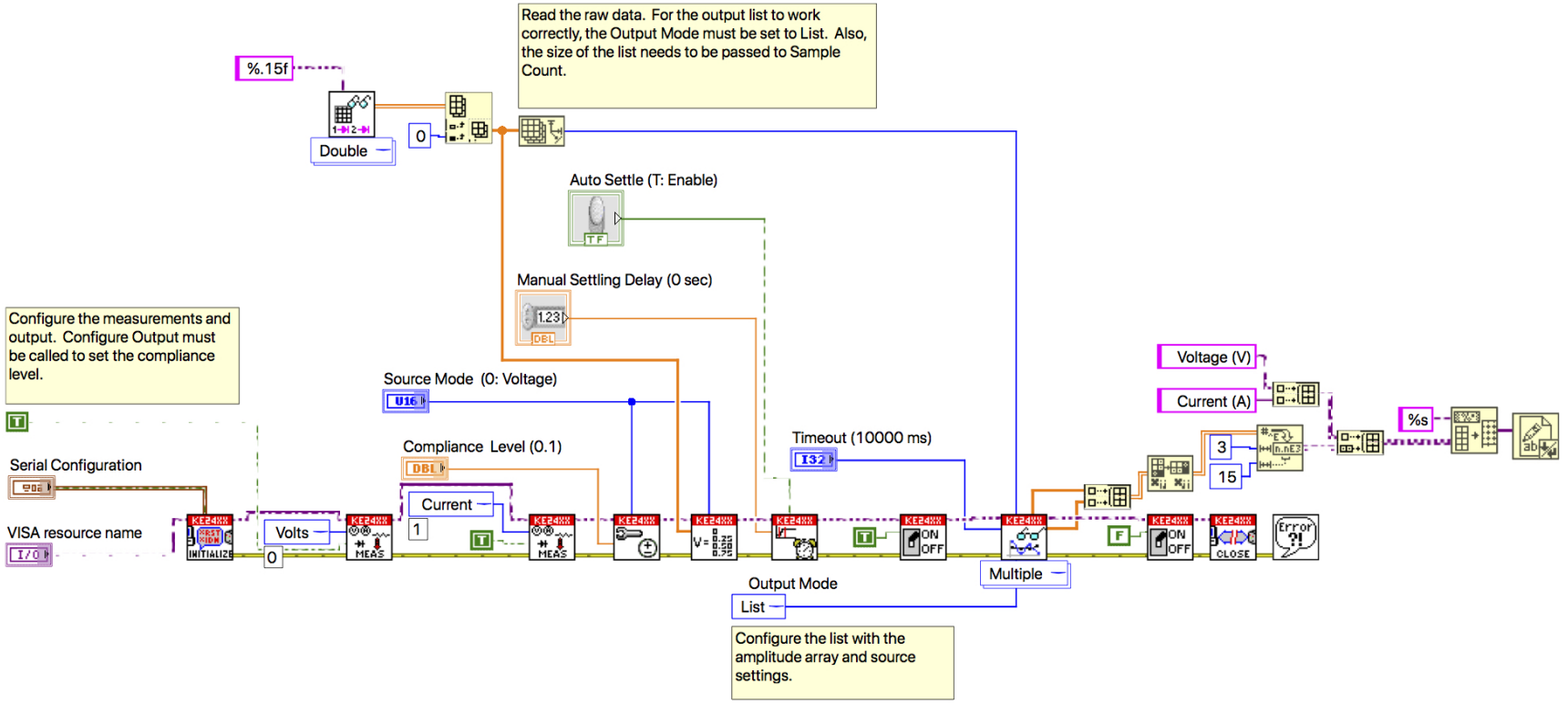

Figure C.2: Keithley 2400 Data Acquisition Block Diagram