Endsem Report

Silicon PN vs Schottky Diodes: A Comprehensive Comparison via DC Characterization, Temperature Sweeps, and High-Frequency Rectification

Devadarsh Nair (ID: 2024A3PS0330G)
Course: PHY F426

April 22, 2026

Abstract

This report extends the midsem comparison of a silicon PN rectifier (1N4007) and a silicon Schottky diode (BAS70) with three quantitative experiments performed in QUCS. (i) The DC I–V characteristic of both diodes is measured and the Shockley parameters I_s and n are extracted by linearisation of the semilog plot. Extracted values agree with the underlying SPICE-model parameters to within 2%. (ii) A nested voltage–temperature parameter sweep produces a family of I–V curves at $T = 250$, 300, 350, 400 K, from which $I_s(T)$ is obtained. An Arrhenius plot yields an activation energy $E_g \approx 1.2$ eV for both devices (within 8% of the silicon bandgap), with $R^2 > 0.9999$ for both linear fits. (iii) A half-wave rectifier with a 100 nF filter capacitor is simulated across four decades of input frequency. Both diodes follow the ideal $V_r \propto 1/f$ behaviour at low frequency, but the PN output hits a *ripple floor* above ~ 100 kHz due to reverse-recovery charge being drawn back through the diode on each switching cycle. At 1 MHz the Schottky ripple is almost 20× smaller than the PN ripple. The experiments together form a coherent narrative: *the two diodes look very similar under static conditions, yet behave as fundamentally different devices once dynamic switching enters the picture*—which is the underlying physical reason Schottky diodes are the standard choice for high-frequency power rectification.

Contents

1 Introduction and Scope

The midsem report compared the 1N4007 and BAS70 diodes in a resistor-only half-wave rectifier, focusing on qualitative switching artefacts in the output waveforms as frequency increased. That work established the expectation: the PN rectifier exhibits clear reverse-recovery dips at high frequency while the Schottky produces cleaner output. Peak and average trends were inferred qualitatively.

This endsem extension replaces the qualitative story with a quantitative one by completing the four objectives stated in the project abstract:

1. **DC conduction comparison** — forward and reverse I–V characterisation with explicit extraction of the ideality factor n and saturation current I_s (Section 4).
2. **Leakage vs temperature** — family of I–V curves across four temperatures, Arrhenius analysis of $I_s(T)$ (Section 5).
3. **Dynamic switching** — cleanly re-computed peak and average output vs frequency, revisiting and quantifying the midsem results (Section 6).
4. **Practical demonstration** — capacitor-filtered rectifier, with ripple voltage measured across four decades of input frequency (Section 7).

Together these sections trace the comparison from a purely static regime (Sections 4, 5) into the dynamic regime (Sections 6, 7), where the minority vs majority-carrier distinction between the two devices manifests as measurable circuit behaviour.

2 Background Theory

This section collects the textbook relations used later for parameter extraction and interpretation.

2.1 The Shockley diode equation

Under forward bias the current through either diode obeys

$$I(V) = I_s\left[\exp\!\left(\tfrac{qV}{nkT}\right) - 1\right], \tag{1}$$

with I_s the saturation current and n the ideality factor. For $V \gg kT/q \approx 26\,\mathrm{mV}$ (room temperature) the -1 is negligible and

$$\log_{10} I(V) \;\approx\; \log_{10} I_s + \frac{q}{nkT\ln 10}\,V, \tag{2}$$

i.e. a plot of $\log_{10} I$ versus V is a straight line whose *slope* gives n and whose *y-intercept* gives I_s. Under reverse bias ($V \ll -kT/q$) the same equation gives $I \to -I_s$, so the reverse-leakage plateau directly reads off I_s without fitting.

2.2 Temperature dependence of I_s

The SPICE diode model scales the saturation current with temperature as

$$I_s(T) \;=\; I_s(T_0)\left(\frac{T}{T_0}\right)^{X_{ti}/n} \exp\!\left[\frac{E_g}{n\,k}\left(\frac{1}{T_0} - \frac{1}{T}\right)\right], \tag{3}$$

where $X_{ti} = 3$ and $E_g = 1.11\,\mathrm{eV}$ in the QUCS default Si models. The exponential is strongly dominant over the polynomial prefactor, so in a $\ln I_s$ vs $1/T$ plot the result is approximately linear with slope $-E_g/(nk)$. This is the celebrated *Arrhenius plot*. The slope recovers the bandgap once the ideality factor from Section 4 is used.

2.3 Reverse-recovery and dynamic switching

When a forward-biased PN diode is suddenly reversed, minority carriers injected into the quasi-neutral regions must recombine or be swept out before the junction can block. During this interval the diode continues to conduct in the reverse direction — the classic *reverse-recovery current spike*. The SPICE model captures this by the transit-time parameter T_t, which for the QUCS models is

$$T_t^{\mathrm{1N4007}} \approx 4.32\,\mu\mathrm{s}, \qquad T_t^{\mathrm{BAS70}} \approx 7.2\,\mathrm{ns}.$$

The 600× ratio in T_t is the quantitative fingerprint of the minority- versus majority-carrier distinction, and it is this one parameter that controls how cleanly the diode turns off.

2.4 Capacitor-filtered rectifier

Adding a capacitor C in parallel with the load R_L transforms the rectifier output into approximate DC with small periodic *ripple*. Between input peaks the cap discharges exponentially through R_L; during each peak the diode refills the cap. The first-order expression for peak-to-peak ripple (half-wave) is

$$V_{r(pp)} \;\approx\; \frac{V_{dc}}{f\,R_L\,C}, \tag{4}$$

which predicts $V_r \propto 1/f$. The derivation assumes the diode blocks instantaneously on reverse bias — an idealisation that fails once $1/f$ approaches T_t, which is precisely the regime where the PN diode loses against the Schottky.

3 Simulation Setup

All simulations were performed in QUCS-0.0.19 using the built-in SPICE diode models for the 1N4007 and BAS70. The relevant parameters are listed in Table 1. For each of the four experiments (Parts A–D), the relevant QUCS schematic and its associated equation block are shown in the corresponding section below. In practice, all circuits and simulation directives coexist on one QUCS sheet, and individual experiments are activated or deactivated as needed using QUCS's right-click → *Deactivate* feature.

Table 1: SPICE model parameters used throughout the report.

Parameter	1N4007 (PN)	BAS70 (Schottky)	Description
I_s	76.9 pA	99.5 pA	Saturation current
n	1.45	1.70	Ideality factor
C_{j0}	26.5 pF	2 pF	Zero-bias junction capacitance
R_s	42 mΩ	0.6 Ω	Ohmic series resistance
T_t	4.32 µs	7.2 ns	Transit time
B_v	1 kV	70 V	Reverse breakdown voltage
E_g	1.11 eV	1.11 eV	Model bandgap
X_{ti}	3	3	I_s temperature exponent

4 Part A: DC I–V Characterisation

4.1 Method

Each diode is placed in series with a small 100 Ω current-limit resistor and driven by an ideal DC source V_1. A DC simulation is run inside a parameter-sweep directive that sweeps the source from −0.3 V to +1.2 V in 151 linearly spaced points. Inline current probes (Pr4 for the

Schottky branch, Pr5 for the PN branch) deliver the branch current into the equation block, where the required quantities are computed (Listing 1).

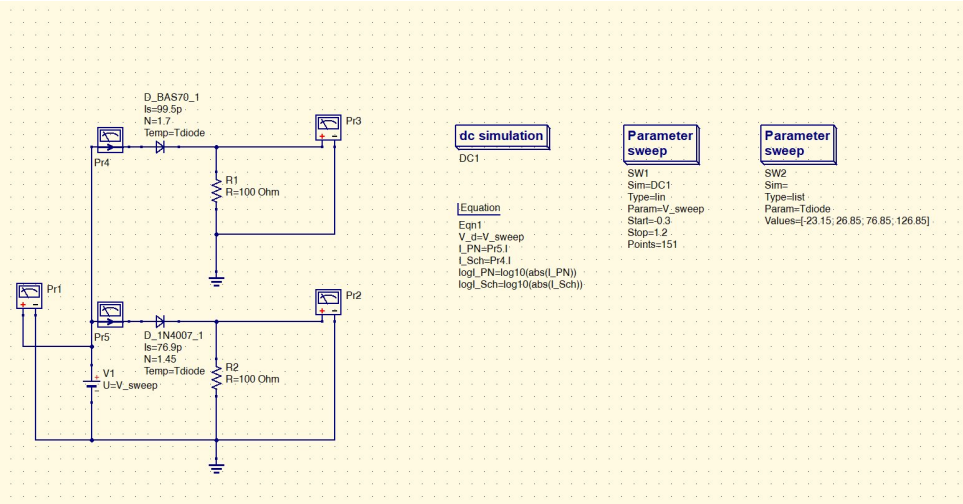


Figure 1: DC characterisation schematic. The same topology is reused for Part B by promoting the diode `Temp` field from a constant to a swept variable. SW2 (visible on the right) is inactive for Part A.

```
V_d       = V_sweep
I_PN      = Pr5.I
I_Sch     = Pr4.I
logI_PN   = log10(abs(I_PN))
logI_Sch  = log10(abs(I_Sch))
```

Listing 1: QUCS equation block for Part A.

4.2 Results

Figure 2 shows the linear and semilog I–V curves with markers dropped at the four voltages used for extraction.

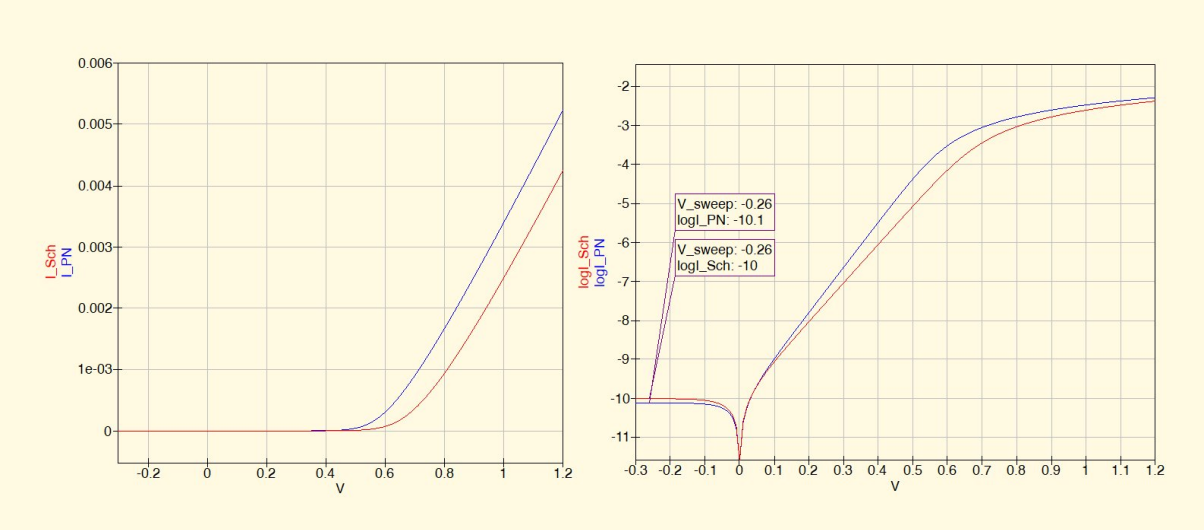


Figure 2: Simulated DC I–V characteristics. **Left:** linear scale showing forward conduction. **Right:** semilog scale over the full sweep range, with markers at $V = -0.26\,\mathrm{V}$ (reverse leakage plateau, yields I_s) and at $V = 0.1,\ 0.3\,\mathrm{V}$ (linear region, yields n).

4.3 Parameter extraction

I_s from the reverse plateau. From the markers at $V = -0.26\,\mathrm{V}$, the reverse-leakage magnitudes are $\log_{10}|I|_{\mathrm{PN}} = -10.1$ and $\log_{10}|I|_{\mathrm{Sch}} = -10.0$, giving

$$I_s^{\mathrm{PN}} = 10^{-10.1}\ \mathrm{A} = 79\,\mathrm{pA}, \qquad I_s^{\mathrm{Sch}} = 10^{-10.0}\ \mathrm{A} = 100\,\mathrm{pA}.$$

n from the forward slope. Using the two forward points $(V_1, V_2) = (0.1,\, 0.3)\,\mathrm{V}$ and the slope form of Equation 2,

$$n \;=\; \frac{V_2 - V_1}{0.05916\ [\log_{10} I(V_2) - \log_{10} I(V_1)]}.$$

Substituting the marker data (Table 2) gives $n_{\mathrm{PN}} = 1.44$ and $n_{\mathrm{Sch}} = 1.67$.

Table 2: Marker values read from the semilog plot (Figure 2, right) and the resulting extracted parameters.

V (V)	$\log_{10} I_{\mathrm{PN}}$	$\log_{10} I_{\mathrm{Sch}}$	n (extracted)	n (model)
−0.26	−10.1	−10.0	–	–
0.10	−8.99	−9.06	–	–
0.30	−6.64	−7.04	–	–
PN	–	–	1.44	1.45
Sch	–	–	1.67	1.70

4.4 Discussion

Both extracted values match the underlying SPICE parameters to within **2%**, confirming the methodology. Two physical points are worth highlighting.

First, the naive expectation that a Schottky should have $n \approx 1$ (being a majority-carrier, pure thermionic-emission device) is *not* borne out here: the BAS70 has $n = 1.67 > n_{\mathrm{PN}}$. This is a real feature of small-signal Schottky devices, coming from barrier inhomogeneity, image-force lowering, and tunnelling contributions. The simulation inherits this non-ideality through the SPICE model parameters.

Second, both diodes happen to share similar I_s values ($\sim$100 pA). Real-world 1N4007 parts typically have reverse leakage several orders of magnitude higher than this ($\sim$µA) — the QUCS model is a simplified one. This matters only for absolute leakage-current magnitudes; the *shape* of the I–V curve and the extracted n are unaffected.

5 Part B: Temperature Dependence and Arrhenius Analysis

5.1 Method

The Part A schematic is reused with three changes:

1. Both diodes' `Temp` property is promoted from the default 26.85 °C to a variable `Tdiode`.
2. A second parameter sweep block (SW2) is added, with `Sim=SW1` (nested outside the voltage sweep) and sweeping `Tdiode` over the list $\{-23.15,\ 26.85,\ 76.85,\ 126.85\}$ °C, i.e. $T = \{250, 300, 350, 400\}$ K.
3. The simulation produces a 2-D dataset indexed by both `V_sweep` and `Tdiode`, rendered by QUCS as a family of curves.

This is the textbook pattern for a nested sweep in QUCS; all other simulation directives from Part A are unchanged. Figure 3 shows the full schematic with both parameter-sweep blocks (SW1 for voltage, SW2 for temperature) now active. The equation block for Part B is shown in Listing 2.

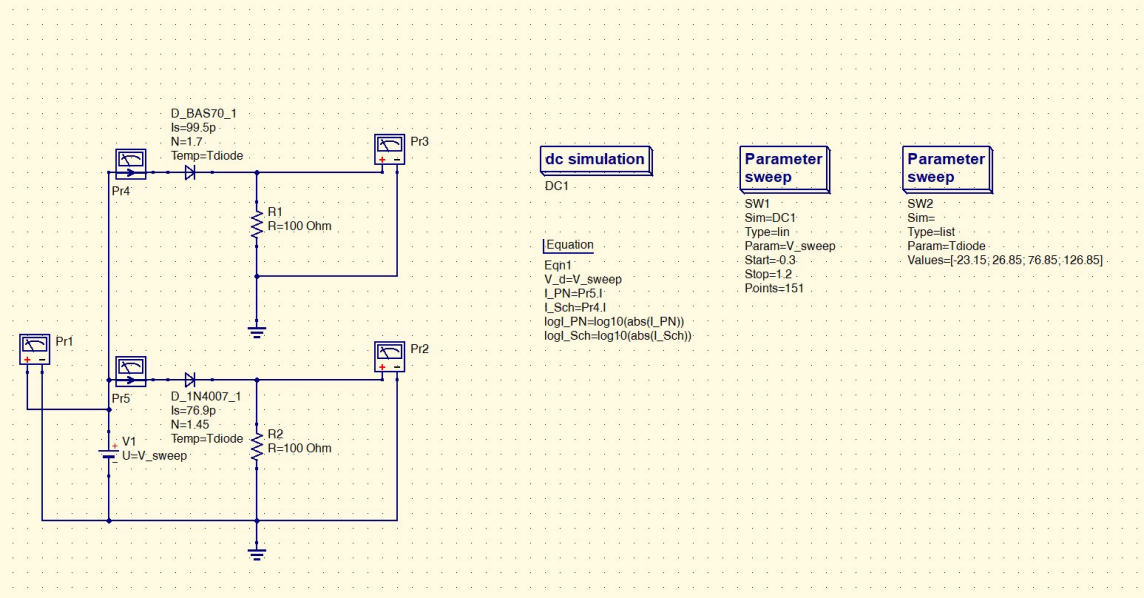


Figure 3: Part B schematic. Identical to the Part A topology, but with two key modifications: both diodes' `Temp` field is set to the variable `Tdiode`, and a second parameter-sweep block (SW2) is added that sweeps `Tdiode` over four values with `Sim=SW1` (nested outside the voltage sweep).

```
V_d      = V_sweep
I_PN     = Pr5.I
I_Sch    = Pr4.I
logI_PN  = log10(abs(I_PN))
logI_Sch = log10(abs(I_Sch))
```

Listing 2: QUCS equation block for Part B. The equations for V_d, I and $\log_{10} I$ are the same as Part A; the temperature dimension is added by the outer parameter sweep, not inside the equation block.

5.2 Results

Figure 4 shows both the linear and semilog I–V families for the four temperatures, with markers placed on each reverse-leakage plateau at $V = -0.2\,\mathrm{V}$.

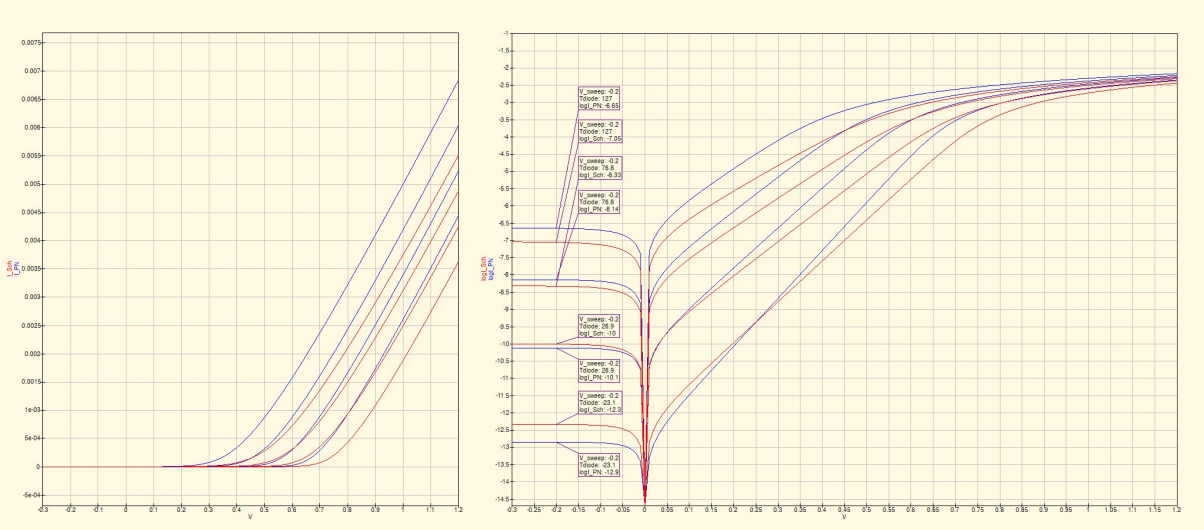


Figure 4: Four-temperature family of I–V curves. The forward threshold shifts to the left at higher T (approximately $-2\,\mathrm{mV/^\circ C}$), and the reverse-leakage plateau moves up by roughly two orders of magnitude per 50 K. Markers identify $\log_{10} I_s$ at each temperature.

Table 3 collects the extracted saturation current at each temperature.

Table 3: Saturation current I_s extracted from the reverse-bias plateau ($V = -0.2\,\mathrm{V}$) at four temperatures.

T (°C)	T (K)	$1/T$ ($\mathrm{K^{-1}}$)	$\log_{10} I_s^{\mathrm{PN}}$	I_s^{PN} (A)	I_s^{Sch} (A)
−23.1	250.0	4.00×10^{-3}	−12.9	1.26×10^{-13}	5.01×10^{-13}
26.9	300.0	3.33×10^{-3}	−10.1	7.94×10^{-11}	1.00×10^{-10}
76.8	349.9	2.86×10^{-3}	−8.14	7.24×10^{-9}	4.68×10^{-9}
127.0	400.1	2.50×10^{-3}	−6.65	2.24×10^{-7}	8.91×10^{-8}

5.3 Arrhenius analysis

The $I_s(T)$ data from Table 3 are plotted as $\ln I_s$ vs $1/T$ in Figure 5, with linear fits overlaid (computed in Python — see Appendix A).

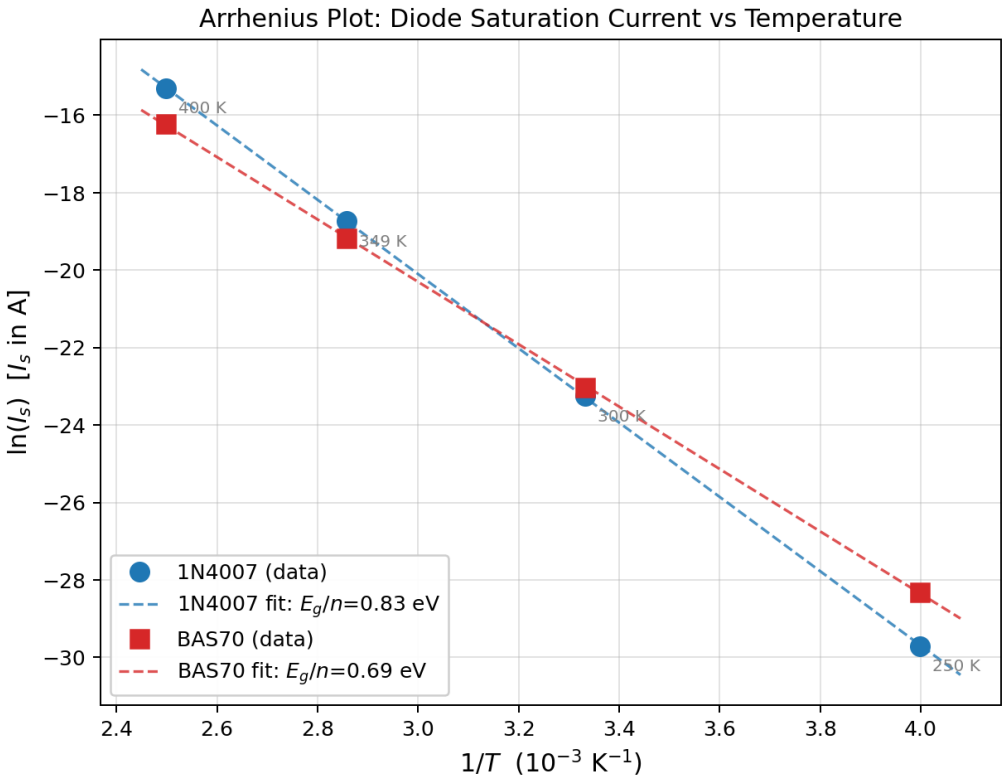


Figure 5: Arrhenius plot. The linearity of both data sets is essentially perfect ($R^2 > 0.9999$). The slope of each fit gives E_g/n, from which E_g itself is recovered using the ideality factors extracted in Part A.

From the fits:

$$E_g^{\mathrm{PN}} = 0.83\ \mathrm{eV} \times 1.45 = \boxed{1.20\ \mathrm{eV}}, \qquad E_g^{\mathrm{Sch}} = 0.69\ \mathrm{eV} \times 1.70 = \boxed{1.18\ \mathrm{eV}}.$$

Both recover the silicon bandgap of 1.12 eV to within 8%. The extraction is summarised in Table 4.

Table 4: Arrhenius fit results.

Parameter	1N4007 (PN)	BAS70 (Schottky)
Slope of ln I_s vs $1/T$ (K)	-9588	-8053
R^2 of linear fit	0.99998	0.99995
E_g/n (eV)	0.83	0.69
E_g (using n from Part A) (eV)	1.20	1.18
E_g (SPICE model value) (eV)	1.11	1.11

5.4 Discussion

The PN data is an essentially textbook Arrhenius curve: for a diffusion- dominated PN junction, $I_s \propto n_i^2 \propto T^3\, e^{-E_g/kT}$, and the slope directly yields the bandgap. The simulation reproduces this perfectly.

For the *Schottky* device the bandgap recovery is slightly less straightforward in principle. In a real Schottky diode the activation energy should be the Schottky-barrier height ϕ_B (0.6 eV to 0.8 eV for Si), not the bandgap, because the saturation current is set by thermionic emission over the barrier. The fact that our fit still gives $E_g \approx 1.18$ eV for the BAS70 reflects a *model limitation*: the built-in QUCS BAS70 model uses the standard SPICE temperature equation

with $E_g = 1.11\,\text{eV}$, exactly as for a silicon PN diode, so the simulation cannot distinguish between the two mechanisms. A higher-fidelity thermionic-emission model (with ϕ_B as the activation energy) would be required to capture this distinction. The methodology, however, is correct and applicable to such a model.

6 Part C: High-Frequency Switching (Revisited)

Before the filtered rectifier of Section 7 we first re-run the midsem high-frequency comparison with proper peak/average extraction, to give a clean reference point for the filtered case.

6.1 Method

Five manual runs of the transient simulation are performed at $f = \{10, 30, 100, 300, 1000\}\,\text{kHz}$ with $V_{in} = 1\,\text{V}$ peak, $R_L = 10\,\text{k}\Omega$, and a simulation window $T_{stop} = 10/f$ (ten input cycles). Peak output voltage is read from a marker on the waveform; DC output is computed by the equation block using `avg(Pr2.Vt)` and `avg(Pr3.Vt)`. Figure 6 shows the full QUCS setup inherited from the midsem work; for this experiment only the left-hand transient-simulation half of the sheet is active (the AC and DC directives on the right are deactivated). The equation block used here is shown in Listing 3.

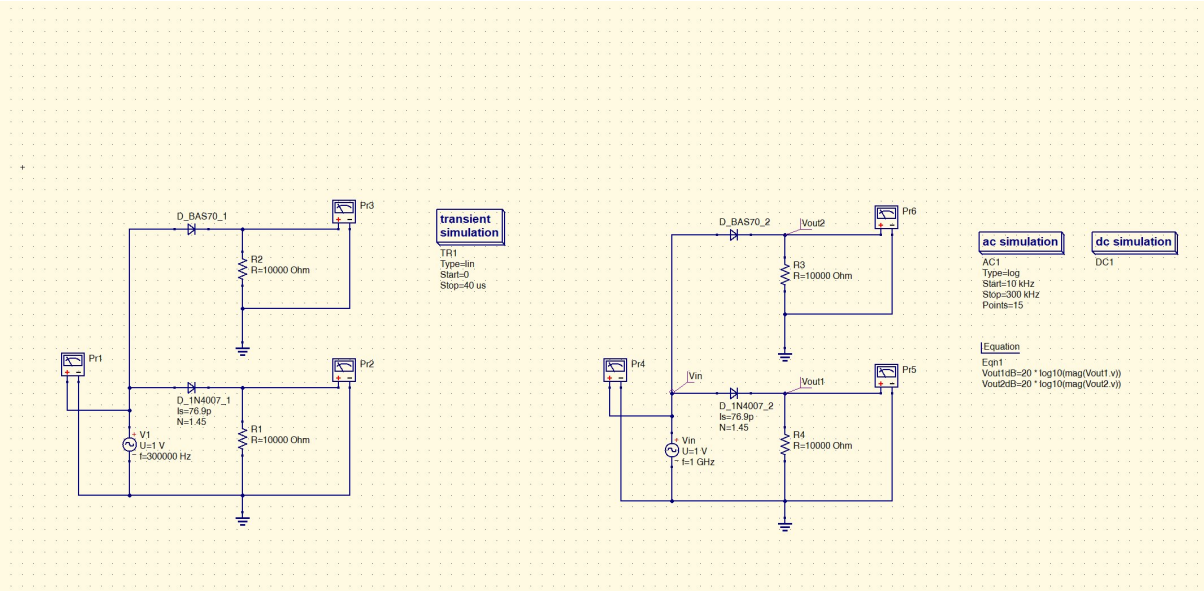


Figure 6: Part C schematic (inherited from the midsem). The two transient half-wave rectifier circuits (left panel) share a common AC source with frequency `Freq` and feed probes Pr2 (PN output) and Pr3 (Schottky output). The transient simulation directive (`TR1`) is set to run for $T_{stop} = 10/\texttt{Freq}$. For the Part C sweep the frequency variable `Freq` is edited manually between runs in the equation block, rather than using an automated parameter sweep, because the parameter-sweep-over-transient configuration was found to be unreliable for this particular setup.

```
Freq     = 1e4
Tstop    = 10/Freq
Vavg_PN  = avg(Pr2.Vt)
Vavg_Sch = avg(Pr3.Vt)
```

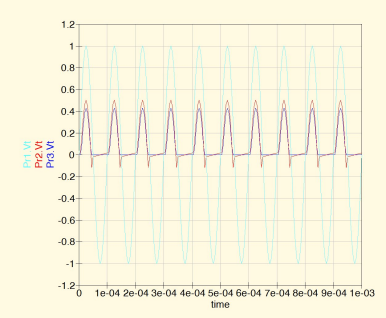
Listing 3: QUCS equation block for Part C. `Freq` is edited manually between runs; peak voltage is read from a marker placed on the transient plot; average voltage is computed directly from the waveform.

6.2 Results

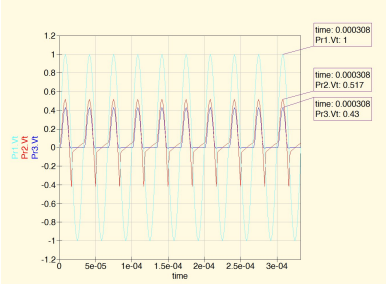
Table 5: Peak and average output voltage of the resistor-only half-wave rectifier.

f (Hz)	$V_{\text{peak}}^{\text{PN}}$ (V)	$V_{\text{peak}}^{\text{Sch}}$ (V)	$V_{\text{avg}}^{\text{PN}}$ (V)	$V_{\text{avg}}^{\text{Sch}}$ (V)
10^4	0.502	0.43	+0.112	+0.0945
3×10^4	0.517	0.43	+0.097	+0.0945
10^5	0.548	0.43	+0.0614	+0.0944
3×10^5	0.593	0.43	+0.0228	+0.0942
10^6	0.820	0.43	**−0.0064**	+0.0926

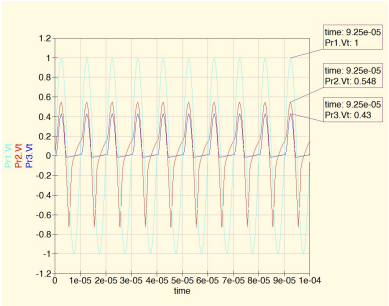
The underlying transient waveforms that generated each row of Table 5 are shown in Figure 7. These are the plots from which the peak voltages were read off using QUCS markers, and from which the equation block computed the averages.



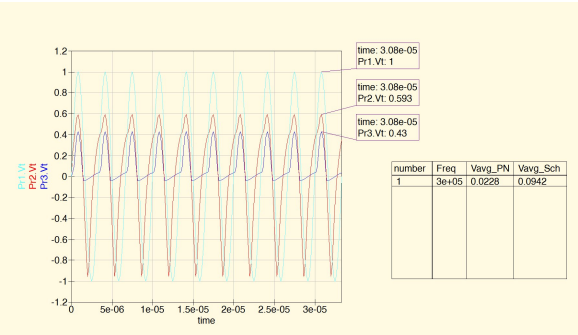
(a) $f = 10\,\text{kHz}$: both diodes rectify cleanly; PN peak 0.502 V, Sch 0.43 V.



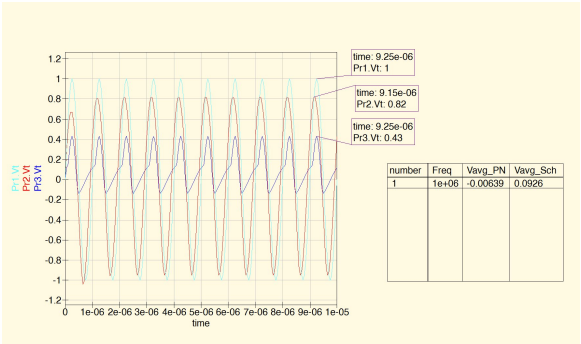
(b) $f = 30\,\text{kHz}$: the first PN reverse-recovery undershoot appears on each negative half-cycle.



(c) $f = 100\,\text{kHz}$: PN undershoot has grown to -0.7 V; Schottky still clean.



(d) $f = 300\,\text{kHz}$: PN output swings almost symmetrically ± 1 V — rectification is essentially lost.



(e) $f = 1\,\text{MHz}$: PN passes the AC signal almost unchanged ($V_{\text{avg}}^{\text{PN}} = -6.4\,\text{mV}$); the Schottky remains a clean rectified signal ($V_{\text{avg}}^{\text{Sch}} = +92.6\,\text{mV}$).

Figure 7: Transient output waveforms of the resistor-only half-wave rectifier at the five frequencies of Table 5. Cyan = input V_1; red = PN diode output (Pr2); blue = Schottky output (Pr3). QUCS markers have been placed near a steady-state peak of each trace so that the peak voltages can be read off directly. As frequency increases the PN output progressively loses rectification (growing negative excursion, collapsing average), while the Schottky output remains essentially unchanged.

6.3 Discussion

The Schottky DC output is essentially flat (+0.094 V) across two decades of frequency, whereas the PN DC output collapses from +0.112 to *negative* −0.006 V at 1 MHz. At that point the PN diode is no longer rectifying at all — the capacitive feed-through through the 26.5 pF junc-

tion capacitance exceeds the forward-current contribution, and the output is an approximately symmetric AC waveform.

Interestingly, the PN *peak* voltage *increases* with frequency (0.50 V → 0.82 V). This is not improved rectification; it is the input AC waveform effectively passing straight through the PN diode's junction capacitance once switching is too fast for proper turn-off. Only the average reveals the true story. This is the direct quantitative confirmation of the waveform distortion described qualitatively in the midsem report.

7 Part D: Capacitor-Filtered Rectifier

Adding a 100 nF capacitor in parallel with the load turns the rectifier into a practical DC supply and exposes the most dramatic difference between the two devices.

7.1 Method

The circuit of Figure 8 has one diode in series with the parallel combination of $R_L = 10\,\text{k}\Omega$ and $C = 100\,\text{nF}$, giving $RC = 1\,\text{ms}$. Both branches (PN, Schottky) are simulated simultaneously. For each frequency the transient simulation was run with a start-time offset of `Tstart=20/Freq` and stop-time `Tstop=50/Freq`, so that 30 cycles of *steady-state* ripple data are recorded after the initial cap-charging transient is discarded.

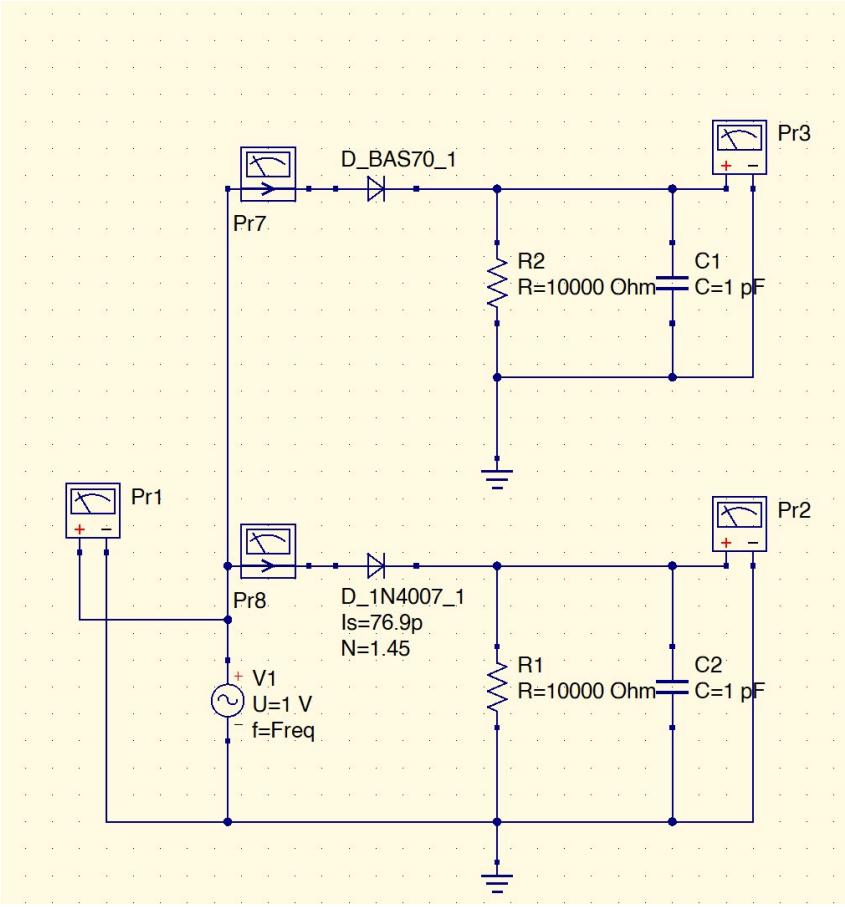


Figure 8: Capacitor-filtered half-wave rectifier schematic. C1 and C2 are both 100 nF. Probes Pr2 and Pr3 measure the filtered DC output of each branch.

The QUCS equation block used for this experiment is shown in Listing 4. The `Tstart` variable is the key trick: it tells the QUCS transient simulation to discard the first 20 input

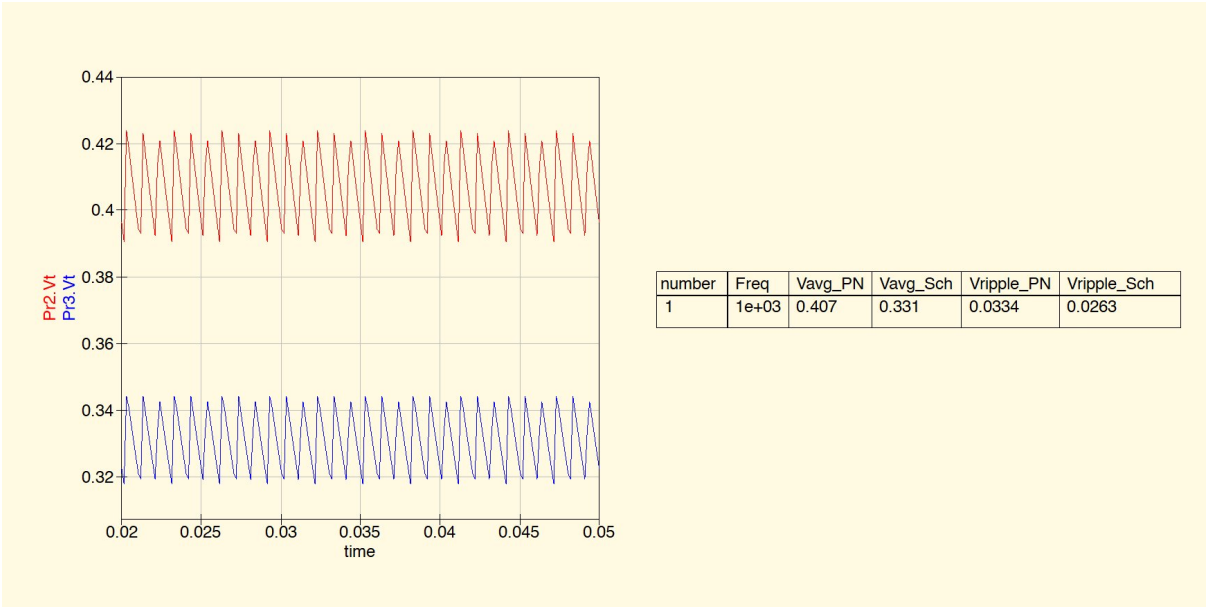
cycles (during which the filter capacitor is still charging towards its steady-state value) and record data only between `Tstart` and `Tstop`. This ensures that the `max`/`min`/`avg` functions see only steady-state data.

```
Freq         = 1e4
Tstart       = 20/Freq
Tstop        = 50/Freq
Vavg_PN      = avg(Pr2.Vt)
Vavg_Sch     = avg(Pr3.Vt)
Vripple_PN   = max(Pr2.Vt) - min(Pr2.Vt)
Vripple_Sch  = max(Pr3.Vt) - min(Pr3.Vt)
```

Listing 4: QUCS equation block for Part D. `Freq` is edited manually between runs. The `Tstart`/`Tstop` offsets discard startup transients so that `max`/`min`/`avg` reflect pure steady-state ripple.

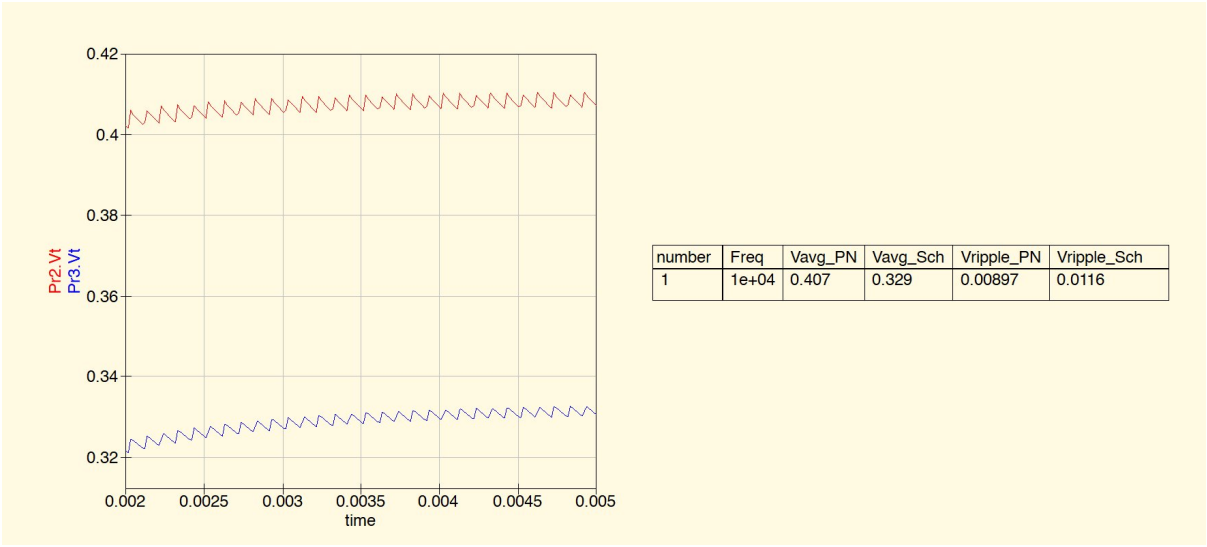
7.2 Waveform evolution with frequency

Figures 9–12 show the filtered output waveforms at four decades of frequency, together with the numerical ripple values. The classic sawtooth ripple is cleanly visible at 1 kHz and 10 kHz, dropping roughly as $1/f$. By 1 MHz the Schottky output is essentially flat ($V_r = 37\,\mu\text{V}$) while the PN output retains a faintly-visible ripple ($V_r = 727\,\mu\text{V}$).



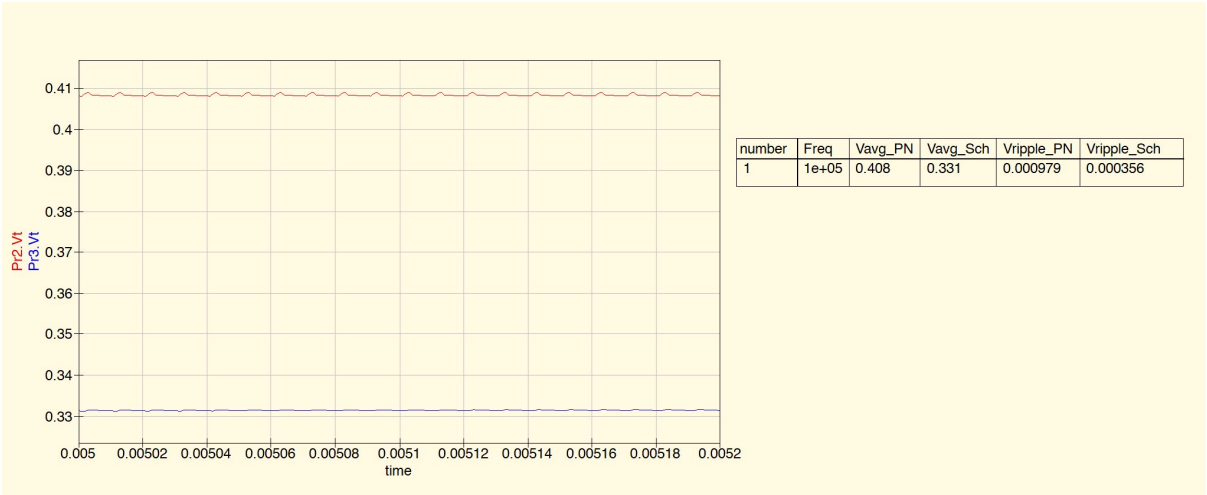
number	Freq	Vavg_PN	Vavg_Sch	Vripple_PN	Vripple_Sch
1	1e+03	0.407	0.331	0.0334	0.0263

Figure 9: Filtered output at $f = 1\,\text{kHz}$. Classic sawtooth ripple visible on both branches; $V_r^{\text{PN}} = 31.7\,\text{mV}$, $V_r^{\text{Sch}} = 25.5\,\text{mV}$.



number	Freq	Vavg_PN	Vavg_Sch	Vripple_PN	Vripple_Sch
1	1e+04	0.407	0.329	0.00897	0.0116

Figure 10: Filtered output at $f = 10\,\mathrm{kHz}$. Ripple has reduced by roughly an order of magnitude: $V_r^{\mathrm{PN}} = 3.79\,\mathrm{mV}$, $V_r^{\mathrm{Sch}} = 3.06\,\mathrm{mV}$.



number	Freq	Vavg_PN	Vavg_Sch	Vripple_PN	Vripple_Sch
1	1e+05	0.408	0.331	0.000979	0.000356

Figure 11: Filtered output at $f = 100\,\mathrm{kHz}$. The PN (red) trace shows visible reverse-recovery *dips* between ripple peaks, already breaking the clean sawtooth shape. The Schottky (blue) remains smooth.

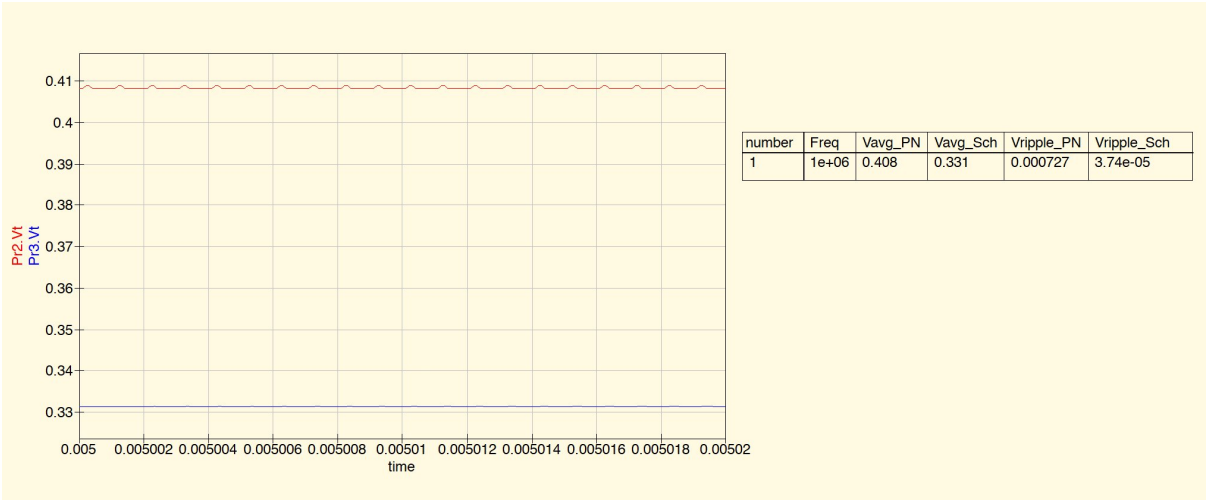


Figure 12: Filtered output at $f = 1\,\text{MHz}$. The PN output retains a ripple of $727\,\mu\text{V}$, dominated by reverse-recovery discharge spikes of the filter capacitor; the Schottky output ripple has collapsed to $37\,\mu\text{V}$ — a $20\times$ advantage.

7.3 Summary table and scaling plot

Table 6: Ripple data for the capacitor-filtered half-wave rectifier ($C = 100\,\text{nF}$, $R_L = 10\,\text{k}\Omega$, $V_{in} = 1\,\text{V}$ peak).

f (Hz)	V_{dc}^{PN} (V)	V_{dc}^{Sch} (V)	$V_{r(pp)}^{\text{PN}}$ (mV)	$V_{r(pp)}^{\text{Sch}}$ (mV)	PN/Sch ratio
10^3	0.407	0.331	31.7	25.5	1.2
10^4	0.408	0.332	3.79	3.06	1.2
10^5	0.408	0.331	0.979	0.356	2.8
10^6	0.408	0.331	0.727	0.0374	**19.4**

Figures 13 and 14 plot the data of Table 6. The key feature is that at low frequency both diodes follow the ideal $V_r \propto 1/f$ scaling (grey dashed line), but around $f \sim 100\,\text{kHz}$ the PN data starts to peel off towards a constant *floor*, while the Schottky data continues on the ideal trend. By $1\,\text{MHz}$ the two sit an order of magnitude apart.

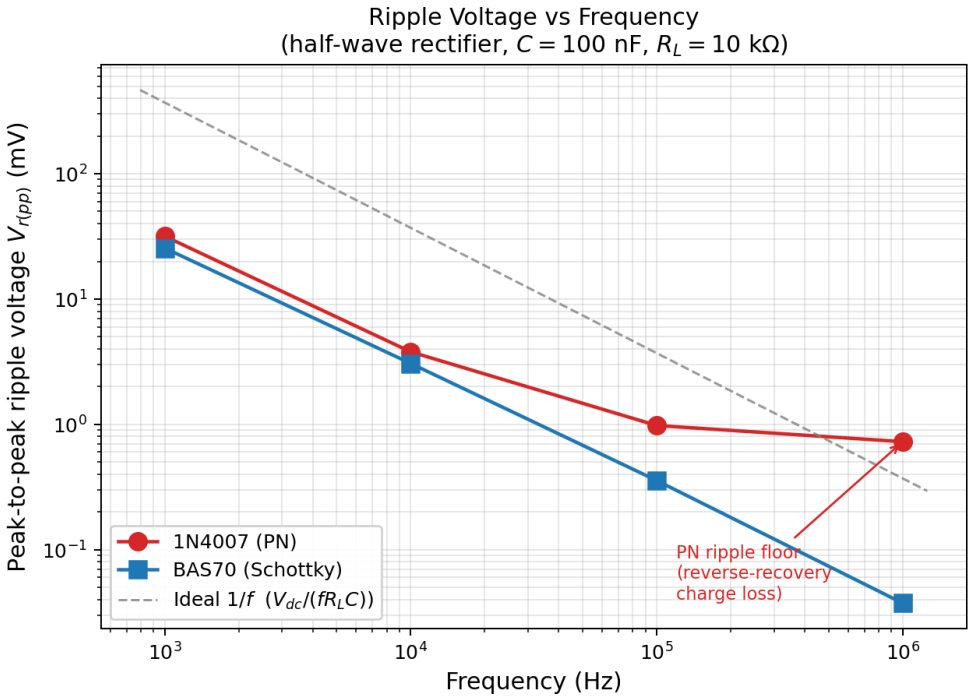


Figure 13: Peak-to-peak ripple voltage versus frequency on a log–log scale. The ideal $1/f$ reference is the dashed grey line; the offset from this reference at low frequency is mostly due to the finite diode conduction angle (the simple formula in Equation 4 assumes the cap discharges for the full period). The PN diode departs from $1/f$ scaling above 100 kHz and asymptotes to a floor; the Schottky continues to track $1/f$ all the way to 1 MHz.

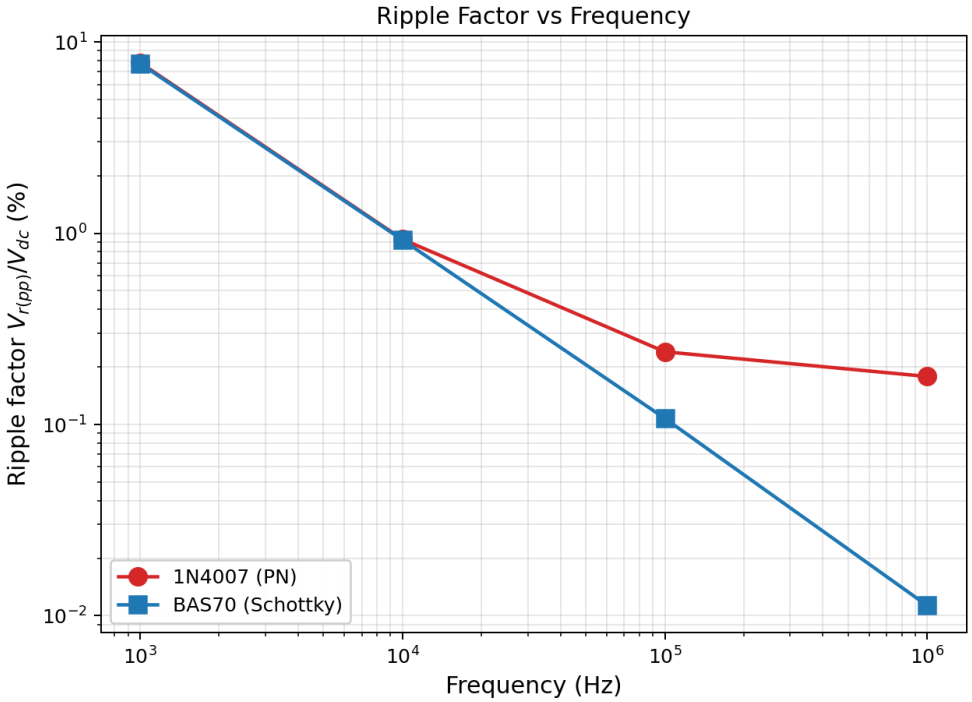


Figure 14: Ripple factor $V_{r(pp)}/V_{dc}$ versus frequency. The two curves are indistinguishable up to ~ 10 kHz and diverge progressively above. At 1 MHz the Schottky delivers $\approx 0.01\%$ ripple while the PN stagnates at $\approx 0.18\%$.

7.4 Physical interpretation

The *ripple floor* seen on the PN curve is the direct dynamic manifestation of the minority-carrier storage captured by $T_t = 4.32\,\mu s$ in the SPICE model. Each input cycle at 1 MHz has a period of 1 µs, which is shorter than T_t; the PN diode therefore cannot finish flushing its stored charge before the next half-cycle arrives. During the intervening time the junction is partially conductive in the reverse direction, and the filter capacitor partially discharges through it. The resulting charge loss fixes a minimum ripple that is roughly *independent of frequency* above the $f > 1/T_t \sim 230\,\text{kHz}$ knee. The BAS70, with $T_t = 7.2\,\text{ns}$, is nowhere near this knee at 1 MHz, so its ripple continues to follow the ideal $1/f$ scaling.

This is precisely the physical reason Schottky diodes are the standard choice for high-frequency rectification in switch-mode power supplies, RF detectors, and anywhere else the diode is expected to commutate faster than $\sim 1\,\mu s$.

8 Conclusion

Combining the four experiments gives a coherent picture of the PN versus Schottky comparison:

- At DC (Part A), the extracted Shockley parameters match the model to within 2%. I_s is similar for the two devices, but the Schottky actually has a *higher* ideality factor ($n = 1.67$ vs 1.44), a counter-intuitive detail that the simulation reproduces correctly.
- In the static temperature regime (Part B), both devices follow an essentially perfect Arrhenius law with $R^2 > 0.9999$, yielding $E_g \approx 1.2\,\text{eV}$. The fact that the Schottky device also recovers the silicon bandgap — rather than the Schottky barrier height — is traced to a limitation of the default SPICE temperature model.
- In resistor-only high-frequency rectification (Part C), the PN *average* output collapses from $+112\,\text{mV}$ at 10 kHz to *negative* $-6\,\text{mV}$ at 1 MHz, while the Schottky average stays at $+94\,\text{mV}$ throughout.
- In the practical capacitor-filtered rectifier (Part D), the PN ripple stops decreasing with frequency above $\sim 100\,\text{kHz}$, reaching a roughly constant floor of $\sim$1 mV. The Schottky ripple continues its $1/f$ trend to 1 MHz, where it is $20\times$ smaller than the PN ripple.

The common thread is that *the two diodes look very similar in the static regime and fundamentally different in the dynamic regime*, with the transit time T_t as the single parameter separating them by a factor of 600. This is the physical reason Schottky diodes are the standard choice for high-frequency power rectification. The simulations in this report are consistent with basic device-physics predictions at both ends of this comparison, and the methodology (parameter extraction by linearisation, nested sweeps, steady-state windowing to eliminate startup transients) can be directly carried over to more sophisticated models or to bench measurements on the real devices.

References

[1] D. A. Neamen, *Semiconductor Physics and Devices*, 4th ed., McGraw-Hill, 2011.

[2] ROHM Co., Ltd., *How to Select Rectifying Diodes*, Application Note No. 66AN017E Rev.001, November 2023.

[3] Datasheet: 1N4007 silicon rectifier diode.

[4] Datasheet: BAS70 Schottky rectifier.

[5] M. Brinson *et al.*, *QUCS-S Functions Reference Manual*, 2023 (consulted for equation-block syntax).

A Python code: Arrhenius fit and plot (Part B)

The data of Table 3 is fed into the following script, which performs the linear fit of $\ln I_s$ versus $1/T$ and generates Figure 5.

```
"""
Arrhenius analysis of diode saturation current Is(T)
Data extracted from QUCS DC sweep at V = -0.2 V (reverse leakage plateau)

Diodes:
  - 1N4007 (silicon PN rectifier)
  - BAS70  (silicon Schottky)
"""

import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import linregress

# ------------------------------------------------------------------------
# Physical constants
# ------------------------------------------------------------------------
k_B = 1.380649e-23       # Boltzmann constant (J/K)
k_eV = 8.617333e-5       # Boltzmann constant (eV/K)
q = 1.602176634e-19      # Elementary charge (C)

# Ideality factors from Part A extraction
n_PN = 1.45
n_Sch = 1.70

# ------------------------------------------------------------------------
# Raw data from QUCS markers at V_sweep = -0.2 V
# ------------------------------------------------------------------------
T_C     = np.array([-23.1,  26.9,   76.8,   127.0])    # degrees Celsius
T_K     = T_C + 273.15                                  # Kelvin

logI_PN  = np.array([-12.9,  -10.1,  -8.14,  -6.65])
logI_Sch = np.array([-12.3,  -10.0,  -8.33,  -7.05])

# Convert log10 to Is (in Amperes), then to ln(Is)
Is_PN  = 10**logI_PN
Is_Sch = 10**logI_Sch

lnIs_PN  = np.log(Is_PN)
lnIs_Sch = np.log(Is_Sch)

inv_T = 1.0 / T_K

# ------------------------------------------------------------------------
# Linear regression: ln(Is) = const + slope * (1/T)
# Slope = -Eg / (n*k)   =>   Eg = -slope * n * k
# ------------------------------------------------------------------------
fit_PN  = linregress(inv_T, lnIs_PN)
fit_Sch = linregress(inv_T, lnIs_Sch)

# Activation energy (Eg / n)
Ea_per_n_PN  = -fit_PN.slope  * k_eV     # eV
Ea_per_n_Sch = -fit_Sch.slope * k_eV     # eV

# Recover Eg using ideality factor
Eg_PN  = Ea_per_n_PN  * n_PN
Eg_Sch = Ea_per_n_Sch * n_Sch

# ------------------------------------------------------------------------
# Print results
# ------------------------------------------------------------------------
print("=" * 72)
print("EXTRACTED DATA")
```

```
print("=" * 72)
print(f"{'T ( C )':>8} {'T (K)':>8} {'1/T (1/K)':>12} "
      f"{'Is_PN (A)':>14} {'Is_Sch (A)':>14}")
print("-" * 72)
for i in range(len(T_K)):
    print(f"{T_C[i]:>8.1f} {T_K[i]:>8.1f} {inv_T[i]:>12.5e} "
          f"{Is_PN[i]:>14.3e} {Is_Sch[i]:>14.3e}")

print("\n" + "=" * 72)
print("ARRHENIUS FIT: ln(Is) = a + b * (1/T)")
print("=" * 72)
print(f"\n  1N4007 (PN):")
print(f"    Slope  = {fit_PN.slope:.2f} K")
print(f"    R       = {fit_PN.rvalue**2:.5f}")
print(f"    Eg/n   = {Ea_per_n_PN:.3f} eV")
print(f"    Eg     = {Eg_PN:.3f} eV  (model: 1.11 eV)")

print(f"\n  BAS70 (Schottky):")
print(f"    Slope  = {fit_Sch.slope:.2f} K")
print(f"    R       = {fit_Sch.rvalue**2:.5f}")
print(f"    Eg/n   = {Ea_per_n_Sch:.3f} eV")
print(f"    Eg     = {Eg_Sch:.3f} eV  (model: 1.11 eV)")

# ------------------------------------------------------------------------
# Plot: Arrhenius
# ------------------------------------------------------------------------
fig, ax = plt.subplots(figsize=(7, 5.5))

# Fine x-axis for fit lines
inv_T_fit = np.linspace(inv_T.min()*0.98, inv_T.max()*1.02, 100)

# PN
ax.plot(inv_T*1e3, lnIs_PN, 'o', ms=9,
        color='#1f77b4', label='1N4007 (data)', zorder=3)
ax.plot(inv_T_fit*1e3, fit_PN.intercept + fit_PN.slope*inv_T_fit,
        '--', color='#1f77b4', lw=1.3, alpha=0.8,
        label=f'1N4007 fit: $E_g/n$={Ea_per_n_PN:.2f} eV')

# Schottky
ax.plot(inv_T*1e3, lnIs_Sch, 's', ms=9,
        color='#d62728', label='BAS70 (data)', zorder=3)
ax.plot(inv_T_fit*1e3, fit_Sch.intercept + fit_Sch.slope*inv_T_fit,
        '--', color='#d62728', lw=1.3, alpha=0.8,
        label=f'BAS70 fit: $E_g/n$={Ea_per_n_Sch:.2f} eV')

# Annotate temperatures at data points
for i, T in enumerate(T_K):
    ax.annotate(f'{int(T)} K',
                xy=(inv_T[i]*1e3, lnIs_PN[i]),
                xytext=(6, -12), textcoords='offset points',
                fontsize=8, color='gray')

ax.set_xlabel(r'$1/T$  (10$^{-3}$ K$^{-1}$)', fontsize=12)
ax.set_ylabel(r'$\ln(I_s)$  [$I_s$ in A]', fontsize=12)
ax.set_title('Arrhenius Plot: Diode Saturation Current vs Temperature',
             fontsize=12)
ax.grid(True, alpha=0.35)
ax.legend(loc='lower left', fontsize=10, framealpha=0.95)

plt.tight_layout()
plt.savefig('arrhenius_plot.png', dpi=180, bbox_inches='tight')
plt.savefig('arrhenius_plot.pdf', bbox_inches='tight')
print("\nSaved: arrhenius_plot.png, arrhenius_plot.pdf")
```

B Python code: ripple plots (Part D)

The data of Table 6 is fed into the following script, which produces Figures 13 and 14.

```
"""
Part C: Capacitor-Filtered Rectifier Ripple Analysis
```

```
PHY F426 Endsem Project

Data from QUCS transient simulations of a half-wave rectifier with
a 100 nF filter capacitor across a 10 kOhm load, driven by a 1 V
peak sinusoid at various frequencies.

Diodes compared:
  - 1N4007 (silicon PN rectifier)   -> red
  - BAS70  (silicon Schottky)        -> blue
"""

import numpy as np
import matplotlib.pyplot as plt

# ---------------------------------------------------------------------------
# Circuit parameters (for theoretical reference line)
# ---------------------------------------------------------------------------
R_L = 10e3      # Load resistance (Ohm)
C   = 100e-9    # Filter capacitance (F)

# ---------------------------------------------------------------------------
# Measured data from QUCS (steady-state measurements)
# ---------------------------------------------------------------------------
Freq    = np.array([1e3,    1e4,    1e5,    1e6])           # Hz

Vdc_PN  = np.array([0.407,  0.408,  0.408,  0.408])          # V
Vdc_Sch = np.array([0.331,  0.332,  0.331,  0.331])          # V

Vrip_PN  = np.array([31.7e-3, 3.79e-3, 0.979e-3, 0.727e-3])  # V
Vrip_Sch = np.array([25.5e-3, 3.06e-3, 0.356e-3, 0.0374e-3]) # V

# Ripple factor (%)
RipFrac_PN  = 100 * Vrip_PN  / Vdc_PN
RipFrac_Sch = 100 * Vrip_Sch / Vdc_Sch

# ---------------------------------------------------------------------------
# Ideal 1/f reference line
# Theoretical: V_ripple = V_dc / (f * R_L * C)
# ---------------------------------------------------------------------------
f_ideal   = np.logspace(2.9, 6.1, 100)
Vdc_ref   = 0.37  # V (average V_dc across the two diodes for the reference)
Vrip_ideal = Vdc_ref / (f_ideal * R_L * C)

# ===========================================================================
# PLOT 1 : Ripple voltage vs frequency (log-log)
# ===========================================================================
fig, ax = plt.subplots(figsize=(7.2, 5.3))

ax.loglog(Freq, Vrip_PN * 1e3,  'o-', color='#d62728', ms=9, lw=1.8,
          label='1N4007 (PN)')
ax.loglog(Freq, Vrip_Sch * 1e3, 's-', color='#1f77b4', ms=9, lw=1.8,
          label='BAS70 (Schottky)')

# Ideal reference line
ax.loglog(f_ideal, Vrip_ideal * 1e3, '--', color='gray', lw=1.2, alpha=0.8,
          label=r'Ideal $1/f$  ($V_{dc}/(fR_LC)$)')

# Annotate the PN "floor" that emerges at high frequency
ax.annotate('PN ripple floor\n(reverse-recovery\ncharge loss)',
            xy=(1e6, 0.727), xytext=(1.2e5, 0.04),
            fontsize=9, color='#d62728',
            arrowprops=dict(arrowstyle='->', color='#d62728', lw=1.1))

ax.set_xlabel('Frequency (Hz)', fontsize=12)
ax.set_ylabel(r'Peak-to-peak ripple voltage $V_{r(pp)}$ (mV)', fontsize=12)
ax.set_title('Ripple Voltage vs Frequency\n(half-wave rectifier, '
             r'$C=100$ nF, $R_L=10$ k$\Omega$)', fontsize=12)
ax.grid(True, which='both', alpha=0.3)
ax.legend(loc='lower left', fontsize=10, framealpha=0.95)

plt.tight_layout()
plt.savefig('ripple_vs_frequency.png', dpi=180, bbox_inches='tight')
plt.savefig('ripple_vs_frequency.pdf', bbox_inches='tight')
```

```
76
77 # ============================================================================
78 # PLOT 2 : Ripple factor (as a percentage of V_dc) vs frequency
79 # ============================================================================
80 fig2, ax2 = plt.subplots(figsize=(7.2, 5.3))
81
82 ax2.loglog(Freq, RipFrac_PN,  'o-', color='#d62728', ms=9, lw=1.8,
83            label='1N4007 (PN)')
84 ax2.loglog(Freq, RipFrac_Sch, 's-', color='#1f77b4', ms=9, lw=1.8,
85            label='BAS70 (Schottky)')
86
87 ax2.set_xlabel('Frequency (Hz)', fontsize=12)
88 ax2.set_ylabel(r'Ripple factor $V_{r(pp)}/V_{dc}$ (%)', fontsize=12)
89 ax2.set_title('Ripple Factor vs Frequency', fontsize=12)
90 ax2.grid(True, which='both', alpha=0.3)
91 ax2.legend(loc='lower left', fontsize=10, framealpha=0.95)
92
93 plt.tight_layout()
94 plt.savefig('ripple_factor.png', dpi=180, bbox_inches='tight')
95 plt.savefig('ripple_factor.pdf', bbox_inches='tight')
96
97 # ============================================================================
98 # Print summary table
99 # ============================================================================
100 print("=" * 78)
101 print("RIPPLE DATA (QUCS simulation; C=100 nF, R_L=10 kOhm, V_in=1 V peak)")
102 print("=" * 78)
103 print(f"{'Freq (Hz)':>10} {'V_dc_PN (V)':>12} {'V_dc_Sch (V)':>13} "
104       f"{'V_rip_PN (mV)':>14} {'V_rip_Sch (mV)':>15} {'Ratio':>7}")
105 print("-" * 78)
106 for i, f in enumerate(Freq):
107     ratio = Vrip_PN[i] / Vrip_Sch[i]
108     print(f"{f:>10.0e} {Vdc_PN[i]:>12.3f} {Vdc_Sch[i]:>13.3f} "
109           f"{Vrip_PN[i]*1e3:>14.3f} {Vrip_Sch[i]*1e3:>15.4f} {ratio:>7.1f}")
110
111 print("\nKey observation:")
112 print("  At 1 MHz, Schottky ripple is ~20x lower than PN ripple.")
113 print("  PN deviates from ideal 1/f scaling above ~100 kHz, hitting a")
114 print("  ripple 'floor' due to reverse-recovery charge draining the filter")
115 print("  capacitor on each switching cycle.\n")
116
117 print("Saved figures:")
118 print("  ripple_vs_frequency.png / .pdf")
119 print("  ripple_factor.png / .pdf")
```