Simulating MOSFETS in Spice

To perform simulation with MOSFETS in our circuits will require that we learn a few more things about Spice simulation. To add to our repertoire we will need to know how to:

- Include models from other files
- Use subcircuits
- Run a transient simulation instead of DC steady state
- Make a plot of voltage versus time on a graph

The circuit we will use as an example is shown below.

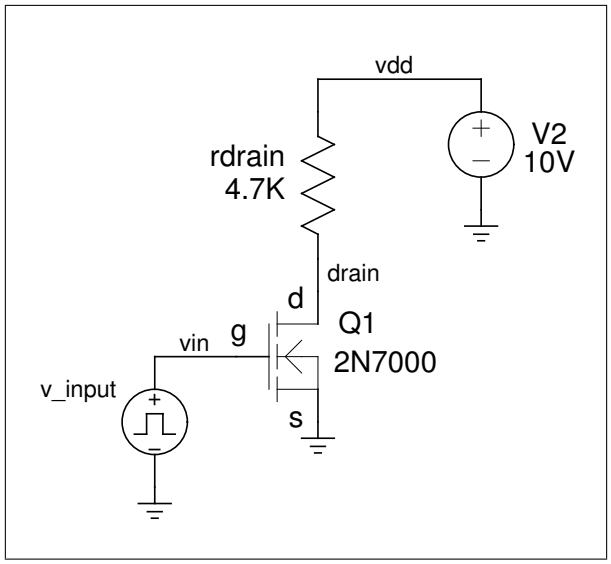


Figure 1: MOSFET Circuit for Simulation

From the schematic we see that our MOSFET is the 2N7000. This is an N-Channel enhancement-mode MOSFET that is cheap, common and rugged. To use an accurate model of the part, the Spice model file was copied from the manufacturers website (NXP in this case). Manufacturers typically supply these models that closely represent the behavior of their parts. The spice model file was copied into a file called 2n7000.inc. The `.inc` suffix is not special in any way except to indicate to the user that this is an *include file*. In other words, it's a file intended to be read into the spice file when the simulation is run. To the simulator, it's as if the contents of the file were actually inside the spice file itself.

We can include as many files as we like and in doing so can hide the complexity of the models and subcircuits and be able to focus only on the simulation at hand. The `.include` directive allows us to have hierarchy in our spice file, not just one gigantic spice file with everything in it.

The model for the 2N7000 actually includes quite a few components including two separate transistors. However, we need not worry about the structure or complexity of that model. All we

need to know is how to connect to it. Specifically, we need to know how to connect to the gate, drain and source terminals. Looking into the model, we see in a self-documenting form, the order connections are to be made in; namely drain, gate, source.

The first few lines of the model for the 2N7000 in the file `2n7000.inc` look line this:

```
.SUBCKT 2n7000 1 2 3
**************************************
*      Model Generated by MODPEX     *
*Copyright(c) Symmetry Design Systems*
*         All Rights Reserved        *
*    UNPUBLISHED LICENSED SOFTWARE   *
*   Contains Proprietary Information *
*      Which is The Property of      *
*     SYMMETRY OR ITS LICENSORS      *
*Commercial Use or Resale Restricted *
*   by Symmetry License Agreement    *
**************************************
* Model generated on Mar 31, 04
* MODEL FORMAT: SPICE3
* Symmetry POWER MOS Model (Version 1.0)
* External Node Designations
* Node 1 -> Drain
* Node 2 -> Gate
* Node 3 -> Source
```

Therefore, when we call this subcircuit model for the transistor we know the order of connection. The first terminal is the drain, then gate and source. In the netlist we see:

```
XQ1      drain    vin     gnd    2n7000  ;call subcircuit for the 2N7000
```

The ”X” in the first column tells us that this element is a subcircuit. Any characters can be used after the ”X” to identify the element to the author. Next, are, in order, the node connections to be made to the drain, source and gate terminals of the MOSFET. This is much like a function call in a procedural language such as C where the node names are the actual parameters and the model terminals are the formal parameters. So, in this instantiation of the 2N7000, the drain is connected to a node called `drain`, the gate to a node called `vin`, and the source to a node called `gnd`. The last item is the name of the model `2n7000` which is the name given to the model in the include file.

To run a transient simulation, we include the `tran` statement in the `.control` block. The simulator will find the DC values of the circuit at time zero, and then step forward in time in the step size given and solve the circuit again. This process is repeated until the transient simulation end time is given. Our `tran` statement given is:

```
tran 1.0u 50ms
```

This indicates that a transient simulation is to be run with time steps of 1 microsecond until the simulation time reaches 50 milliseconds. Note that the *s* is optional as it is ignored by the simulator. The units are understood to be in seconds.

To provide the time varying input to the transient simulation, we add a pulsed DC source. This is a source similar to the DC source but instead of a steady output voltage, its output is a pulse of variable duration. The pulse-type DC source has a number of parameters that must be specified. The general form of the pulse modifier to the DC source is as follows:

```
pulse( v1 v2 td tr tf pw per) where;
 v1 - initial value
 v2 - pulsed value
 td - delay to starting edge of pulse
 tr - rise time of pulse
 tf - fall time of pulse
 pw - pulse width
 per - period, or the time before the pulse repeats
```

In our example, the instantiation of the source provides a pulse from zero to five volts with a initial delay of 10ms. The rise and fall time of the edges is 10 ms and the pulse width is also 10ms. The pulse will repeat with a period of 50ms.

```
v_input  vin      gnd     0.0 pulse(0 5 10m 10m 10m 10m 50m)
```

To get a screen plot of our simulation results, we use the `plot` command within the `.control` block. We used it like this:

```
plot v(vin) v(drain) xl 0 50ms
```

To make the plots more readable, some other statements are used in the control block:

```
  set hcopypscolor=0
  set color0=rgb:f/f/f
  set color1=rgb:0/0/0
```

These commands provide a white background and black axes. If we want to create the postscript file `output.ps` to print the plots, the addition of the following commands enables that:

```
set hcopydevtype=postscript
hardcopy output.ps  V(in) V(drain) xl 1ns 50ns
```

The complete spice file for our circuit looks like this:

```
.title Sample MOSFET circuit

.options badchr=1 ingold=1 numdgt=4

.include 2n7000.inc  ;include the subcircuit and model for 2N7000
*model terminals in order are: drain,gate,source

*Voltage source at input to MOSFET
*transitions from 0v to 5v after 10ms delay, pulse width is 10msec, repeats in 50ms
```

```
v_input  vin      gnd     0.0 pulse(0 5 10m 10m 10m 10m 50m)
Vdd      vdd      gnd     10      ;power supply for the circuit
XQ1      drain    vin     gnd     2n7000  ;call subcircuit for the 2N7000
rdrain   vdd     drain    4700    ;resistor in drain lead

.control
  set hcopydevtype=postscript
  set hcopypscolor=0
  set color0=rgb:f/f/f
  set color1=rgb:0/0/0
  tran 1.0u 50ms
  plot v(vin) v(drain) xl 0 50ms
  hardcopy  output.ps  v(in) v(drain) xl 1ns 50ns
*  gnuplot switch_closes v(vin) v(drain) v(ctrlp) xl 0 0.1
*  gnuplot delay_to_off  v(vin) v(drain) v(ctrlp) xl 0 15
.endc
.end
```

If we run ngspice on this spice file, we get the following:

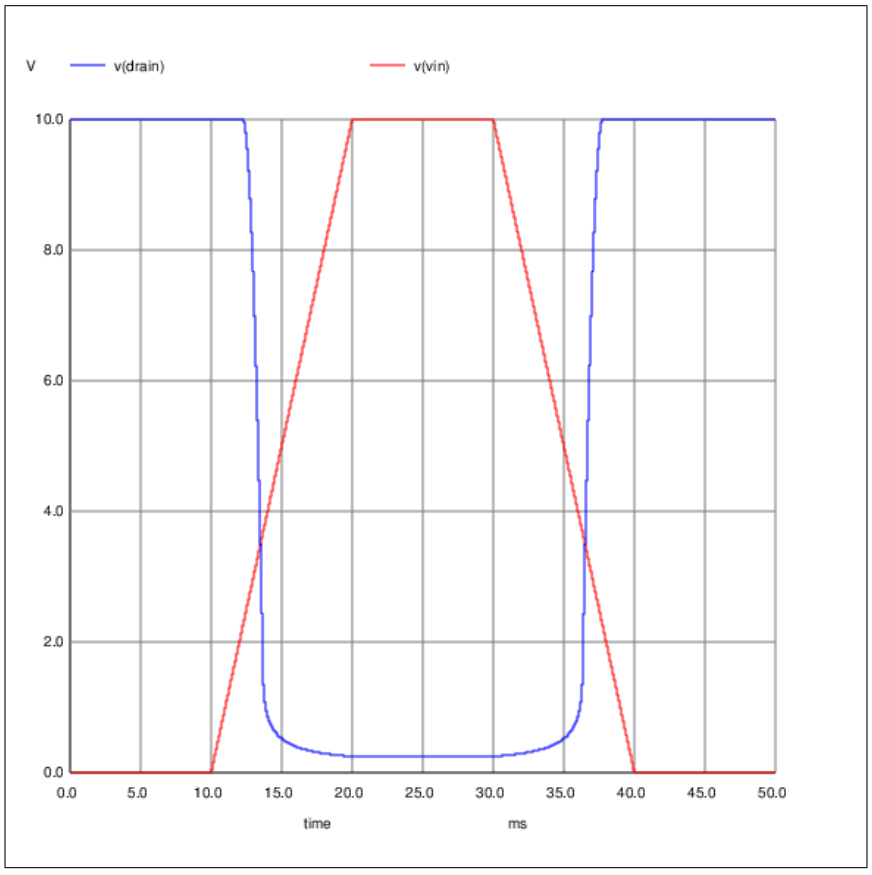


Figure 2: Output Plot for MOSFET Circuit Simulation

You can clearly see that when the input voltage to the gate V_{gs} exceeds about 2 volts, the transistor begins to turn on. By the time the V_{gs} has reached 8 volts, the transistor has pulled it drain down to nearly zero volts.

Comparing the MOSFET to a Nearly Ideal Switch

Before, we had asserted that the simplest model of a MOSFET was that of a mechanical switch. When the MOSFET was on, it has a very low resistance between drain and source terminals. When off, it has a very high resistance between drain and source. This is a very much like a mechanical switch.

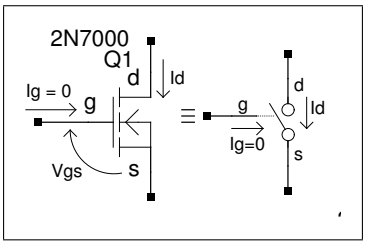


Figure 3: MOSFET Schematic Symbol and its Switch Model

We can use the spice model of a voltage controlled switch to compare the behavior of the MOSFET with a mechanical switch that is actuated by a voltage. With the MOSFET in our previous netlist, we included a file (`2n7000.inc`) that contained the model of the component we wanted to use. The spice model for the switch is very simple, so we simply include describe the model in the spice file. We do so like this:

```
.model switch_model sw(vt=2.0 ron=1.0 roff=10Meg) ; pushbutton switch model
```

Here we declare a model named *switch_model* which is a switch which remains open until the control voltage (vt) reaches 2v. When off or open, its resistance (*roff*) is 10 megohms.

To utilize the spice switch model we use the `s` or *switch* element by using the `s` in column one of the spice file. Following that, we list the two terminals for the switch that are used in our circuit; in this case, `sw_term1` and `sw_term2`.

Then we list the terminals, between which, when the potential reaches 2 volts, tells the switch to close. Those terminals are `ctl_pos` and `ctl_neg`, which is the reference terminal.

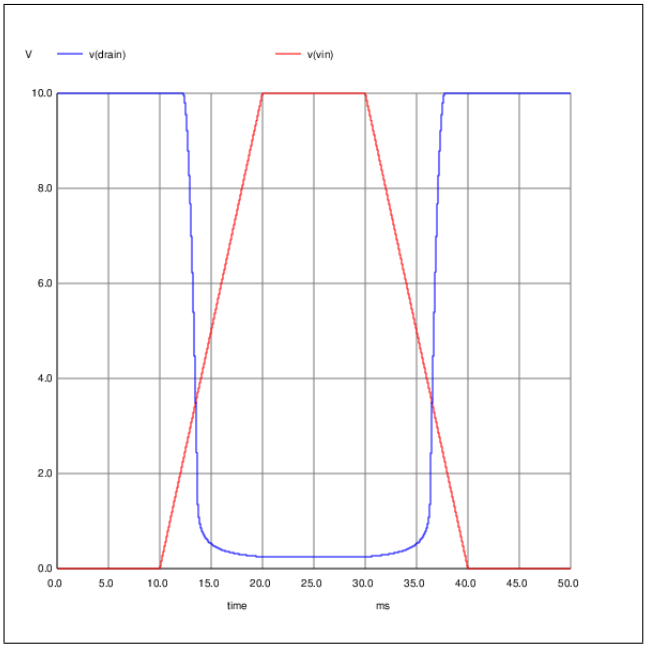
Finally, we call out the model we wish to use, `switch_model`, and its default setting of `off`. The complete instantiation of the switch is formatted like this:

```
s1  sw_term1  sw_term2  ctl_pos  ctl_neg  switch_model off ; an instantiated switch with specified model
```

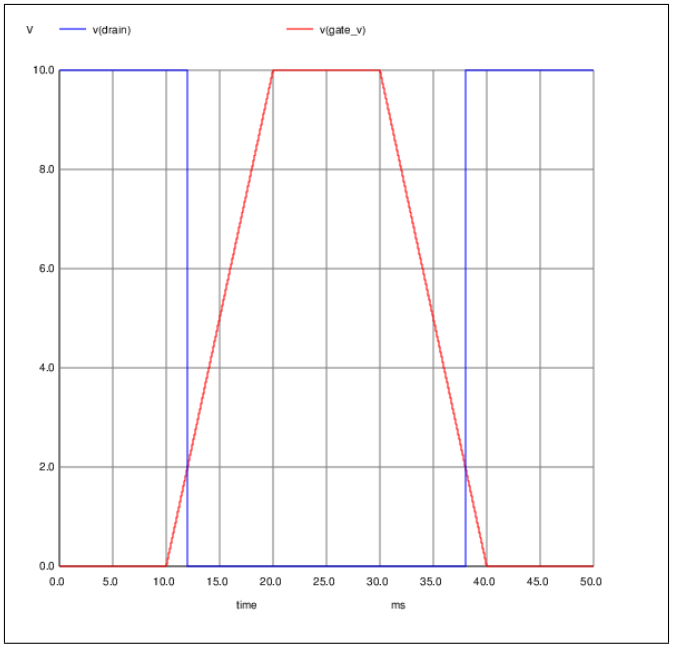
Now we can take our previous circuit and substitute the spice switch for the MOSFET and see any differences. The new spice netlist would look like ths:

```
.title switch acting like a MOSFET
.options badchr=1 ingold=1 numdgt=4

v_input  gate_v  gnd    0.0 pulse(0 5 10m 10m 10m 10m 50m);control input to switch
Vdd      vdd     gnd    10                                 ;power supply for the circuit
s1       drain   gnd    gate_v  gnd  switch_model off      ;the switch model of MOSFET
rdrain   vdd     drain  4700                               ;resistor in drain lead

.control
  set hcopydevtype=postscript
  set hcopypscolor=0
  set color0=rgb:f/f/f
  set color1=rgb:0/0/0
  tran 1.0u 50ms
  plot v(gate_v) v(drain) xl 0 50ms
  hardcopy  switch_as_mosfet.ps v(gate_v) v(drain) xl 1ns 50ms
.endc
.end
```

Here the two simulations may be compared. The two circuits behave very nearly identically, except for some rounding at the edges of the drain waveform. Thus, we see that a switch really is a good model for the basic MOSFET operation.



(a) MOSFET in circuit



(b) Switch acting as MOSFET