SPICE 4: Diodes

Chris Winstead

ECE 3410. Spring, 2015.

Preparing for the Exercises

In this session, we will simulate several diode configurations corresponding to your lab assignment, and we'll look at a few additional examples of diode operation. Create a new directory for this session:

```
3410/
└── spice/
    ├── lab1/
    ├── lab2/
    ├── lab3/
    └── lab4/
```

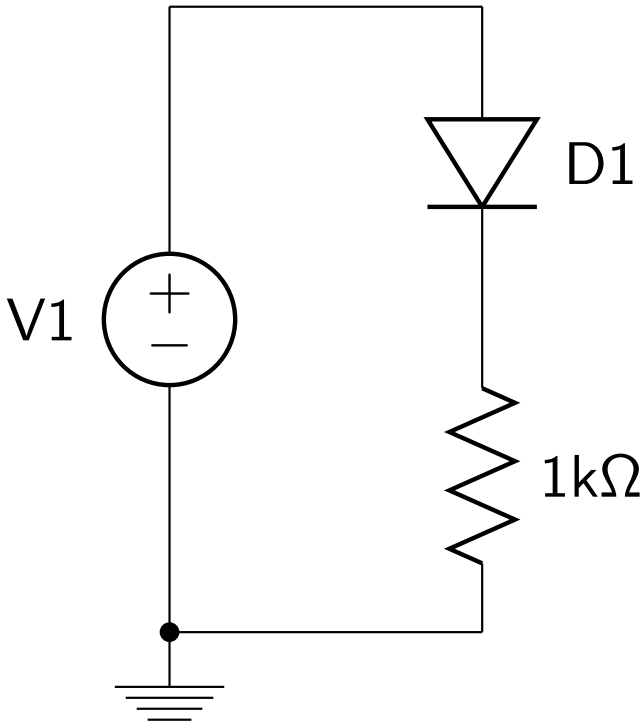
Obtain the Example Files

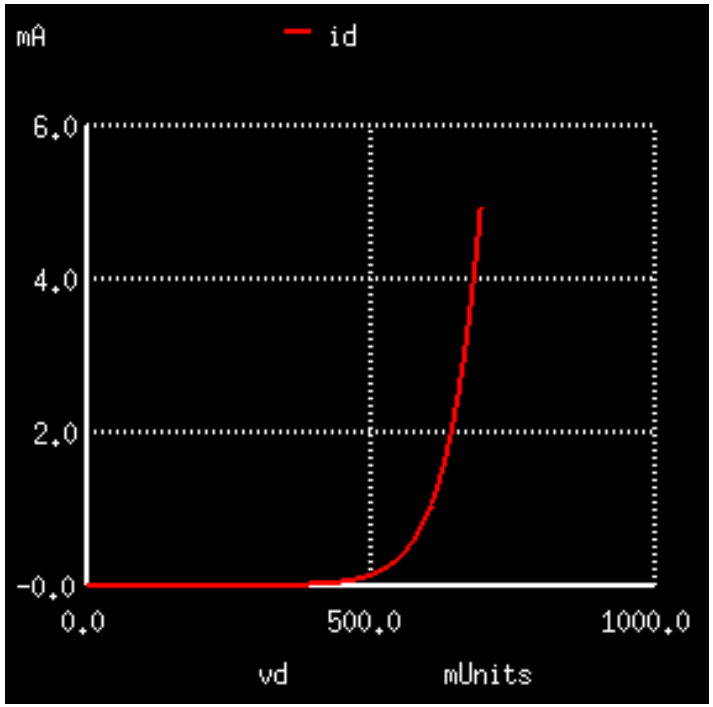
In this lab, you will download a collection of files containing the exercises: `exercise1.sp`, `exercise2.sp`, and `exercise3.sp`. Download these files and place them in these locations:

```
3410/
└── spice/
    ├── [files from previous labs]
    └── lab4/
        ├── exercise0.sp
        ├── exercise1a.sp
        ├── exercise3.sp
        ├── exercise4.sp
        ├── boost_converter.sp
        └── superdiode.sp
```

Study and Simulate Exercise 0

Navigate to your `lab4` directory and open the `exercise0.sp` file. This simple exercise performs an operating point simulation to verify the diode's forward voltage drop at 1 mA, and also performs a DC sweep analysis to reveal the diode's I-V transfer characteristic.





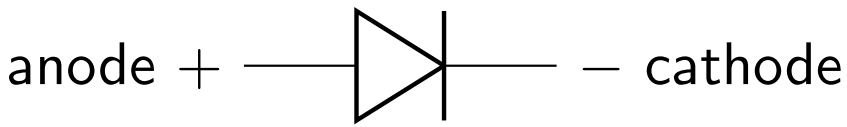
It should confirm that the forward drop is about 600 mV.

Diodes in SPICE

One thing you should notice in `exercise0.sp` is the diode declaration:

```
D[name] [anode] [cathode] [modelname]
```

This declaration is similar to resistors or capacitors, but the diode is directional. The first declared node is the anode (+) and the second one is the cathode (-). When the diode is forward biased, there should be a positive voltage at the anode relative to the cathode.



Diode Models

When a diode is declared, the last entry is the `modelname`, which references a detailed model specific to the type of diode you want to simulate. In this lab, we are using the common 1N914 diode. The model for this diode is declared in `lab_parts.md` in these lines:

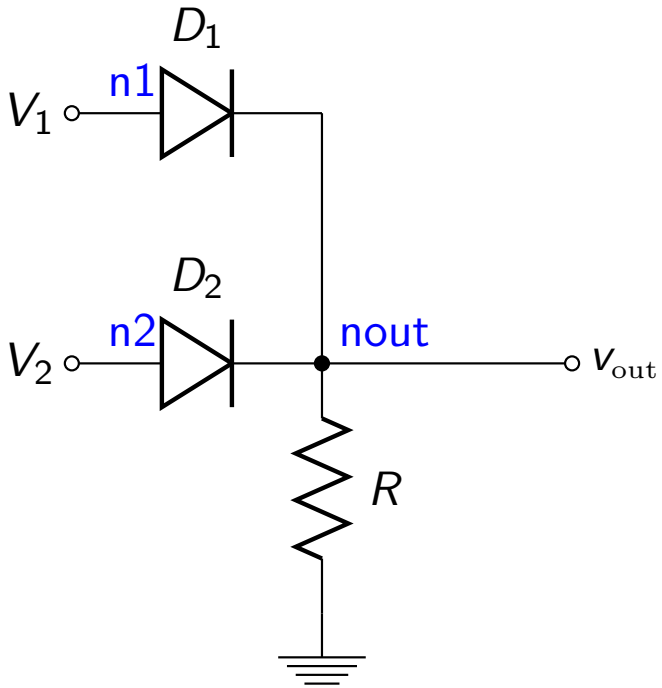
```
.MODEL D1N914 D  ( IS=6.2229E-9 N=1.9224 RS=.33636 IKF=42.843E-3 CJO=764.38E-15 M=.1001
+ VJ=9.9900 ISR=11.526E-9 NR=4.9950 BV=100.14 IBV=.25951 TT=2.8854E-9)
```

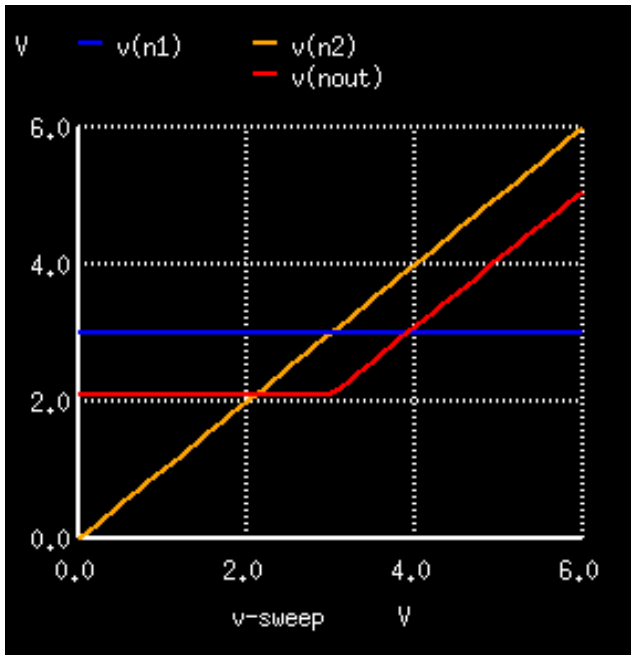
(The '+' at the start of a line indicates continuation of the previous line.)

Clearly a lot of parameters are needed to model a diode's behavior. The parameters familiar to us are `Is` and `N`, which are part of the standard exponential diode equation. The remaining parameters relate to the detailed physics affecting diode operation.

Study and Simulate Exercise 1A

Next, open exercise1a.sp and study its contents. This exercise simulates one of the diode logic circuits from Pre-Lab exercise 1.





In the console, you should see measurements for all the cases requested in exercise 1. It also reports the diode's forward drop in this circuit, which is near 0.6 V. Create a `log.txt` file and record these measurement results.

Exercise 1B

Now modify the contents of `exercise1a.sp` to implement the second circuit shown in Exercise 1 of the pre-lab. Save the modified file as `exercise1b.sp`, and run the simulation. Save the results in your log.txt file.

Exercise 2

Next, create a new file called `exercise2.sp` and create a model of the circuit shown in Exercise 2 of the pre-lab. Perform a transient simulation for the scenario described in that exercise.

Plot the output waveform and save it as a hardcopy plot called `halfwave.ps`. Use a `meas` command to measure the peak value of the output waveform. Save the result in your `log.txt` file.

Exercise 3

The code for `exercise3.sp` is provided, and corresponds to the peak rectifier circuit shown in Exercise 3 of the prelab. In this simulation, the control script uses `alter` commands to simulate the three frequencies specified in the exercise. In each simulation, two meas commands are used to measure the peak output voltage and the peak-to-peak ripple amplitude. These measurements are reported to the console.

Notice that the `meas` command uses `FROM` and `TO` modifiers like this:

```
meas TRAN vpp PP v(nout) FROM=3m TO=5m
```

Exercise 3 (cont...)

This tells SPICE to look only in the time window from 3 ms to 5 ms. The purpose of this is to capture the steady-state behavior and ignore the initial transient response.

Record the measurement results in your `log.txt` file. Optionally save the plots for use in your lab report.

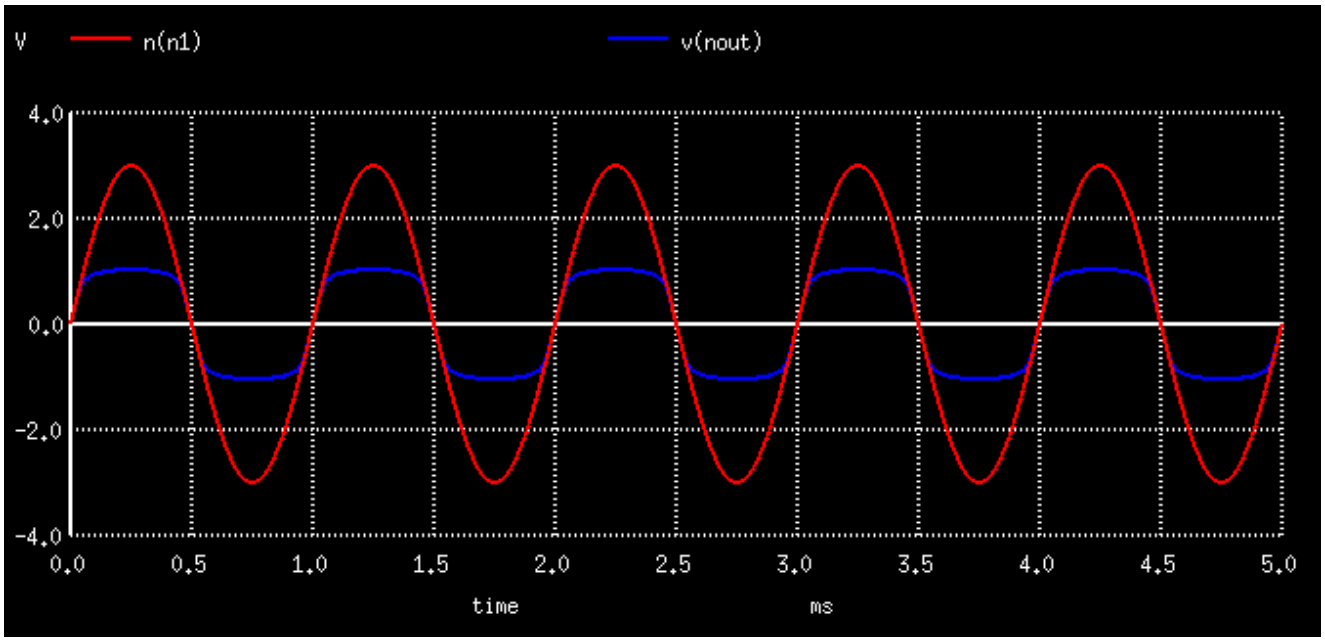
Whether to use AC or TRAN?

In the case of the peak rectifier, our goal is to measure the output ripple amplitude. Why not use AC simulation for this?

AC simulation will not work because the circuit's behavior is entirely dependent on the diode's non-linear large-signal behavior. AC simulation is a linearized simulation method that is suitable for simulating small-signal behavior only when a circuit is biased to give an approximately linear response. If we try running an AC simulation, it will compute the initial DC bias condition and find the the diode is OFF, and it will hold that condition – the simulation will not account for alternating ON/OFF states or the progressive charging of C.

Exercise 4

The file for exercise 4 is provided for you. This file models the voltage-limiting circuit shown in the pre-lab. Open the file and study the commands used for this simulation, then run it with NGSpice. The simulation should display a plot that looks like this:



Exercise 4 (cont...)

The console output should also reveal measurements for the max and min ouputs, which should correspond to two forward-bias voltage drops.

Record these measurements in your `log.txt` file and optionally save the plotted waveform for use in your lab report.

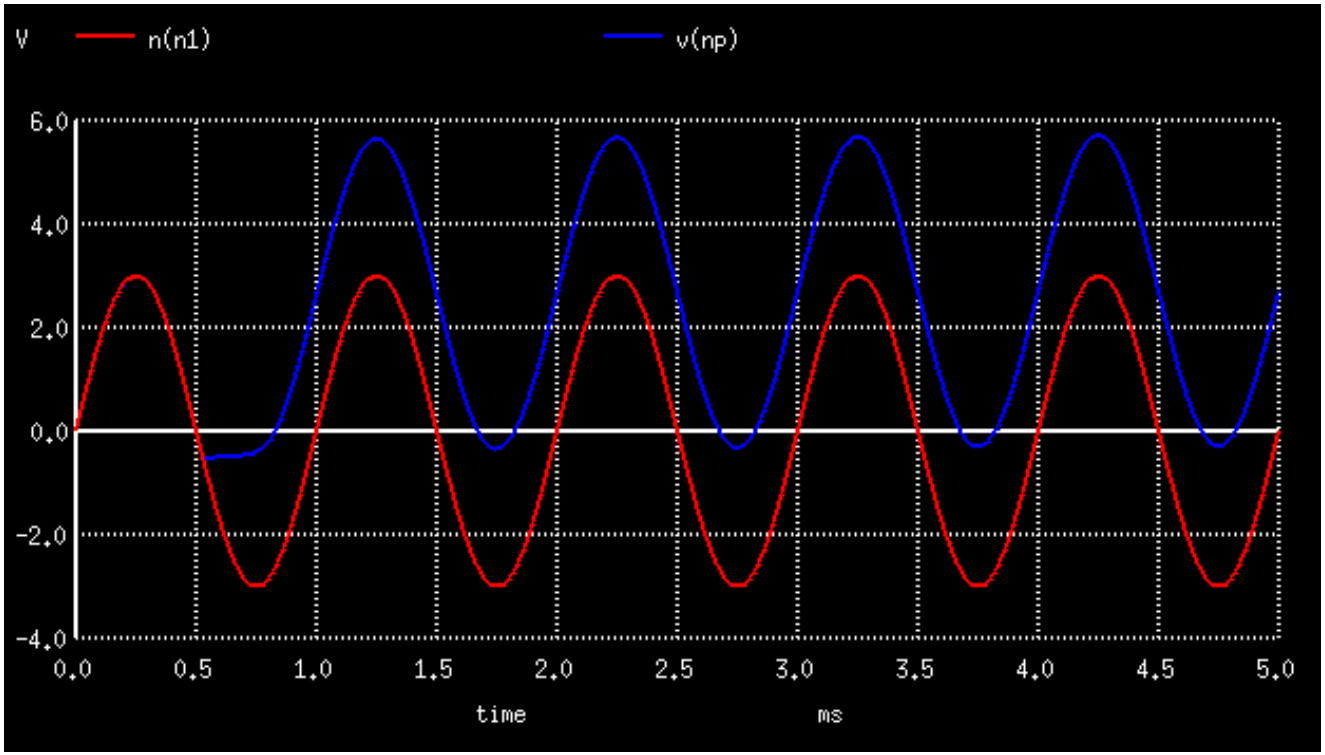
Exercise 5

For exercise 5, create a new SPICE file called `exercise5.sp` and model the clamped-capacitor circuit shown in pre-lab exercise 5. Use a `tran` command to perform the simulation and use a `meas` command to measure the output's minimum value and the peak-to-peak amplitude.

Record the measurement results in your `log.txt` file and save a hardcopy of the waveform in a file named `clampedcap.ps`.

Exercise 5 (cont...)

Your result should look like this:

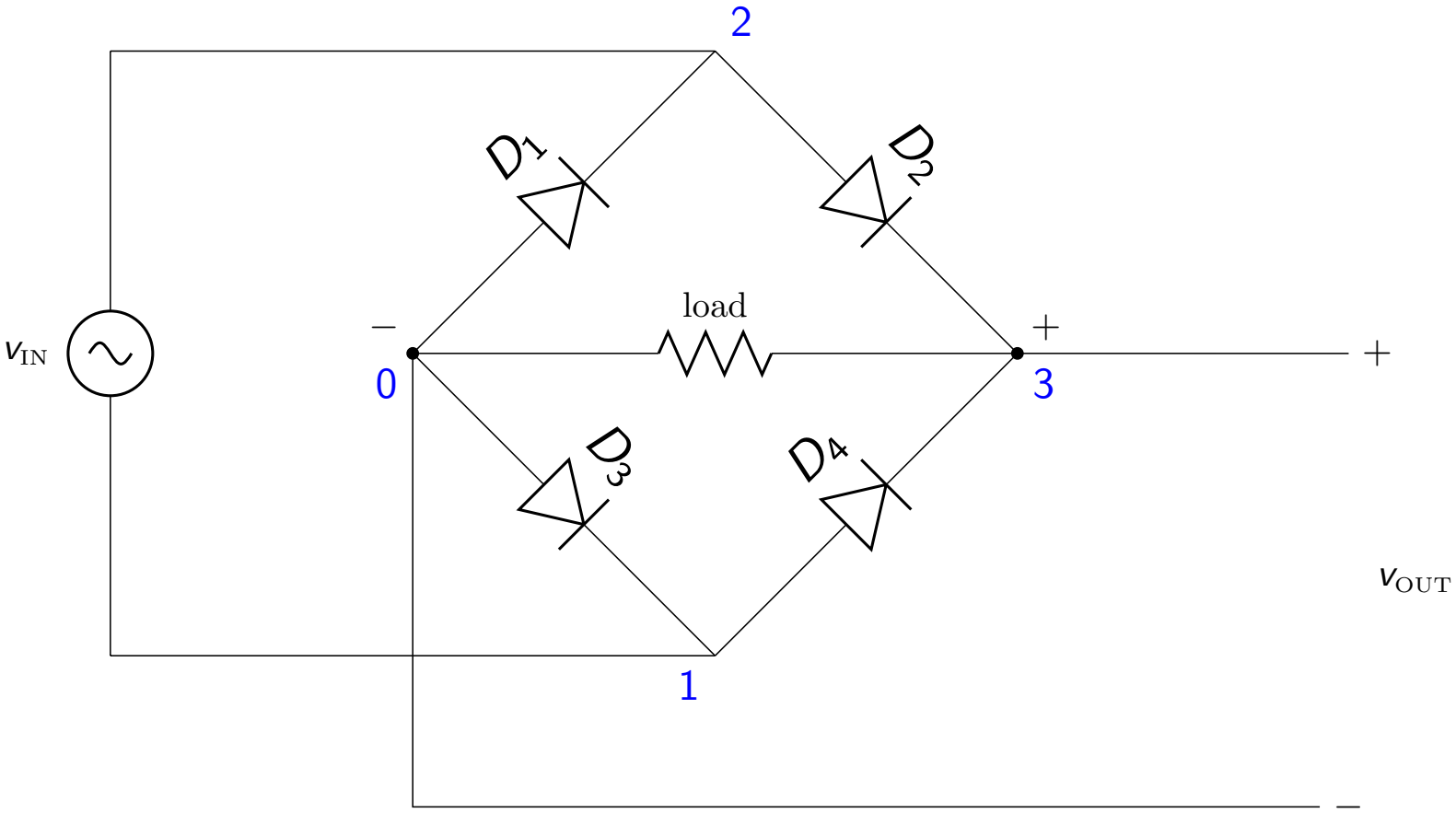


Supplemental Exercises

In addition to the pre-lab circuits, some supplemental models are included in this tutorial. These include a full-wave bridge rectifier, a "super-diode" precision half-wave rectifier, and a DC-to-DC boost converter.

Bridge Rectifier

Open `bridge_rectifier.sp` and examine its contents. This file models the standard full-wave rectifier:

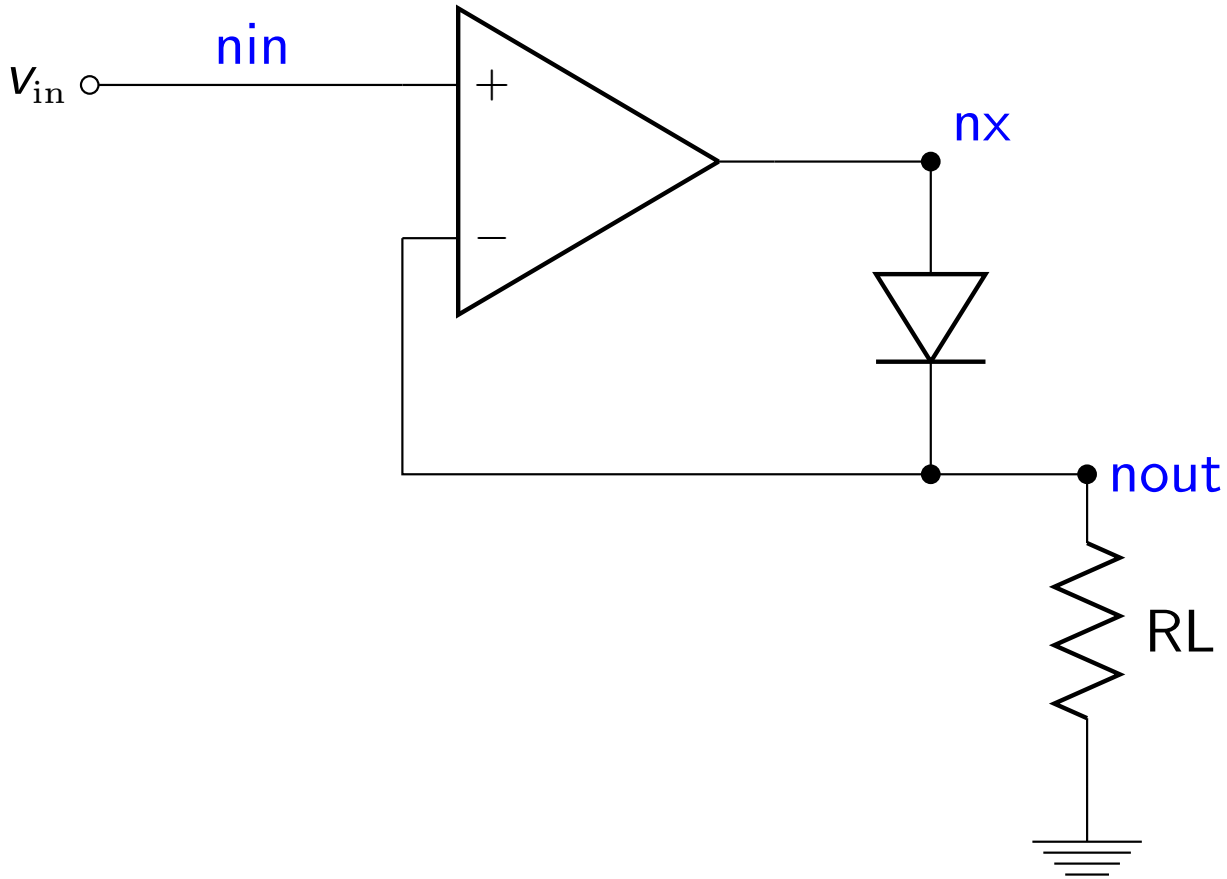


Bridge Rectifier Exercise

Modify `bridge_rectifier.sp` by adding a `meas` statement to measure the peak output. Report the result in your `log.txt` file.

Super-Diode

Open `superdiode.sp` and examine its contents. This file models a precision rectifier circuit:



Super-Diode Exercise

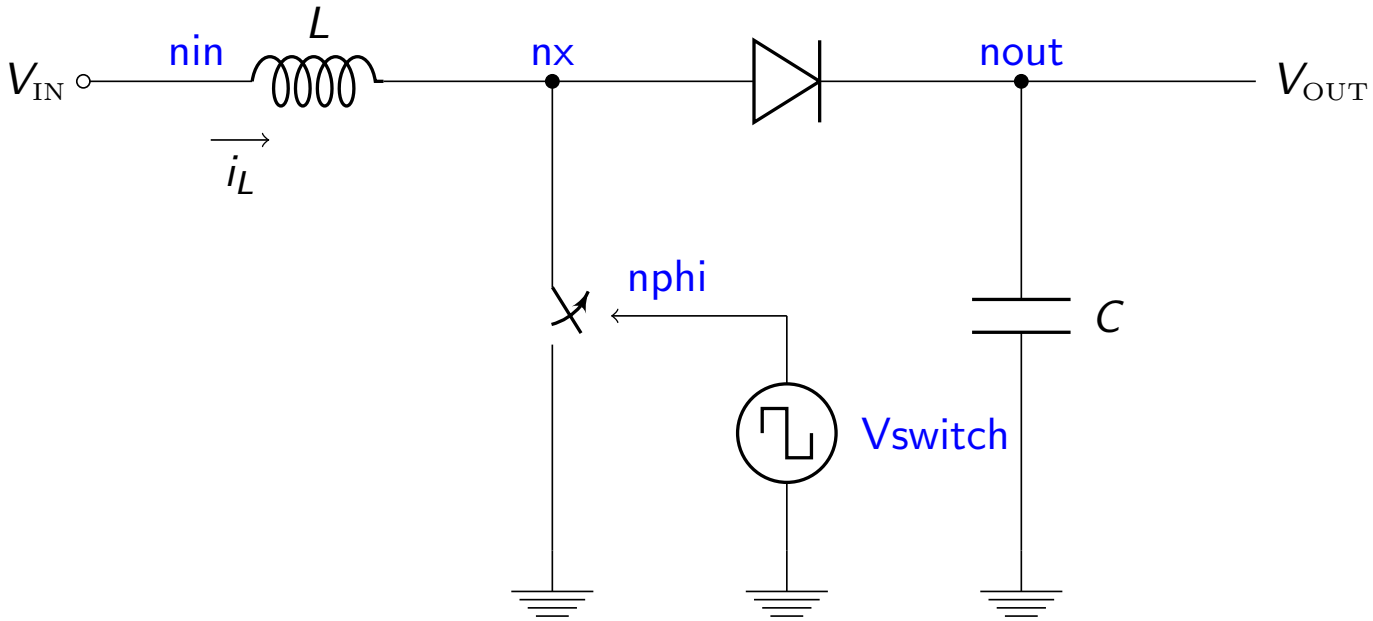
Run NGSpice on `superdiode.sp`. It will generate three plots, one showing v_{IN} and v_{OUT} together, another showing v_{OUT} by itself (so you can clearly see it), and another showing v_{IN} and v_x. Note that v_x drops all the way to V_{SS} when the diode is off.

The wide swings in v_x create a problem for this circuit at high frequencies, where the op amp's slew rate will prevent it from restoring the value of v_x. To test this, modify `superdiode.sp` to simulate a waveform at 10 kHz. Run the simulation and explain what happens. (Note, you will have to change the `tran` simulation parameters to a step-size of 1 µs and an end-time of 500 µs).

Save hardcopies of the three plots as `sd1.ps`, `sd2.ps` and `sd3.ps` (in no particular order).

Boost Converter

The final supplemental file is `boost_converter.sp`. Open this file and examine its contents. It models the DC-to-DC boost converter circuit:



Switches in SPICE

The boost converter circuit uses an ideal switch controlled by a clock signal ϕ. In SPICE, a switch is an element that toggles between an ON-resistance R_{on} and an OFF-resistance R_{off} when its control signal crosses a specified threshold voltage. These parameters are set in a model declaration:

```
* switch model:
.model switch sw(Ron=25, Roff=100000, Vt=0.001, Vh=0.0001)
```

According to this model, the switch turns ON when the control signal crosses the threshold `Vt`, and turns OFF when the control signal cross the hysteresis voltage `Vh`. For stability reasons, `Vt` should usually be larger than `Vh`.

Declaring a Switch Instance

The switch component is declared using the following syntax:

```
S[name] [n+] [n-] [control+] [control-] [modelname]
```

In the boost converter example:

```
S1 nx 0 nphi 0 switch
```

This places a switch between `nx` and ground. The switch will close whenever V(`nphi`)>`Vt`.

The Pulse Signal

The switching clock is set using a pulse source:

```
V[name] [n+] [n-] PULSE(V1 V2 TD TR TF PW T)
```

In this declaration, the pulse will switch from voltage `V1` to `V2` after delay `TD`, with rise/fall times `TR`/`TF`. It will stay at `V2` for time `PW` and then fall. The pulse is repeated with period `T`. In the boost converter example, the pulse statement is

```
Vswitch nphi 0 PULSE(0 1 0 1n 1n 'D*T' 'T')
```

This declares a clock signal that switches between 0V and 1V, with zero delay, and with rise/fall time of 1 ns. The pulse width is set by the duty cycle parameter D and the period parameter T, which are declared at the top of the file.

Theoretical Prediction

According to the theory described in the notes provided on Canvas, the steady-state output voltage should be

$$V_{\mathrm{OUT}} = V_{\mathrm{IN}} \frac{1}{1 - D}.$$

For our circuit, $V_{\mathrm{IN}} = 4\,\mathrm{V}$ and $D = 0.75$, so we expect to see $V_{\mathrm{OUT}} = 16\,\mathrm{V}$. Run the simulation and see what happens (it may take a long time to complete, due to all the switching events).

In practice, the circuit doesn't quite achieve the expected output. Can you explain why?

Boost Converter Exercise

Now try operating the boost converter as a simple voltage doubler by setting $D = 0.5$. Increase the `tran` end-time to 10 ms and repeat the simulation. How close does it get to the expected output value?

Save a hardcopy of your V_{OUT} plot as `boost.ps`.

What to Turn In

Once you are finished with these exercises, prepare a zip archive with the following files:

- `log.txt`
- `exercise1b.sp`
- `exercise2.sp`
- `exercise5.sp`
- All requested hardcopy plots.

Submit this zip archive on Canvas to complete your assignment.