The Metal-Oxide-Semiconductor Field-Effect Transistor (MOSFET)

Introduction

Recall that in Lab 3 we studied the current versus voltage properties of a forward biased diode. The diode consisted of a PN semiconductor junction (in practice we used an NPN BJT transistor but focused our attention on the Base-Emitter PN junction). By capturing the current versus voltage characteristics of the junction we were able to determine Boltzmann’s constant, and we were also able to use the temperature dependency of the diode’s reverse bias current, $I_0$, to determine the band gap of the silicon diode.

The development of the PN junction set the stage for the fabrication of many subsequent “active” semiconductor devices. One of the most important such devices is the metal-oxide-semiconductor field effect transistor (MOSFET). Whereas transistors in general are important because of their versatility, behaving as switches, amplifiers, or oscillators depending on their configuration, MOSFETs in particular exhibit extremely beneficial low-power switching properties when compared against alternative transistor types. As such they have become absolutely fundamental in computing and memory applications, where their physical dimensions have been scaled down to allow the fabrication of many millions of MOSFET transistors on individual semiconductor chips or integrated circuits.

In this lab we will measure several characteristics of a discrete MOSFET transistor and compare the properties to the expected parameters provided by the manufacturer.

MOSFET Structure

Figure 1 illustrates the profile of an n-channel MOSFET. Two n-doped regions, the drain and the source, are embedded in a p-type semiconductor substrate.

On the surface of the semiconductor is a layer of insulator (SiO$_2$ in the case of Si substrates). Openings etched or masked into the insulator allow the deposition of metallic electrodes contacting n-doped drain and source regions. Note that beneath the gate electrode the insulator layer remains intact, isolating the gate and substrate. Voltages applied to this “gate” electrode control the flow of current between the source and drain by introducing or depleting charge carrier states in the substrate region beneath the
Theory of Operation

The MOS Capacitor

The MOSFET's behavior and operation is based in large part on the physics of the MOS capacitor. We’ll initially ignore the presence of the source and drain, leaving only the capacitor-like gate, insulator, and semiconductor structure. We’ll briefly consider how a voltage applied to the gate modifies the energy bands of the semiconductor substrate in the vicinity of the area beneath the gate, and how this in turn influences the carrier densities and depletion regions therein.

When $V_G$ is 0V, the metal and semiconductor Fermi-levels align. The semiconductor's Fermi-level is $q\phi_F$ electron volts below its intrinsic level, $E_i$, and is indicative of how strongly p-type the substrate is, as we recall that the concentration of majority carriers in doped p-type semiconductor is given by

$$p = N_a \exp(-\frac{E_F - E_V}{kT})$$

When $V_G < 0V$ is applied at the gate, the Fermi-level in the metal increases by $qV_G$. This has the effect of depositing negative charges at the gate, which in turn attracts additional holes to the oxide-semiconductor interface. As such, the semiconductor bands are bent near the interface and the Fermi-level, $E_F$, and valence band, $E_v$, are closer to each other in energy due to the increased majority carrier density. The device is said to be in an “accumulation” state. Its capacitance is given by

$$C_{ox} = \frac{\varepsilon_{ox}}{d_{ox}}$$

where $\varepsilon_{ox}$ is the permittivity of the insulator layer and $d_{ox}$ is the insulator thickness.

When $V_G > 0V$, things become more interesting. The redistribution of carrier states cause the bands of the semiconductor to bend near the interface such that the Fermi-level and $E_i$ become farther apart. That is, $V_G > 0V$ causes positive charge to build at the gate, and in turn induces a reduction or “depletion” state in the p-type semiconductor in
the area near the oxide-semiconductor interface. The effect is analogous to the depletion region between a pn+ junction\(^1\), the width of which is [2],

\[
W = \left[ \frac{2\varepsilon_s (\phi_s)}{qN_d} \right]^\frac{1}{2}.
\]

Here, \(\phi_s\) is the potential difference across the depletion region (i.e. energy band bending, as shown in Figure 6), where \(V_G = V_i + \phi_s\) and \(V_i\) represents the potential difference across the oxide insulator layer.

As a result, the capacitance becomes like a series combination of the oxide capacitance and that due to the depletion region width,

\[
C_d = \frac{\varepsilon_s}{W},
\]

where \(\varepsilon_s\) is the permittivity of the depletion layer.

As \(V_G\) becomes even larger, eventually \(E_F\) becomes greater than \(E_i\). In this situation the region of semiconductor near the oxide interface becomes “inverted,” meaning that conduction band carrier states become filled with minority electrons forming an n-channel. (Realize that when we add drain and source electrodes to this MOS capacitor, we can induce current conduction across the newly formed n-channel just by applying a \(V_{DS}\!).

At this point the depth of the semiconductor depletion layer is at a maximum, and the total capacitance of the device is at a minimum,

\[
C = \left[ \frac{1}{C_{ox}} + \frac{1}{C_d} \right]^{-1} = \frac{C_{ox}C_d}{(C_{ox}+C_d)}.
\]

Figure 5: \(V_G \gg 0\) V causes a state of inversion, where a channel of n-type carrier forms at the oxide semiconductor boundary.

The gate voltage required to induce this inversion state is defined as the device’s “threshold voltage,” \(V_T\). \(V_T\) represents the point at which the MOSFET becomes conductive, and it can be controlled in device fabrication by tailoring the material parameters and physical dimensions of the MOSFET. As \(V_G\) becomes greater than \(V_T\) the inverted MOS capacitor enters a state of “strong inversion,” where \(\phi_s = 2\phi_F\).

\(^1\) Recall that for a biased pn junction the depletion width is given by

\[
W = \left[ \frac{2\varepsilon_s}{q} \left( \frac{V_a - V}{N_a+N_d} \right) \right]^\frac{1}{2},
\]

where \(V_a - V\) is the change in the potential barrier due to bias voltage \(V\), and \(V_a\) is the potential barrier at equilibrium.
Measurements of $V_T$, $C_{ox}$, and $C_d$ can thus be used to characterize many of the physical parameters of the device, like oxide thickness and doping densities. (That is, C-V measurements can serve as a useful characterization and reliability testing tool.)

*The MOSFET Transistor*

So having considered the states of the MOS capacitor, what of the MOSFET transistor physics?

With the addition of n-doped source and drain regions (or p-doped regions for a p-channel MOSFET), recognize that we’ve introduced two pn junctions into the device structure. If we consider the MOS capacitor’s accumulation and depletion states as we vary $V_g$, the substrate remains p-type and all that is altered is the concentration of majority p-type carriers in the area beneath the gate. As such, voltage applied from drain to source, $V_{DS}$, is equivalent to reverse biasing the drain-substrate np junction. The potential barrier is increased, the width of the depletion boundary between drain and substrate is widened, and no drain current flows.

However, as $V_g$ becomes greater than $V_T$ we’ve seen that inversion occurs, and suddenly the drain and source become conductively connected via the induced n-channel. Current can flow freely if a voltage $V_{DS}$ is applied. The MOSFET is switched into its conductive mode.

We would still like to consider how the device behaves in this inversion state as we vary $V_{DS}$. While $V_{DS}$ is small, the current from drain to source is observed to be roughly proportional to $V_{DS}$ and the MOSFET acts somewhat like an ohmic resistive load. In this scenario the device is particularly useful for switching applications. (Millions of MOSFET switches are used in computer memory and processors.)

As $V_{DS}$ is increased for a given $V_g>V_T$ eventually all carriers generated in the channel layer are quickly swept from drain to source. As part of your report you’ll comment on how this is reflected in the measurements you’ll make of $I_D$ versus $V_{DS}$. 

![Figure 6: The onset of "strong inversion," where $\phi_s = 2\phi_F$ in the neutral $p$ substrate.](image)

![Figure 7: The MOSFET in profile. $ VG > 0 $ causes "strong inversion" and an n-channel forms, linking a conductive path from drain to source. The MOSFET is in "on" state.](image)
**Experiment**

In this experiment you will characterize the behaviour of a MOSFET transistor by measuring drain current, \( I_D \), under various \( V_G \) and \( V_{DS} \) conditions, and you will compare the results observed against values provided from manufacturer data sheets.

Recall that in Lab 3 we measured the I-V curve for a PN junction by manually varying a current supply and recording the resultant current and voltage across the diode. In this laboratory we will need to collect similar curves. However we will automate the process somewhat by building an I-V curve tracing tool using an Arduino microcontroller and certain voltage supply and current sensing “breakout boards.”

Initially you will wire your equipment and write a short program to acquire a graph of \( I_D \) as a function of \( V_{DS} \) for various gate voltages, \( V_G \).

As well, you’ll revise your program to acquire a graph of \( I_D \) vs \( V_G \) for a fixed large value of \( V_{DS} \). From this graph you’ll determine the transconductance gain of your transistor (often written \( g_{fs} \)) and compare against the datasheet specification.

Finally based on your characterization of your MOSFET transistor, you will design a simple transistor switch to provide power to a small load device like a light bulb or motor.

**Components**

MOSFET transistor (2N7000, ZVN2106, or similar)
Arduino (USB Boarduino, Adafruit Metro Mini, or suitable alternative version)
Adafruit MCP4745 12-bit 5V DAC breakout board (2 required)
Adafruit INA219 DC High-Side Current Sensor breakout board
MCP6002 dual op-amp (CMOS)
TIP41C BJT transistor and external power supply
Electrical prototyping board
Jumper wires
[If amplifying the 5V DAC signal, extra components: *External power supply
10kOhm resistors x3]

---

2 Note, the objective of the lab is to determine properties of the MOSFET transistor. The Arduino I-V curve tracer you will build and program serves as a tool to help achieve that end goal. However the tool should not be a main focus of your attention when composing your report.
I-V Curve Tracer Circuit Layout

Figure 8: Layout of electrical components and wiring for I-V curve tracing application. (Version shown uses Metro Mini Arduino. See appendix for alternative version and configurations.)

Arduino-based I-V Curve Tracer - Configuration and Principles of Operation

The goal of the first part of the experiment is to capture I-V curves where we monitor drain current, $I_D$, while varying drain-source bias voltage, $V_{DS}$, for various gate voltages, $V_{GS}$.

The goal of the second part of the experiment is to measure $I_D$ while varying $V_{GS}$ as $V_{DS}$ is held at a fixed voltage. The configuration of the circuit is the same for both parts of the experiment; only code changes are required.

Note that transistors are often static sensitive, so it is best to avoid touching their pins. Furthermore, it is important to know which pin of a MOSFET is which. For the MOSFETs used in this lab, the pins are as in the Figure, though it is always good to check the specification sheet to be certain.

To configure the electrical components, one DAC board will be used to supply our $V_{DS}$ voltage (dac_vds). Note that we route our dac_vds Vout pin through an MCP6002 op-amp stage, followed by a TIP41
npn BJT transistor. The op-amp stage serves to “isolate” any DAC output resistance from influencing voltages levels at later stages of the circuit (i.e. the DAC could act as a parallel resistor and create a “voltage divider” effect). However, despite that the op-amp stage isolates our dac.v.ds voltage, the MCP6002 chip cannot “source” enough current to properly drive our MOSFET. To compensate, a TIP41 transistor stage is added to increase the current available to the MOSFET. This is at the expense of a small drop in our voltage due to the base-emitter PN junction.

After the BJT stage, a wire connects to the V+ input on the INA219 Current Sensor board, and a wire from the V- output of the INA219 connects to the MOSFET’s drain. The MOSFET’s source pin is connected to ground.

The second DAC board is used to supply our Vgs voltage (dac.v.g). There is no need to “isolate” this DAC because the MOSFET gate provides extremely high (effectively infinite) insulation, or resistance, so no voltage dividing effect is expected. The V.out pin from dac_vgs is therefore wired directly to the gate pin of our MOSFET.

**Arduino Code**

We will write two short programs to capture the MOSFET IV curves.

**Program 1**

The first program should use one DAC to set values of gate voltage (~0V, 0.5V, 1.0V... 5V) and use a second DAC to scan the drain-source voltage in small increments (steps of a few mV).

**Program 2**

Modify your program so that you apply the largest drain-source voltage available (maximum DAC V_out). Now step the gate voltage from 0-5V in small steps (a few mV).

**Presentation of Results**

You will capture a set of Id vs V_ds data from your program by opening an Arduino serial terminal. Copy and paste the data for a given Vgs condition into a spreadsheet. You should be able to plot several Id vs V_ds curves on the same plot to illustrate the effect of changing Vgs. (You’ll use the plot to help design a basic MOSFET transistor switch.)

Once you’ve modified your program to capture Id vs Vgs for a large value of VDs you’ll similarly capture your results by pasting the data from a serial terminal into a spreadsheet. You’ll use this graph to determine both Vt and g_s for your transistor, and you’ll determine whether it falls within manufacturer specification.
Designing a Simple MOSFET Transistor Switch

Based on your $I_D$ vs $V_{DS}$ plot and your value of $V_T$, conceptually design a simple transistor switch as shown in Figure 10. The intent of the switch circuit is to turn on a resistive load like a light bulb or small motor. Assume the power demand from the load is equal to the last digit of your student number (if working in pairs, select one partner’s student number), i.e. for student number 123426, use a power load of 6W. Assume a supply voltage, $V_{dd}$, of 24V.

Recall that $P=IV$. Select a suitable current (and therefore $V_G$) from your $I_D$ vs $V_{DS}$ plot to supply the appropriate power to the circuit load.

How much power will be dissipated across the MOSFET drain/source? Will the MOSFET survive? Explain.

Analysis and Discussion

Comment on whether your $I_D$ vs $V_{DS}$ curves and $I_D$ vs $V_{GS}$ curve are as expected. Explain the shape of the curves of $I_D$ vs $V_{DS}$ as $V_{DS}$ increases.

Where appropriate, compare your results to manufacturer specification: determine the slope of the plot of $I_D$ vs $V_G$ for $V_G > V_T$ (linear region). Compare against the tolerances provided in the datasheet. (Note that transistor parameters can vary dramatically, and therefore values of $V_T$ and $g_m$ may been significantly different than typical values and still fall within specification.)

Note that in our code we have intentionally “pulsed” our MOSFET on for a few milliseconds at a time to take measurements, and then powered $V_{DS}$ or $V_G$ off for some length of time. Why is this recommended for our measurements? Refer to the MOSFET specification sheet in your answer.

References

Appendix 1: Introduction to Arduino

Arduino is a computing platform that consists of a hardware device (a microcontroller known as the board) and a software package to operate it. Arduino users can write programs that read information from sensors and can control output devices like motors or lights. The platform is relatively easy to use even for individuals with minimal programming experience.

The Arduino programming language is based on C/C++ and is used to write code and communicate with the board. An Arduino program, also known as a sketch, is written in an open source software package known as the integrated development environment (IDE). The Arduino IDE is available online and can be downloaded from the following link: https://www.arduino.cc/en/Main/Software.

The IDE allows users to Verify their program before it is uploaded to the board to ensure that they are free of any syntax errors. Figure A1 above shows the interface of IDE and a simple sketch used to blink a light emitting diode (LED).

When your sketch is complete and compiles successfully, it can be uploaded to the board using the Upload button shown in Figure A1.
Installing Arduino Libraries

In some cases when additional circuit components are used, special library files might be required. In this experiment, you will be using an INA219 DC current sensor and an MCP4725 digital-to-analog converter (DAC). Each of these “breakout boards” give the Arduino extra capabilities but require installation of an Arduino driver library file.

To install or update a library, go to Sketch > Include Library > Manage Libraries.

Use the search feature to locate and install/update Adafruit MCP4725. Do the same for the Adafruit INA219 library.

Uploading Your Code and Running Your Program

Now that you have installed the Arduino IDE and the required library files, restart the IDE and plug your Arduino board using the USB cable provided.

To upload your program you'll need to tell the IDE which type of Arduino you're using (typically and Arduino Uno or Duemilanova). Go to the Tools menu and select your board model.

You’ll also need to tell the IDE which serial port your Arduino board is connected to. Again this can be selected under Tools > Serial Port menu. For Macintosh users, select the port that begins with /dev/cu.usbserial-. Windows users might select a COM port listed.

Finally, upload the sketch to the board. Your Arduino will continuously run your sketch until it is unplugged or a different sketch is uploaded. If you unplug your Arduino your sketch will remain installed and will automatically restart next time you power up.

Go build an I-V curve tracer. Have fun!
Appendix 2: Alternate I-V Curve Tracer Hardware Configuration

Figure A2: Circuit configuration using alternate Boarduino Arduino.