Contents

Chapter 1

Introduction: The World Needs Green Electronics

1.1 The Energy Crisis

An increasing, industrialized world population is putting undue pressures on our world's delicate ecosystem and on our natural resources. In the United States, 85% of energy generated is from fossil fuels (DOE Annual Energy Outlook). While there are large reserves, that are increasing as people get better at exploration and recovery, the supply is finite. We will ultimately run out of fossil fuels.

Even if our supply of fossil fuels was infinite, burning fossil fuels generates greenhouse gasses like CO_2 that accumulate in the atmosphere leading to climate change. In recent years, atmospheric CO_2 has been increasing at a rate of about 20ppm/decade (5% per decade) (NOAA data). We need to burn less fossil fuels, or sequester the carbon generated by burning these fuels, or both — or suffer the consequences of climate change.

To deal both with the limited supplies of fossil fuels and their environmental impact, we need to develop technologies for economic renewable energy sources. Wind and solar power are two of the most promising candidates — both depend strongly on green electronics. Photovoltaic controllers, generator controllers, and inverters are important components in efficiently getting energy from solar cells and windmills to the grid — and ultimately to the end user.

Conservation plays a large role in reducing our consumption of fossil fuels and our generation of greenhouse gasses. Replacing inefficient incandescent lamps with highly-efficient LED lamps and using intelligent control to have lamps only illuminate areas where people are looking can dramatically reduce the energy consumption due to lighting. Data centers, which consume about 2% of the electric power in the US, use power supplies that are often only 70-80% efficient and processors that spend the bulk of their power on overhead. More efficient power supply and processor design woudl dramatically reduce

electricity consumption. There are many more examples where efficient power electronics and intelligent control can greatly reduce consumption.

1.2 Green Electronics: Part of the Solution

A key technology for both sustainable energy generation and conservation is the combination of intelligent control with power electronics — the brains and brawn of energy systems. The power electronics provides the brawn — doing the heavy work of converting energy from one form to another. This includes electrical conversion — as in a photovoltaic system where the DC output voltage from a string of solar panels is converted to AC power at a different voltage to drive the grid — and mechanical conversion where the system includes a motor or generator — as in a wind farm or an electric vehicle.

Intelligent computer control provides the brains of the system, optimizing operation to improve efficiency. A photovoltaic controller searches to find the optimum operating point for each solar cell in a string — optimizing efficiency. A controller electrically switches the windings of a brushless generator on a wind mill, achieving greater efficiency than with an alternator or a generator with a commutator. A lighting controller uses cameras to sense the location and gaze of people in a building to turn on lights only where they are needed.

1.2.1 Example 1: Photovoltaic Generation

As an example of a *green-electronics system*, consider the grid-connected photovoltaic system of Figure 1.1. A *photovoltaic (PV) array* of solar panels converts solar energy to electricity. With a typical configuration of 10 40V panels in each series *string*, the output of this array is 400V DC. The power depends on the number of strings and for a residential system is typically in the range of 4-10kW. So the DC current out of the PV array is in the rane of 10-25A.

A *PV Controller* and *Inverter* connects the PV array to a bank of batteries (that operate at 48V DC) and to the electric grid (240V AC). The PV controller converts electrical energy from one form to another. For a 4kW system, it converts 10A at 400V DC to 16.7A at 240V AC to drive the grid, or to 83.3A at 48V DC to charge the batteries.

The controller also sets the operating point of the PV array to maximize the energy produced. As shown in Figure 1.2, a solar panel acts as a *current source* — producing a nearly constant current up until a maximum voltage where the current falls quickly. To get maximum power out of the panel, it must be operated at the *knee* of this curve — at the voltage where the current just begins to drop. The PV controller adjusts its output impedance to operate the array at this point.

Because of non-uniformity of illumination and manufacturing not all panels may have the same opimal operating point. In a simple system like that shown in Figure 1.1 the controller must set a single operating point that is a compromise — it gives the highest power over all but may not be the best for each panel.

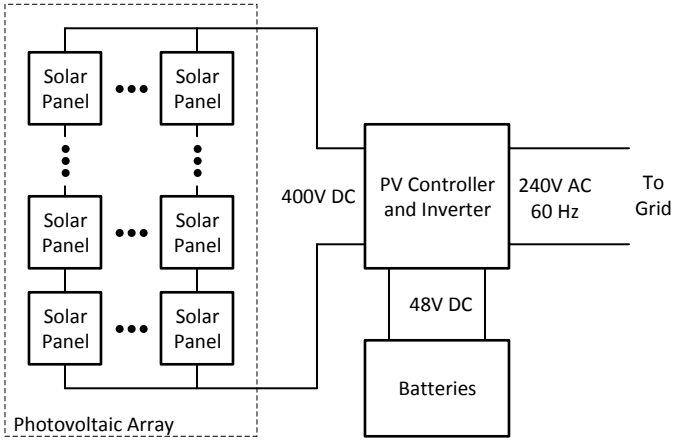
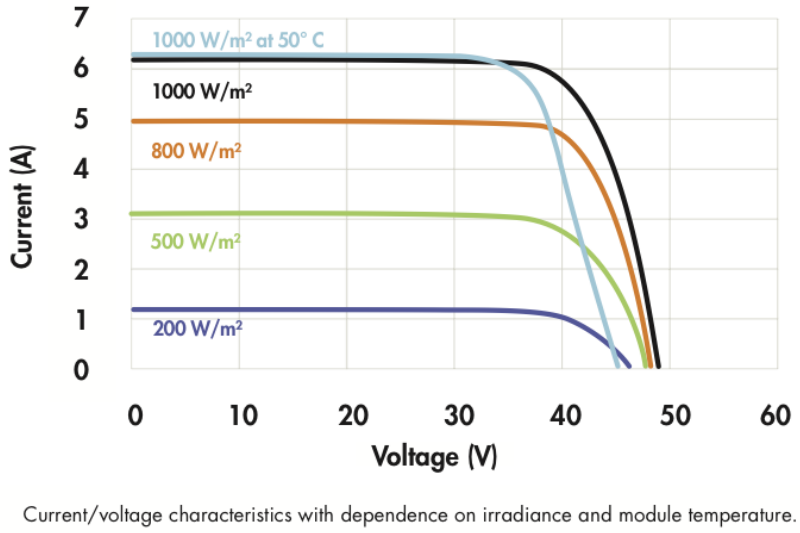


Figure 1.1: A photovoltaic system. The PV controller converts between 400V DC (string of panels), 48V DC (batteries), and 240V AC power and sets the operating point for the PV array to maximize efficiency.



Current/voltage characteristics with dependence on irradiance and module temperature.

Figure 1.2: Current/voltage (IV) curve of a SunPower E19/240 solar panel. Optimum power is produced when the panel is operated at the *knee* of the curve.

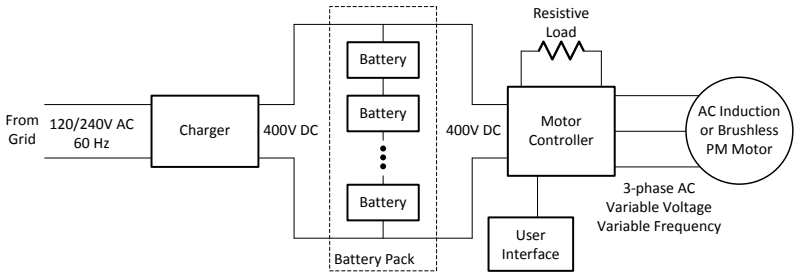


Figure 1.3: Block diagram of an electric car. Energy is stored in a central battery pack that can be charged from the grid when the car is parked. A controller can transfer energy in either direction between the battery pack and an electric motor.

More sophisticated PV controllers monitor the voltage and current across each panel independently and by bypassing current around some panels (using a switched-inductor shunt) operate every panel at its peak power point.

In a grid-connected system that includes battery storage, an intelligent controller decides when to charge the batteries to make best use of *time-of-day* power rates. It will charge the batteries during off-peak times when electricity is substantially cheaper than during peak times[1]. Not only does this optimize economics, it also optimizes efficiency because the *peaking plants* used by the electric utility to provide power at peak times are substantially less efficient than the plants used to provide base capacity.

The controller will also tailor the charging profile to maximize efficiency and battery life. A sophisticated controller will monitor the voltage of individual battery cells and balance the cells to optimize operation.

We will take a closer look at aspects of this model PV system in the labs that make up the bulk of this course. In Chapter 5 we investigate solar panels and build a photovoltaic controller for a single solar panel and develop an algorithm for operating at the peak power point. We examine the properties of batteries and build a battery charger in Chapter 4.

1.2.2 Example 2: An Electric Car

A second example of a green electronics system is the electric car of Figure 1.3. The electronics here charge and manage the batteries and control the motor that provides vehicle propulsion.

When the car is parked, a *charger* converts 120V or 240V AC power to 400V DC power to charge the battery pack. While individual battery cells range from

[1]Under the California E-7 Tariff, the peak rate is nearly four times the off-peak rate.

2V (lead-acid) to 3.7V (Lithium ion), hundreds of these cells are connected in series to give an efficient operating voltage of 400V DC. The charger must carefully manage voltage, current, and temperature during the charging process to maximize efficiency and battery life.

The motor controller converts the 400V DC voltage to a three-phase AC voltage to drive the motor. The frequency of the motor drive depends on the motor speed. The effective voltage of the motor drive is varied using *pulse-width modulation* to control the motor current which determines the torque produced by the motor.

The motor controller employs a *control law* to decide how much current, and hence torque, to apply at any given time. The decision depends on throttle position, vehicle speed, and battery state. The decision made is a compromise between drivability, energy efficiency, and battery longevity.

To recycle energy when slowing the car, the motor controller employs *regenerative braking* by operating the motor as a generator. In this case, the kinetic energy of the vehicle is converted to electrical energy that is used to charge the battery. During this process, the motor controller acts as a generator controller — sequencing the motor phases to optimize generator efficiency and controlling the battery charging process. If regenerative braking occurs when the batteries are fully charged the excess energy is disspated as thermal energy in a resistive load.

In all modes of operation — whether accelerating or braking — the battery pack must be carefully managed to balance cell voltages, and avoid excessive currents and temperatures. Lithium Ion cells are particularly sensitive to proper battery management.

In a hybrid vehicle where a heat engine can be turned on to provide propulsion and charge batteries the controller has an additional degree of freedom. The controller must decide when to turn on the heat engine, how to divide the torque load between the heat engine and the electric motor, and how much energy to divert to charging batteries when the heat engine is running. Making optimal power management decisions may involve predicting the future route of the car — for example, whether it is likely to arrive at a charging station soon, or whether it is likely to be doing up or down a large hill.

A user interface is provided to that the driver can understand how their control actions result in energy movement within the system. A good user interface can train the driver to drive more efficiently. Of course one must be careful with the user interface to avoid distracting the driver[2]

We will explore topics related to the electric vehicle example in more depth during several of the labs in this course. The motor controller lab (Chapter 3) involves controlling a simple DC permanent magnet motor including developing controllers for speed and torque and implementing regenerative braking. The battery charger lab of Chapter 4 introduces some of the issues of battery management.

[2]In the long run the computer will drive the car, resulting in more efficient operation.

1.3 The World Needs More Green Engineers

To realize *green electronic* systems like efficient photovoltaic controllers, wind mill controllers, energy converters, electric- and hybrid-vehicle drives, etc... requires engineers who understand power electronics and computer control. There is a shortage of such talented individuals.

The goal of this course is to teach the basics of these two critical disciplines as an introductory engineering course. The course teaches *engineering thinking* using green electronic examples. In doing so, it conveys engineering fundamentals — learning the process of specification, design, and analysis and learning techniques like modeling and simulation. At the same time, it gives a working knowledge of simple power electronics and microprocessor-based control.

1.3.1 What you will learn

At the completion of the course, a student will have a basic understanding of the following fundamental engineering concepts:

Engineering Process Understand the basic development process that starts with a *specification*, proceeds to a *design*, and which includes *analysis* to understand the design and inform design choices.

Modeling Given a physical device, build an abstract model that captures the relevant behavior of that device — we will build models of many devices including batteries and motors.

Simulation Understand and verify the operation of an engineering system by building a mathematical model of the system and *simulating* its operation using a computer program.

Optimization Vary the parameters of a design to optimize a metric, like efficiency.

Trade-offs Understand the principle of a Pareto-optimal frontier and how to choose a design point when one metric (e.g., efficiency) must be sacrified to improve another metric (e.g., cost).

The student will also understand the following principles of electrical engineering and power electronics:

Basic Circuit Theory Be able to calculate the DC and AC response of simple L, R, C circuits with switches.

Measurement Circuits for measuring current and voltage.

Pulse-Width Modulation (PWM) Efficiently varying the power level of a system by modulating the pulse-width of a fixed-frequency signal.

Periodic Steady-State Analysis Compute the response of a PWM system.

Buck and Boost Converter Topologies Basic circuit topologies for efficiently converting voltages downward (buck) and upward (boost) without using transformers.

Characteristics of Energy Storage Components Critical parameters of power inductors and capacitors.

Characteristics of Semiconductor Switches Critical parameters of modern power MOSFETs, diodes, and IGBTs.

Motor/Generator Basics The basics of electromechanical energy conversion.

Battery Basics The basics of electrochemical energy storage.

And the following basics of microcomputer-based control:

Feedback Control The principle of feedback control and the analysis of stability.

Interrupt-Driven Software The organization of a software system that is triggered by events.

Digital Number Representation Performing calculations using limited-precision, fixed-point binary numbers.

Digital Control Building feedback control systems using computer control.

Layered Control Building hierarchical control systems.

1.3.2 Learn By Doing

This course is based on the principle of *just-in-time*, or *on-demand* learning. Rather than drowning you in dry theory, we present you with a series of design and lab projects that require you to learn material to complete the project. The motivation of completing the design and getting the lab to work will encourage you to learn the required theory. The theory will also make a lot more sense when you see it put to practice in the laboratory exercises.

In these notes, most of the theory has been relegated to appendices that you can refer to to learn, or refresh you knowledge of, various topics. The main-line chapters are centered around the four lab/design projects.

1.3.3 Prerequisites

The intent is for this course to be taken by freshmen. Thus, the course assumes only that you know basic high-school mathematics including algebra and single variable integral and differential calculus. Some simple programming skills are also useful.

1.4 Bibliographic Notes

DOE Annual Energy Outlook

Chapter 2

Build an Energy Meter

If we are going to design electronics to save power and energy, we first need to be able to measure these quantities. To that end, this chapter describes a laboratory where we will build a meter to measure power and energy. We will use this meter to measure the efficiency of green electronic devices we will build in future labs.

In the course of doing this lab, you will learn:

Electrical Quantities: A basic understanding of electrical quantities and units (voltage, current, power, and energy).

Circuit Theory: How do design simple electrical circuits with resistors and operational amplifiers including voltage dividers, voltage followers, and differential amplifiers.

Real-Time Programming: How to program a simple embedded system that operates using real-time clock interrupts and A/D converter interrupts.

Embedded System Programming: How to program a basic embedded system using fixed-point signed and unsigned integer arithmetic and basic input-output devices.

User Interface Programming: How to program a basic user interface with display and user input.

Basic Lab Skills: How to use a lab power supply and voltmeter to calibrate your energy meter. How to wire up and debug circuits.

2.1 Hardware

Our power and energy meter connects between a power source and a load as shown in Figure 2.1. The meter has three terminals, NS, NL, and P. The meter senses the load voltage V_L by measuring the voltage between the negative load

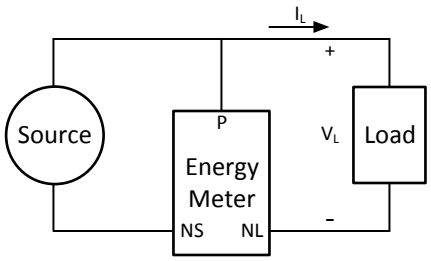


Figure 2.1: The energy meter sits between a source and a load. It has three terminals negative source (NS), negative load (NL), and positive (P) that are used to sense both the load voltage and load current.

(NL) terminal and the positive terminal (P). The meter senses the load current by measuring the current that flows between the negative load (NL) and negative source (NS) terminals.

A schematic of our power and energy meter is shown in Figure 2.2. The component values are left for you to calculate as a lab exercise[1]. The meter uses our microprocessor module (Chapter B) for analog measurement and display, the current sense resistor on the component module to measure the load current (Chapter C), and an operational amplifier to amplify the voltage across the current sense resistor.

Two analog inputs of the processor module, A0 and A1 are used to sense the load voltage and current respectively. A voltage divider (Section H.4.1) composed of resistors R1 and R2 scales the load voltage V_L to the 0-2.5V input range of the processor's A/D converter $V(A0)$. The zero point, the voltage at A0 when V_L is zero, is set by the voltage divider formed by R_7 and R_8 and the voltage follower (Section K.2) of $U1B$. This voltage follower also sets the zero point for the current amplifier. For a DC energy meter, the voltage divider of R_7 and R_8 is used to bias the zero-input voltage on A0 and A1 up 10-20mV to avoid the dead zone at the bottom of the A/D range that is caused by the internal ground of the processor being slightly higher than the external ground. For an AC energy meter, the voltage divider of R_7 and R_8 sets the zero-input voltage on A0 and A1 to 1.25V (the center of the A/D range) to allow bipolar (positive and negative) measurement.

The $R_S = 10\text{m}\Omega$ current sense resistor connected between NL and NS converts the 0-20A load current into a 0-200mV voltage. For the AC meter this same resistor converts the -10A to 10A load current into a -100mV to 100mV across R_S. We connect NL to processor ground so that our voltage measurement

[1]Different component values will be used for the DC and AC energy meters. The circuit otherwise remains the same.

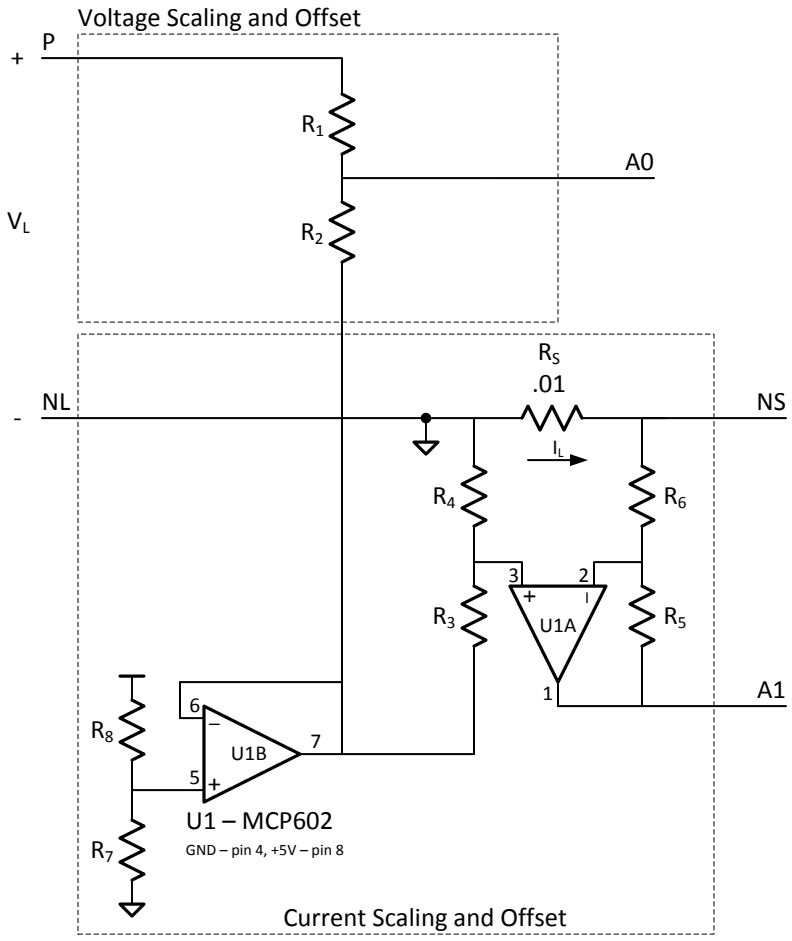


Figure 2.2: Schematic diagram of the energy meter lab.

Figure 2.3: Photograph of the energy meter

is the voltage across the load - which differs from the voltage across the source by the amount dropped across the current sense resistor[2] We use an operational amplifier (U1A) connected as a differential amplifier (Section K.2) to amplify the 200mV voltage across the sense resistor to fit the 2.5V input range of the A/D. We use a differential amplifier here, rather than a single-ended inverting or non-inverting amplifier, to cancel any noise that exists between terminal NL and the actual processor ground. The gain of this amplifier is set by the choice of resistor values.

A second operational amplifier (U1B) sets the output voltage on A0 A1 when there is zero voltage and zero current respectively. This opamp is wired as a voltage follower that provides a low-impedance copy of the voltage generated by the voltage divider of R_8 and R_9. For a DC energy meter, a voltage of 10-20mV is used to avoid dropping below the bottom end of the A/D range. For an AC energy meter, a voltage of 1.25V is used to enable bipolar measurement.

A photo showing the assembled energy meter is shown in Figure 2.3

For DC power measurements we need only to measure positive voltage and current. To take AC measurements we need to measure bipolar (both positive and negative) voltages and currents. Different resistor values will be used for AC and DC energy meters to accomodate different input ranges for voltage and current.

2.2 Software

Figure 2.4 shows a block diagram of the energy-meter software. The software consists of a sensing module, a calibration module, a calculation module, and a display module. The sensing module periodically samples the instantaneous current and voltage and generates variables `current.raw` and `voltage.raw`. The calibration module stores the measurements of known, precise current and voltage levels and generates variables `current_zero`, `current_1a`, `voltage_zero`, and `voltage_1v` which are used to calibrate the raw current and voltage measurements to generate accurate measurements in the presence of offsets and variation in the measurement circuitry, `current.value` and `voltage.value`. The calculation module takes the calibrated current and voltage and calculates power and energy `power`, and `energy`. A display module outputs current, voltage, power and energy to the display and provides functions for performing calibration.

[2]This connection of ground to NL does mean that NS will drop below ground when there is a positive current into the load. This is acceptable because the OpAmp input voltages remain positive. Also, the MCP602 can tolerate negative input voltages down to -300mV.

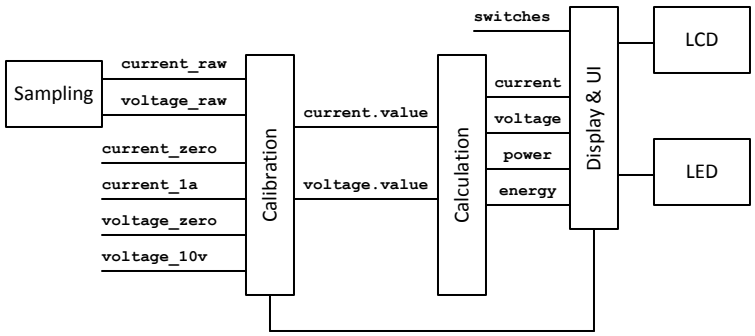


Figure 2.4: Block diagram of energy meter software

2.2.1 Sensing Software

Sensing is performed by the analog-to-digital converter (ADC) in our microprocessor. This ADC operates on a 125kHz clock and generates a 10-bit conversion in 13 cycles, 104μs. We request ADC operations for both voltage and current every 500μs as sequenced by a real-time clock interrupt. The actual conversions will occur 104μs apart.

Figure 2.5 shows the real-time-clock interrupt handler code that starts A/D conversions. Each 500μs, timer 2 invokes this procedure. The code sets the A/D request vector to request both current and voltage samples. and starts the A/D converter. Interrupts are then reenabled with the call to `sei()`. This allows an A/D completion to interrupt any lower priority code. After interrupts are enabled, a counter loop then calls the 3ms service routine, `tick3()`, once every 3ms. This routine updates the display, samples inputs, and performs other less urgent functions.

The real-time-clock interrupt starts the A/D conversion. When the conversion completes, one of the two ADC interrupt handlers shown in Figure 2.6 is triggered. This code stores the input from the A/D (`adcv`) in the `raw` field of the appropriate calibration structure and calls the library function to compute a calibrated value. After the current variable is calibrated updated values of power and energy are computed by calling `do_compute()`.

For the DC energy meter you may want to compute filtered versions of voltage and current to reduce measurement noise using the library filter function. For the AC energy meter lab you should compute power and energy with unfiltered values because averaging before computing power will reduce accuracy. For the AC lab you will need to compute the average magnitude or root-mean-square (RMS) value of the voltage or current for display.

You will write the computation and display code as part of the laboratory

```
rtc()
{
  // request conversion on channels 0 and 1
  adreq |= AD_1 | AD_0 ;
  if(adidle) start_ad() ;

  // reenable interrupts
  sei() ;

  // 3ms timer
  if(++t3 > T3) {
    t3 = 0 ;
    tick3() ;
  }
}
```

Figure 2.5: RTC interrupt handler code

```
void ad_0() { // voltage
  voltage.raw = adcv ;
  calibrate(&voltage) ;
}
void ad_1() { // current
  current.raw = adcv ;
  calibrate(&current) ;
  do_compute() ;
}
```

Figure 2.6: ADC interrupt handler code

assignment.

2.3 The Lab

For the laboratory you are to do the following:
Part 1: DC energy meter:

1. Calculate the required resistance values to measure 0-60V DC, 0-20A with a 10mΩ sense resistor.
2. Build the energy meter of Figure 2.2 with these values.
3. Using the *starter code* provided, complete the calculation and display modules to computer power and energy and to display current, voltage, power, and energy on the LCD display. Your code will also need to provide a way to measure and store the calibration values for your energy meter.
4. Using a lab supply and a lab voltmeter and ammeter, calibrate your energy meter.
5. Using your energy meter, measure the voltage, current, power and energy of (a) a 48V battery driving a heating element, (b) a 48V battery driving an unloaded electric motor, (c) a 48V battery driving a loaded electric motor, (d) the load of the setup of (c), and (e) a photovoltaic cell connected directly to a resistive load.

Part 2: AC energy meter:

1. Calculate the required resistance values to measure -200 to 200V, -10 to 10A with a 10mΩ sense resistor. Note that a 120V AC line will have voltages from -170V to 170V over one AC cycle.
2. Build the energy meter of Figure 2.2 with these values.

   The AC input will need to be fused but not rectified. To do this with the component module you will need to jumper across the rectifier diode D_1. To avoid reverse biasing the electrolytic capacitor, wire POW to COM. Connect nothing else to COM. Wire AC *Hot* (black wire) from the line cord to the quick-connect lug that feeds the fuse. Connect AC *Neutral* (white wire) from the line cord to one end of the current sense resistor. Connect your load between the other end of the current sense resistor and POW. You can monitor the load voltage — with a 100kΩresistor to serve as the upper branch of your voltage divider — via terminal *P+*.
3. Modify your energy meter code from part 1 to output average AC voltage and current and average power[3]. Voltage, current, and power computations will need to be integrated over at least one AC cycle to get an accurate measurement of RMS or average values.

[3]We leave as an extension computing RMS voltage and current which is what is actually wanted.

4. Using a lab supply and a lab voltmeter and ammeter, calibrate your energy meter.

5. Check out your AC energy meter using a DC lab supply. Reverse the polarity of the input and verify that current and voltage are negative but that power and energy are positive. Swap the source and load and verify that power and energy are negative.

6. (Optional) Using your energy meter, measure the voltage, current, power and energy of (a) the 110V line driving a resistive load, (b) a 110V line driving a light bulb, (c) a 110V line driving a fan, and (d) a 110V line driving a 2.2uF 200V capacitor.

2.4 Extensions

The following are some optional extensions that can be added to the lab:

Timer: Add a feature to the energy meter that allows it to measure energy for a fixed period of time and then stop — holding the final value measured.

RMS Voltage and Current: Add a feature to the AC energy meter to compute *root-mean-square* values for voltage and current measured over the last 50ms or 1s.

Efficiency Meter: Design an efficiency meter - two energy meters one for input and one for output and compute the efficiency of the device connected between the two meters. Be careful to account for energy stored in the device (in capacitors, inductors, and batteries). You can implement this with one processor module by using four of the analog channels.

Temperature Compensation: The resistors we use in the lab have a fairly high temperature coefficient. As a result, your calibration is really only valid at a single temperature. You can compensate for this by measuring calibration coefficients at two widely spaced temperatures and then measuring the actual temperature and computing appropriate coefficients via interpolation. Using an MCP9701A thermal sensor on analog channel 3 implement a power and energy meter with two-point temperature compensation.

Anti-Aliasing: Our input signal conditioning circuit (Figure 2.2) has a flat frequency response. Thus, noise of all frequencies is input to the A/D converter. The sampling process *aliases* high-frequency noise down into the band sampled. To reduce noise, and eliminate aliasing, we can implement a low-pass filter in each A/D input. With our 1kHz sample rate we can measure frequencies only up to 500Hz. Higher frequencies will alias into the band from DC to 500Hz. Thus, our low-pass filter should attenuate frequencies above this point.

Design an anti-aliasing filter — i.e., a low-pass filter with a cutoff freuqency of $f_c = 500$Hz (or lower). This can be accomplished by adding just two capacitors to Figure 2.2. Implement your filter, and test the resulting energy meter. Compare its performance to the energy meter without filtering.

Power Factor Measurement: Power factor is the ratio $\frac{P}{|VI|}$ averaged over an AC cycle. For a resistive load, the power factor is 1 while for a reactive load, a capacitor or inductor, the power factor is zero. Modify your energy meter to measure and display power factor.

Autoranging: Our energy meter has poor relative precision at the low end of its range because the fixed gain is set for the largest possible signal. Fix this issue by modifying the energy meter to automatically vary its current and voltage gain depending on the magnitude of the signal being measured.

For example, for the DC energy meter, the voltage gain can be varied by using a few digital outputs of the microprocessor to pull down on one of several resistors to form the lower branch of the voltage divider (R_2). The outputs connected to the unused resistors are left floating. Gain variation can aslo be achieved using an external analog multiplexer (be careful about common-mode range) or by using multiple analog inputs with different gain into each.

Better A/D Converter: The 10-bit A/D converter in the AVR microprocessor is limited. Modify the energy meter to use a higher resolution external A/D to improve resolution.

Chapter 3

Motor Controller

Many green electronic systems use motors to interact with the physical world. For example, an electric or hybrid vehicle uses a motor to propel the vehicle. In other cases motors are used to pump fluids, move air or other gases, control valves, etc.... Motors are how all electronic systems affect the physical world.

In all systems that use motors, a *motor controller* is used to control the motor by regulating the motor voltage and current to achieve a desired angular velocity ω and/or torque τ. Intelligent control of a motor can often result in significant energy savings. For example, rather than controlling motor speed by linearly dropping a fixed supply voltage — which dissipates very high power in the regulator, we use pulse-width modulation to provide the correct *average* voltage to the motor with very low losses in the controller. As a second example, the motor can act as a generator during braking — converting the kinetic energy of the vehicle to electrical energy and returing it to the battery. Many electric and hybrid vehicles use this technique, which is called *regenerative braking* to improve their efficiency.

This chapter describes a laboratory where we will build a motor controller and use this controller to drive a permanent-magnet DC motor at constant speed under variable load.

In the course of doing this lab you will learn:

Motor Theory: The basics of how voltage v, current i, angular velocity ω, and torque τ are related.

Pulse-Width Modulation: How we can create an effective voltage $\hat{v}$ by controlling the duty-factor of a square-wave signal that switches between two power supplies.

Buck and Boost Converters: How switching an inductor can (to first approximation) losslessly convert power from a higher voltage to a lower voltage (a buck converter) and how that same converter running backwards (a boost converter) converts power from a lower voltage to a higher voltage.

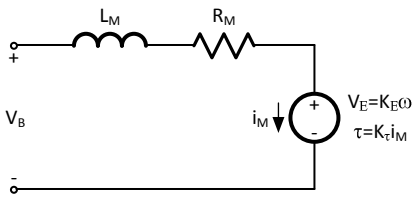


Figure 3.1: We model a motor as a velocity-dependent voltage source V_E in series with the inductance and resistance of the rotor winding. The motor torque τ is proportional to motor current i_M.

Modeling: How to build a mathematical model of a physical device so you can reason about it. In this case, you will build a model of the motor.

Feedback Control: You will learn the theory of feedback control and build a simple controller to hold your motor at constant velocity under variable load.

Simulation: You will build a Matlab model of your motor and controller and verify the operation of your controller in simulation before trying it on the real motor.

3.1 Motor Model

Developing a model, a simple representation of a more complex device or system, is a key aspect of engineering. A model allows us to reason about the device, include the device in a design, and analyze systems that include the device.

A good model is one that captures the properties of the device that are relevant for the task at hand and omits detail not necessary for this task. Here we develop an electrical model that captures the properties needed to build our controller. If instead our task was designing the chassis for an electric car we would omit the electrical properties but include the shape and mass of the motor.

Motors come in many types depending on how they are wound and whether they are intended for AC or DC excitation. In this chapter we will design a controller for a *DC permanent-magnet motor*. This is a motor that uses a permanent magnet for its stator and uses a set of brushes to commutate a DC voltage across a winding on its rotor. We choose the DC permanent magnet motor because it has a simple behavior and is easy to control. Other motor types behave in similar ways but with more complexity.

The simple way to think about motors is to remember that voltage v is proportional to motor speed ω and current i is proportional to motor torque τ.

Hence electrical power ($P_E = iv$) is proportional[1] to mechanical power ($P_M = \omega\tau$).

Figure 3.1 shows the model of a DC permanent-magnet motor we will use in this chapter. The motor is modeled as a voltage source V_E that represents the electromotive force generated by rotating the rotor windings through the magnetic field of the stator. As the motor rotates faster, $d\phi/dt$ increases, and thus V_E increases.

$$V_E = \frac{d\phi}{dt} = K_E\omega. \tag{3.1}$$

The current through the rotor generates a force with the magnetic field of the stator ($F = iL \times B$). This force in turn generates a torque τ on the rotor.

$$\tau = i(L \times B)r = K_\tau i. \tag{3.2}$$

The motor dynamics are governed by the angular momentum of the rotating system

$$\omega = \frac{1}{\mathcal{I}_\mathcal{M}} \int \tau dt, \tag{3.3}$$

where $\mathcal{I}_\mathcal{M}$ is the moment of inertia of the rotating part of the motor. In the case where the motor is propelling a mass, such as a vehicle, the mass can be converted to an addition to the moment of inertia.

Our model includes the parasitic inductance L_M and resistance R_M of the rotor winding.

Figure 3.2 shows a plot of the torque of a motor as a function of speed with a fixed voltage applied subject to a current limit. Consider applying a fixed voltage V_M to a motor with a stopped rotor, $\omega = 0$, and a very high moment of intertia — so its mechanical time constant is long compared to L_M/R_M. Initially, $V_E = 0$ and the current quickly rises to $i_0 = V_M/R_M$ which in turn induces a torque of $\tau_0 = K_\tau V_M/R_M$. Because R_M is typically very low, it is easy to get very high currents (and high torques) when applying voltage to a stopped motor. To avoid destruction of the motor, we limit the maximum current to $I_{\max}$ which gives a flat torque of $\tau_{\max} = I_{\max}K_\tau$ until the motor reaches a velocity where $i(\omega) = (V_M - K_E\omega)/R_M$ falls below $I_{\max}$.

As a motor gathers speed, the electromotive force generated by its rotor opposes the applied voltage and reduces the current and hence the torque. At speed ω, we have

[1] Except for losses (like resistive heating of windings), the electrical power into the motor should equal the mechanical power out of the motor.

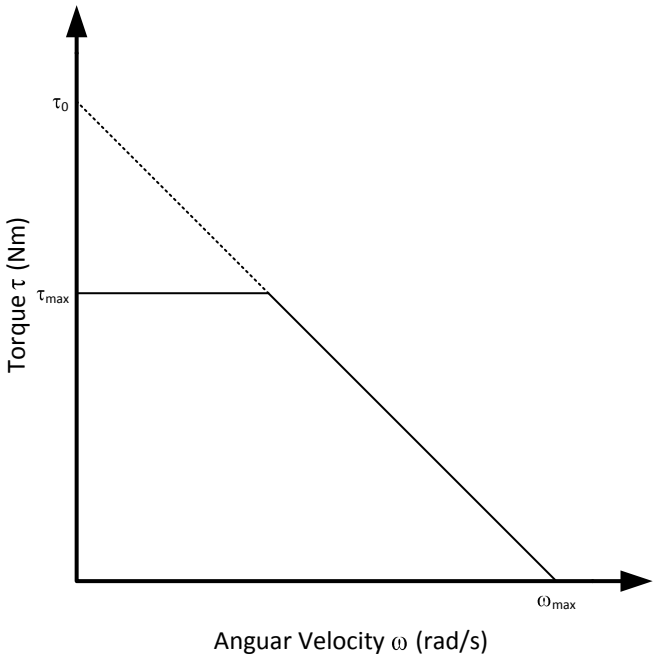


Figure 3.2: Motor torque τ as a function of velocity ω for a constant applied voltage subject to a current limit $I_{\max}$. While current limited τ is constant, then τ falls off linearly with ω.

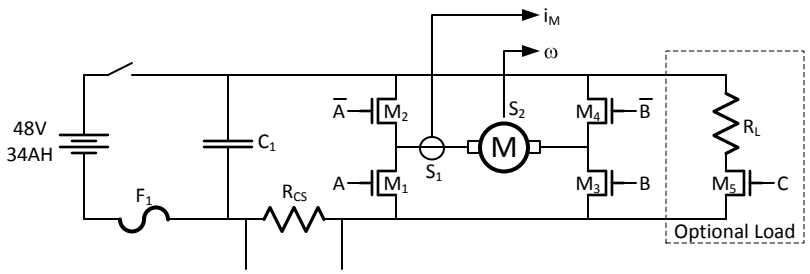


Figure 3.3: Our motor controller power path uses a *bridge* of power MOSFETs to connect the battery voltage or ground to either terminal of the motor. A Hall-effect current sensor and an angular velocity (RPM) sensor give feedback on the state of the motor.

$$i(\omega) = \frac{V_M - K_E\omega}{R_M}, \tag{3.4}$$

$$\tau(\omega) = K_\tau i(\omega) = K_\tau\left(\frac{V_M - K_E\omega}{R_M}\right). \tag{3.5}$$

In the absence of friction, our motor will increase speed until it reaches a speed $\omega = \frac{V_M}{K_E}$ where $V_E = V_M$ and $i_M = 0$. In practice, the speed will stabilize at a point where there is just enough current flowing to generate a torque that balances the friction on the rotor.

3.2 Power Path

As with most green-electronic systems, our motor controller consists of a *power path* and a controller. The power path handles the high-power and high current portions of the circuit. The controller monitors sensors and manipulates signals to make the decisions required to control the power path.

We control our motor by using *pulse-width modulation* (PWM) to apply a desired *average* voltage $\hat{v}$ to the motor by alternating between zero voltage and the full battery voltage. For example, if we apply a $V_B = 48$V battery voltage with a 10% duty factor, the result is a $\hat{v} = 4.8$V average voltage. If our PWM frequency is sufficiently high, the inductance of the rotor, L_M in Figure 3.1 smooths the resulting current flow, so the instantaneous current i is very close to the average current $i \approx \hat{i} = \frac{\hat{v}-K_E\omega}{R_M}$. This current applies a torque that when summed with the load on the motor will accelerate or decelerate the motor.

The power path that applies pulse-width modulated battery voltage to the motor is shown in Figure 3.3. The circuit uses a bridge of four MOSFETs M_1 through M_4 to apply either V_B or ground to either terminal of the motor. The transistors in the bridge operate in pairs so that if M_1 is on M_2 is off and vice versa. Similarly for M_3 and M_4. To apply positive voltage across the motor we turn on M_2 and M_3. Turning on M_1 and M_4 applies negative voltage across the motor.

If M_2 and M_4 are turned on, the current in the inductance of the rotor *freewheels* around the loop formed. The inductor current freewheels around the lower loop when M_1 and M_3 are turned on. If either of these cases persists long enough for the current in the motor inductor reverse direction, the motor will be *braked* by having its EMF, V_E in Figure 3.1 induce a reverse current i that produces a torque that opposes the direction of rotation. More on braking in Section 3.3.

If both M_3 and M_4 are turned off (or if both M_1 and M_2 are turned off), the motor current will go to zero — after the inductor returns its stored energy to the battery via the body diodes of the MOSFETS. In this state the motor will *coast*.

By convention we will use M_3 and M_4 to control the sign of the applied average voltage $\hat{v}$ and M_1 and M_2 to control the magnitude of $\hat{v}$. If we only want to operate the motor in one direction, we can omit M_3 and M_4 and tie the right terminal of the motor to ground. Setting $B = 1(\bar{B} = 0)$ gives a positive $\hat{v}$ and setting $B = 0(\bar{B} = 1)$ gives a negative $\hat{v}$. With $B = 1$, the duty factor of M_2, $\langle \bar{A} \rangle$ sets the magnitude of $\hat{v}$ while the duty factor of M_1 $\langle A \rangle = 1 - \langle \bar{A} \rangle$ sets the magnitude when $B = 0$.

In summary:

$$\hat{v} = \begin{cases} (1 - \langle A \rangle) V_B & \text{if } B = 1 \\ -\langle A \rangle V_B & \text{if } B = 0. \end{cases} \tag{3.6}$$

In the steady state, the motor controller converts energy at the battery voltage V_{Batt} to energy at the motor voltage V_E. It does this using a configuration called a *buck converter*, shown in Figure 3.4. Switch a corresponds to M_2 in Figure 3.3, switch b corresponds to M_1 and in this configuration M_3 is on. The inductor L is the motor inductor L_M of Figure 3.1.

Waveforms showing steady-state operation of the buck converter are shown in Figure 3.5. At t_0, the start of a PWM cycle, the inductor current is initially i_0 and switch a turns on. With a closed, the voltage across the inductor is $V_L = V_{\text{Batt}} - V_E$ so the current in the inductor ramps up with slope $\frac{di_L}{dt} = \frac{V_{\text{Batt}} - V_E}{L}$. When switch a turns off at time t_1 the inductor has reached a peak current of:

$$i_P = i_0 + \frac{(t_1 - t_0)(V_{\text{Batt}} - V_E)}{L} = i_0 + \frac{t_{\text{on}}(V_{\text{Batt}} - V_E)}{L}. \tag{3.7}$$

At this point switch b turns on, the voltage across the inductor becomes $V_L = -V_E$, and the inductor current ramps down with a slope $\frac{di_L}{dt} = \frac{-V_E}{L}$. It reaches its initial value at t_2, the end of the PWM cycle.

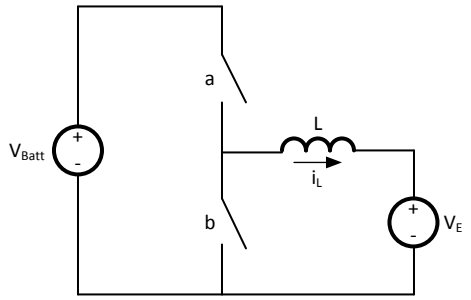


Figure 3.4: A buck converter converts energy from a higher volage — in this case V_{Batt} — to a lower volage — in this case V_E by storing energy from V_{Batt} in the current of the inductor i_L when switch a is closed and then driving this energy out at the lower voltage when switch b is closed.

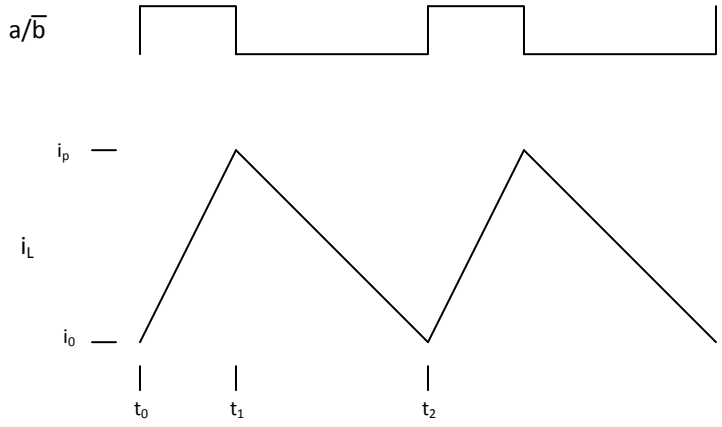


Figure 3.5: Waveforms illustrating operation of the buck converter of Figure 3.4. Upper waveform is high when switch a is closed and low when switch b is closed. Lower waveform shows the current in the inductor.

For steady-state operation the current at the end of the cycle must be the same as the current at the beginning of the cycle. Thus we have the relationship

$$t_{\text{off}} V_E, = t_{\text{on}}(V_{\text{Batt}} - V_E), \tag{3.8}$$

$$(t_{\text{off}} + t_{\text{on}})V_E = t_{\text{cy}} V_E = t_{\text{on}} V_{\text{Batt}}, \tag{3.9}$$

$$\frac{V_E}{V_{\text{Batt}}} = \frac{t_{\text{on}}}{t_{\text{cy}}} = \langle A \rangle. \tag{3.10}$$

Thus, we see that the buck converter — by storing energy in the magnetic field of an inductor while moving it from one voltage to another — produces an average voltage proportional to the duty factor at its output $\hat{v} = \langle A \rangle V_{\text{Batt}}$.

The buck converter works equally well regardless of the initial inductor current i_0. In fact, if i_0 is negative the converter can work backwards — transferring energy from the lower-voltage "output" to the higher voltage "input". Operating in this mode, we call it a *boost converter* (see Section 3.3). The direction and magnitude of i_0, and hence the current and power flow, in a buck converter is determined by the voltages and resistances of the input and output circuits. The buck converter just sets the voltage ratio.

For example, suppose the duty factor is set at 1/3, the input is an ideal voltage source (zero resistance) of V_{Batt}=48V, and the output is voltage source of 15V in series with a resistance of 1Ω. In this case, the converter converts the 48V to 16V which gives a 1V drop across the load resistance giving a 1A current. If the load source increases to 17V we get 1A flowing in the opposite direction.

The buck converter is widely used in green electronics, and in power electronics more generally. We will see it again.

As a compromise between minimizing switching losses and minimizing ripple current in the inductor's windings we choose a PWM frequence of 15kHz.

The power path of Figure 3.3 includes three sensors. A hall effect sensor S_1 measures the current into the positive terminal of the motor. This lets us directly measure i_M and hence τ. An optical sensor S_2 generates a pulse train with a frequency proportional to the motor speed $|\omega|$. Finally, a sense resistor R_{CS} measures the instantaneous current applied during each pulse.

The input circuit includes a filter capacitor and a fuse. Filter capacitor C_1 filters the 15kHz current pulse train to limit the voltage ripple across the bridge. The battery is switched and is fused at its negative terminal. Fusing at the negative terminal of the battery protects against a short from an intermediate terminal of the battery stack to ground.

The power path includes an optional load resistor R_L that is switched by MOSFET M_5. This load is used to draw current away from the battery to prevent the battery from overcharging during regenerative braking. By pulse-width modulating signal C controlling M_5 with duty factor $\langle C \rangle$ the effective load resistance becomes $R_{\text{eff}} = R_L / \langle C \rangle$.

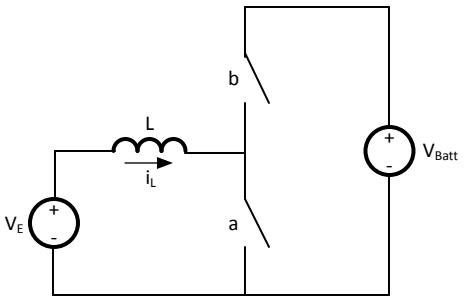


Figure 3.6: A boost converter converts a low volage — in this case V_E — to a higher volage — in this case V_{Batt} by storing energy from V_E in the current of the inductor i_L when switch a is closed and then driving this energy out at the higher voltage when switch b is closed. A boost converter is just a buck converter with its input and output reversed.

3.3 Regenerative Braking

To perform regenerative braking, we want to allow the motor to operate as a generator and return its mechanical kinetic energy to the battery. The problem, however, is that the voltage generated by the motor is too low to drive current into the battery. At speed ω the voltage out of the motor is $V_E = K_E\omega$ which is strictly less than the applied voltage V_M that we used to originally generate this velocity.

To return energy to the battery we need to convert the energy out of the motor from V_E to V_{Batt}. To convert from a lower voltage to a higher voltage we use a circuit called a *boost converter* shown in Figure 3.6. The attentive reader will notice that this converter looks a lot like the buck converter of Figure 3.4. In fact, a boost converter is just a buck converter with its source and load reversed.

Operation of this circuit is depicted in the waveforms of Figure 3.7. The upper trace shows the state of the switches and the bottom trace shows the inductor current i_L. Operation is identical to that of the buck converter of Figure 3.4 but with the perspective of input and output, and hence the polarity of inductor current, reversed.

At t_0, the start of a PWM cycle, the inductor current is initially i_0 and switch a turns on. With a closed, the voltage across the inductor is $V_L = V_E$ so the current in the inductor ramps up with slope $\frac{di_L}{dt} = \frac{V_E}{L}$. When switch a turns off at time t_1 the inductor has reached a peak current of:

$$i_P = i_0 + \frac{(t_1 - t_0)V_E}{L} = i_0 + \frac{t_{\text{on}}V_E}{L}. \tag{3.11}$$

At this point switch b turns on, the voltage across the inductor becomes $V_L =$

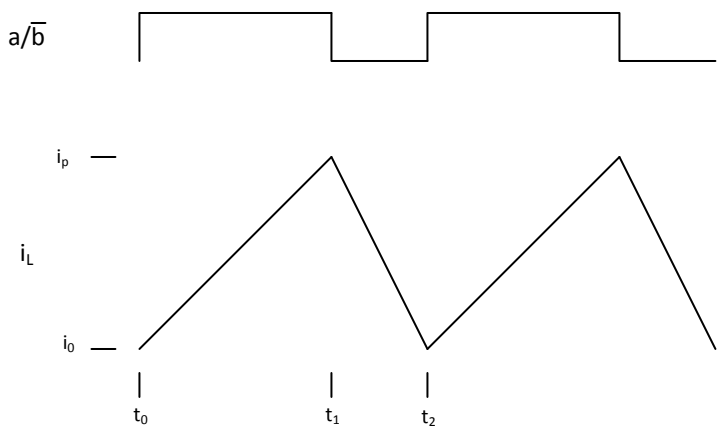


Figure 3.7: Waveforms illustrating operation of the boost converter of Figure 3.6. Upper waveform is high when switch a is closed and low when switch b is closed. Lower waveform shows the current in the inductor.

$V_E - V_{\rm Batt}$, and the inductor current ramps down with a slope $\frac{di_L}{dt} = \frac{V_E - V_{\rm Batt}}{L}$. It reaches its initial value at t_2, the end of the PWM cycle.

For steady-state operation the current at the end of the cycle must be the same as the current at the beginning of the cycle. Thus we have the relationship

$$t_{\rm off}(V_{\rm Batt} - V_E), \quad = \quad t_{\rm on} V_E, \tag{3.12}$$

$$(t_{\rm off} + t_{\rm on})V_E = t_{\rm cy} V_E \quad = \quad t_{\rm off} V_{\rm Batt}, \tag{3.13}$$

$$\frac{V_E}{V_{\rm Batt}} \quad = \quad \frac{t_{\rm off}}{t_{\rm cy}} = \langle A \rangle. \tag{3.14}$$

As expected, we have derived the same relationship as (3.2) but with $t_{\rm on}$ replaced by $t_{\rm off}$ because we have reversed the switches.

What this analysis shows, perhaps redundantly, is that when we apply an effective average voltage less than the current motor EMF, $\hat{v} < V_E$ current will flow from the motor to the battery. The magnitude is set by the motor resistance, R_M in Figure 3.1.

3.4 Matlab Simulation

Simulation and analysis are the main tools an engineer uses to check that a design works and to refine a design — to make it work better. Analysis involves using a mathematical models of an engineering system, like a circuit, to calculate properties of the circuit. Simulation also uses a mathematical model of the

```
v_e = omega * k_e ;          % motor emf
v_l = v_in-v_e-i_m*r_m ;     % inductor voltage
i_m = i_m + v_l*dt/l_m ;     % integrate motor current
tau = tau_a + i_m*k_tau ;    % total torque
omega = omega + tau*dt/mi ; % integrate angular velocity
```

Figure 3.8: Matlab code to simulate the motor controller for one time step `dt`. State variables `i_m` and `omega` are advanced using forward differencing in response to input variables `v_in` and `tau_a`.

system. However, rather than analyzing properties of the model, we use a computer to *simulate* the behavior of the model.

Simulation and analysis are complementary tools. With analysis we can prove very general results about a system across a wide range of operating conditions and parametric values. However, we can typically only analyze very simple systems before the calculations become prohibitively complex. Simulation, on the other hand, sheds light only on the particular operating conditions and parameters simulated. However we can simulate very complex systems — systems that are far too complex to analyze by hand.

To verify that our controller works, we can model it with a computer program and simulate its operation. The Matlab package is particularly convenient for this type of modeling.

The type of model we build depends on what we want to accomplish. If we want to study the ramp of the current in the inductor during a PWM cycle we need to model the inductor and pick a time step that is small compared to $t_{\rm cy}$. If, on the other hand, we only want to simulate motor dynamics we can use a larger time step and just model the response of the motor to the effective average voltage $\hat{v}$ and applied torque τ_a.

A Matlab script that simulates the motor model, including the inductor, for one time step is shown in Figure 3.8. This script evaluates the equations of Section 3.1 using *forward differencing* — we approximate an integral by multiplying the rate of change by the timestep[2]. For example, from (3.1) we know $\omega = \int \frac{\tau}{\mathcal{I}_\mathcal{M}} dt$. For one time step Δt we approximate this integral as $\omega(t+\Delta t) = \omega(t) + \Delta t \left(\frac{\tau}{\mathcal{I}_\mathcal{M}}\right)$. We perform a similar integration for i_M.

Figure 3.9 shows the simulated inductor current i_M as a function of time for three $f_{\rm cy} = 15$kHz PWM cycles. For this simulation we set the duty factor $\langle A \rangle$ =0.25 which corresponds to $\hat{v}$=12 and set ω to give steady-state operation — i.e., i_M varies over a cycle but not from one-cycle to the next. As predicted by our analysis (Figure 3.5) the current is a sawtooth waveform that slopes upward during the first quarter of the cycle — when M_2 is on — and downward

[2]If you take a class in numerical methods you will find out that there are much better integration methods, we use forward differencing here because it is simple and suffices for our needs.

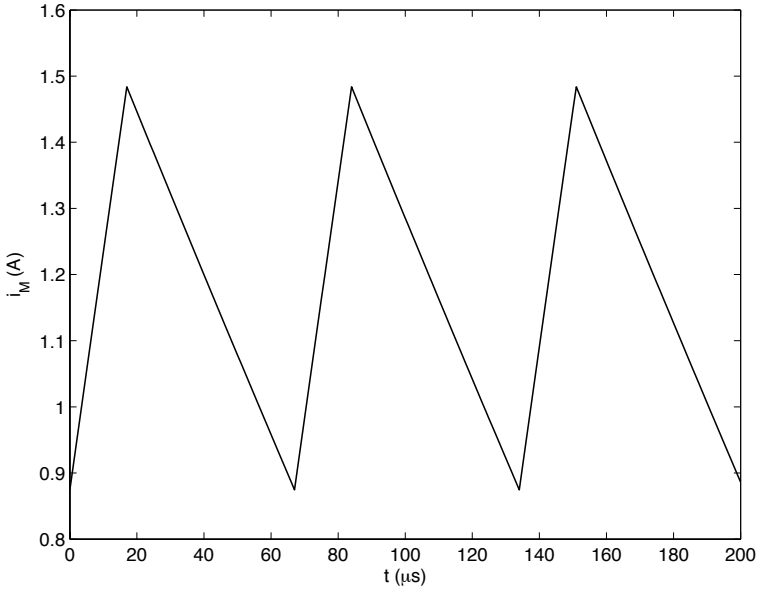


Figure 3.9: Waveform showing simulated inductor current i_M of a motor as a function of time with 15kHz PWM ($t_{\rm cy} = 66.7\mu$s) and a duty factor of $\langle A \rangle$=0.25.

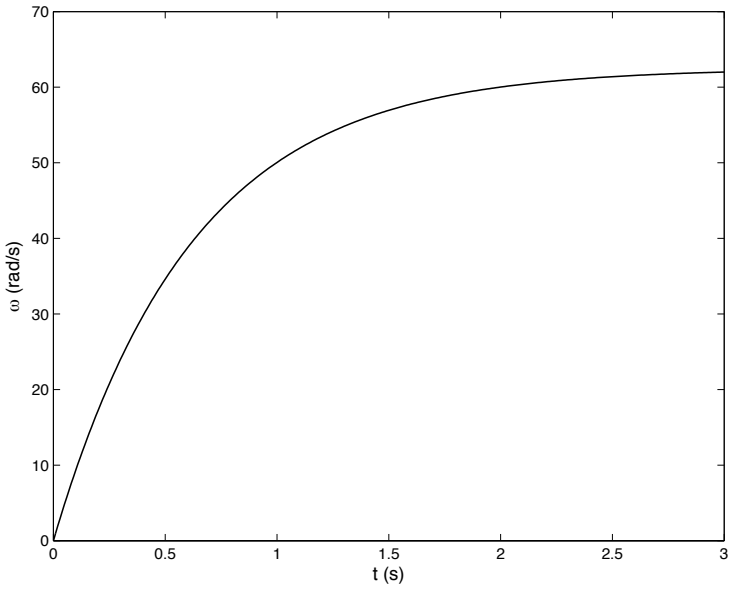


Figure 3.10: Waveform showing simulated angular velocity ω of a motor as a function of time with a constant applied voltage V_M=10V.

for the last three quarters of the cycle — when M_1 is on.

Figure 3.10 shows the transient response of the motor to a 10V step in $\hat{v}$. This simulation shows the motor exponentially converges to its steady-state speed of ω= 62.5 rad/s with a time constant of τ_M=0.625s.

The simulation of Figure 3.10 was run using the model of Figure 3.8 which required using a small time step (1μs) to maintain stability for the numerical integration of inductor current. To allow the use of longer time steps for simulations where we care about the motor dynamics, not the dynamics of a PWM cycle, we can use the model of Figure 3.11. This model omits the inductor from Figure 3.1 and just solves for i_M using Ohm's law. With this simplification much larger time steps (10ms) can be used which makes the simulations run much faster.

This omission of detail — the motor inductance — is the essence of good modeling. A good model omits the details that are not important to the problem at hand while accurately modeling the important factors. A factor is *important* here if it affects the quantity we are trying to measure with our simulation.

```
v_e = omega * k_e ;      % motor emf
i_m = (v_in-v_e)/r_m ;   % motor current - assumes no inductor
tau = tau_a + i_m*k_tau ;% total torque
omega = omega + tau*dt/mi ; % angular velocity
```

Figure 3.11: Matlab code to simulate the motor controller for one time step `dt` ignoring the inductor. This permits the use of a larger time step than the model of Figure 3.8 but cannot be used to study PWM dynamics.

3.5 Controller

When we use a motor in a Green Electronics system we typically want to control its position, velocity, or torque. To compute the appropriate input $\hat{v}$ to get the desired output we use a *controller*. Most controllers use feedback control as discussed in Appendix N.

Consider the problem of building a speed control. The input to our controller is a desired velocity ω_d. The output of our controller is the effective motor voltage $\hat{v}$ or equivalently the PWM duty factor $\langle A \rangle$. To compute $\hat{v}$, a *feedback control* system computes an *error signal*, $e(t) = \omega_d - \omega$ and then computes $\hat{v}$ as a function of $e(t)$.

A proportional-integral controller (Appendix N) computes the control signal by summing a weighted error signal and a weighted integral of the error signal.

$$\hat{v}(t) = pe(t) + q \int e(t)dt. \tag{3.15}$$

Here p is the proportional gain and q is the integral gain. Choosing these gains determines the performance and stability of a PI controller.

Our motor has a first-order low-pass response with a time constant τ_M that is related to the motor parameters[3].

$$\tau_M = \frac{R_M \mathcal{I}_\mathcal{M}}{K_\tau K_E}. \tag{3.16}$$

Thus, we can write the transfer function of the motor as

$$\omega(s) = \hat{v}(s) \left(\frac{1}{1 + \tau s} \right). \tag{3.17}$$

Lets pick the frequency[4] of our control system as $\omega_c = \frac{2}{\tau}$. We use the gain of the control system to give an overall system response that is faster than the plant being controlled.

[3] Be careful not to confuse τ_M the motor time constant with τ its torque — we clearly need more greek letters.

[4] Do not confuse ω_c, the natural frequency of the closed-loop system with ω the speed of the motor.

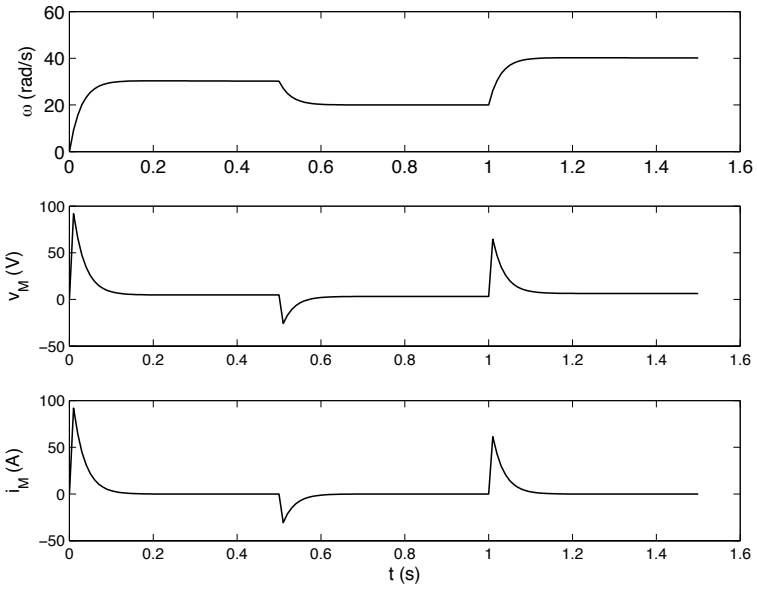


Figure 3.12: Simulation of the motor using a PI controller with the parameters of (3.5). From a stop, ω_d is set to 30 at $t = 0$, to 20 at $t = 0.5$s and to 40 at $t = 1s$. Top trace is motor speed ω, center trace is motor voltage $\hat{v}$, and bottom trace is motor current i_M.

We will pick the damping factor to be $\zeta_c = 1$ which gives critical damping. Using these choices for ω_c and ζ_c we can compute q and p from (N.10) and (N.11).

$$q = \omega_c^2 \tau = \frac{4}{\tau}, \tag{3.18}$$

$$p = 2\zeta\tau\omega_c - 1 = 3. \tag{3.19}$$

Figure 3.12 shows a Matlab simulation of our motor being driven by a PI controller with the control parameters derived in (3.5). In this simulation the motor starts at rest and the desired speed ω_d is set to 30 rad/s at $t = 0$s, to 20 rad/s at $t = 0.5$s, and to 40 rad/s at $t = 1$s. The top trace of Figure 3.12 shows that the PI controller works well — driving the motor to these desired speeds smoothly and with a time constant of about 0.1s — much faster than the 0.625s τ_M.

There is one serious problem with our PI controller, however. The voltages and currents it is applying to the motor exceed the bounds of what is possible.

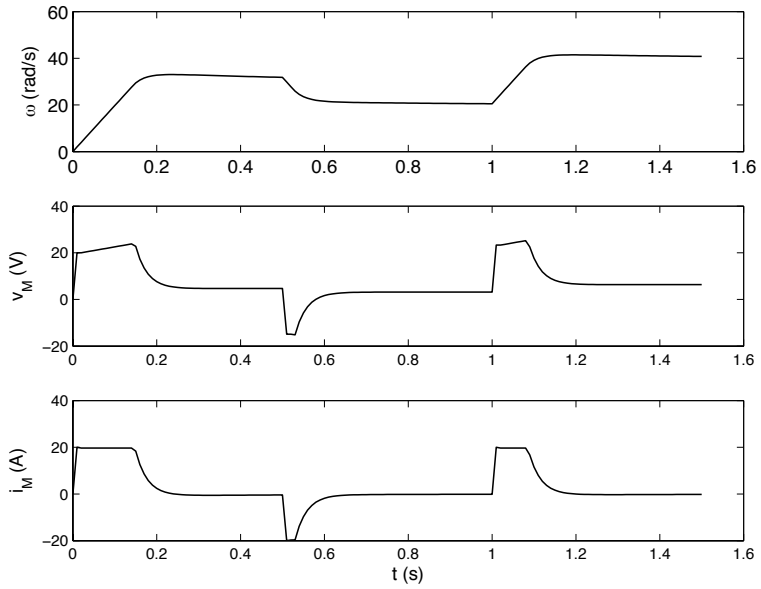


Figure 3.13: Simulation of the motor using a current-limited PI controller with the parameters of (3.5) and $i_{\max}$=20A. From a stop, ω_d is set to 30 at $t = 0$, to 20 at $t = 0.5$s and to 40 at $t = 1s$. Top trace is motor speed ω, center trace is motor voltage $\hat{v}$, and bottom trace is motor current i_M.

To achieve a quick response to the initial motor command, the controller is attempting to apply about 90V to the motor. This is not possible, our battery voltage is only 48V.

More seriously the motor current exceeds reasonable levels — going to nearly 90A. Even with a 48V voltage limit, driving the current to 48A is likely to cause damage to the power FETs, the wiring, and the motor.

Figure 3.13 shows simulation of a current-limited PI motor controller. Here we have modified our control law to limit current magnitude, $|i_M| \leq i_{\max} = 20$A. The result is a response that is slightly slower than that of Figure 3.12 but which avoids destruction of the motor controller, the motor, and interconnecting wiring. The response of Figure 3.12 is about as fast as can be achieved while staying within the specified current limit.

3.6 The Lab

1. Build the motor controller of Figure 3.3 using two Green Electronics *half-bridge* modules (Appendix D). Omit the optional load portion. As long

as the motor is to be run in only one direction the second half-bridge can also be omitted and the right terminal of the motor can be connected to GND.

2. Develop a motor model. This involves finding the parameters L_M, R_M, K_E, K_τ, and I_M from Figure 3.1, (3.1),(3.1), and (3.1). Write a program for your motor controller to carry out a series of experiments to measure these parameters. You have control of voltage v via the PWM duty factor and load and can measure current i and velocity ω. Some experiments you might consider include (a) running at a constant voltage at various loads and measure velocity ω and current i, (b) apply a voltage step and measure the transient response of ω and i. Run these experiments and derive your model. Test your model to see how well it predicts observed behavior.

3. Simulate your motor system in Matlab. Write a Matlab `.m` file that simulates operation of your motor system. Pick an appropriate timestep Δt and write code to update the state variables ω, and i each timestep as a function of applied voltage and torque. Run your simulation model on the test cases you used to develop your model and verify that you get the same response.

4. Develop a constant-speed controller. Develop a control algorithm and write the corresponding control software to maintain your motor at constant speed which not exeeding peak current. Test your controller first on your simulation model and then on the actual motor. You controller should hold speed constant while different loads are applied to the motor — corresponding to a car going up and down hills. It should also transition smoothly between two programmed speeds. The course staff will program an adversary load to make it difficult to hold a constant speed.

3.7 Extensions

Regenerative Braking: Add to your controller the ability to perform braking — reducing ω at a specified amount of negative torque τ or eqivalently current i_M. This is needed to slow a car with a controllable force depending on how hard the driver steps on the brake pedal.

Torque Control: Add to your controller the ability to acclerate with a specified amount of torque τ or equivalently current i_M. This is needed to give an acceleration force that is proportional to how far the driver pushes the accelerator pedal.

Optimize PWM Frequency: We picked the PWM frequency for this lab 15kHz somewhat arbitrarily before we measured the motor parameters.

Given the motor parameters you measured in your lab, determine an optimal PWM frequency. You may choose to optimize for efficiency, a frequency that gives the lowest losses, performance, a frequency that gives the fastest response to speed commands (staying within a peak current limit).

Program an adversary load: Write the code to drive the "load" generator to give a more sophisticated load for testing motor controllers. Your "load" should simulate the momentum of a car of a specified mass and should be able to model this car going up and down hills of various slope. Note that on a downhill the load will actually drive the motor and the motor will need to provide braking — reversing roles with the load.

Vehicle dynamics: Extend your motor simulation model with a model of the dynamics of a vehicle containing the motor. You should take into account the mass of the vehicle and account for the momentum of the vehicle and energy expended and released with the vehicle going up and down hills.

Motor estimation: With accurate motor parameters and a measurement of motor current and occasional updates of rotor position your should be able to estimate rotor position at any point in time. Write software to perform this estimation.

Brushless DC Motor Control: A brushless DC motor replaces the mechanical commutator of a conventional DC motor with semiconductor switches. Based on the position of the motor multiple windings are switched on and off and their polarity reversed to provide torque in the desired direction. Write control software to drive the windings of such a motor.

3-Phase AC Motor Control: An AC induction motor requires a three-phase sinusoidal drive where both amplitude and frequency can be controlled. Using three half-bridges build such a motor controller. As a first step, write sofware to generate three-phase sine-waves of specified frequency and amplitude. As a second step, couple this with a motor estimator and motor control algorithm to set the frequency, phase, and amplitude to perform constant-speed and constant-torque control with a peak current constraint.

Chapter 4

Battery Charger

Energy storage is an important component of many Green Energy systems, and is often provided by batteries. In this chapter we will develop a model of a battery and will then use our model do drive design of a battery charger. Different batteries have different properties, different models, and different charging requirements. In this chapter we restrict our attention to deep-cycle lead/acid batteries such as are found in golf carts and some other electric vehicles.

This chapter describes a laboratory where you will build a battery charger to charge a 48V lead-acid battery pack from 110V 60Hz AC wall current. In the course of this laboratory you will learn

Battery Characteristics: The basics of how batteries work as viewed from their terminals, in particular their V-I characteristics and charging characteristics.

Modeling: You will practice the modeling skills you developed in Chapter 3 by developing a model for our battery pack.

AC Rectification: How AC current is converted to DC current using a diode.

Buck Converter: Some details of the buck converter, introduced in Section 3.2, as it is applied to efficiently convert a high voltage to a lower voltage.

Layered Control: How to decompose a complex control task into several simple layers.

Current-Mode Control: A method for controlling a pulse-width-modulated system, like the buck converter, by controlling the current in the inductor each cycle.

4.1 Modeling a Battery

Just as we developed a motor model in Section 3.1 we will develop a battery model that captures just the behavior of the battery that we need to know to

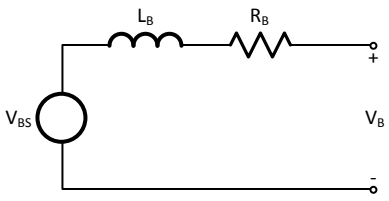


Figure 4.1: We model a battery as a charge-dependent voltage source with a series resistance and inductance.

reason about our curren task, designing a battery charger, and that abstracts away unecessary detail.

In this section, we will develop a circuit model of a lead-acid battery. This model is useful for designing circuits that include the battery — like the charger we will develop in this chapter, and the motor controller of Chapter 3. A different task may require a different model. For example, if the task is mechanical design of an electric vehicle, the model may include the shape, mass, and rigidity of the battery while ignoring its circuit properties.

Figure 4.1 shows our battery model which includes a charge- and temperature-dependent voltage source with series inductance and resistance. We model the voltage of the source as

$$V_{BS} = \begin{cases} 0 & \text{if } Q = 0 \\ N\left(V_{cell} + K_T T + V_Q\left(\frac{Q}{Q_C} - 1\right)\right) & \text{if } Q > 0 \end{cases} \tag{4.1}$$

where N is the number of cells in our battery, Q is the current batter charge (in Coulombs), Q_C is the capacity of our battery, V_{cell} is the nominal voltage of one battery cell at $T = 0$ degrees C, K_T is the temperature coefficient of one cell (in $\frac{V}{\deg C}$), and T is the battery temperature. This gives a linear $Q - V$ relationship where the source voltage varies from $V_{cell} - V_Q$ when fully discharged to V_{cell} when fully charged.

You will measure the parameters for your particular battery as a part of the lab. For comparison, Table 4.1 shows parameters for a typical battery.

4.2 Charging Sequence

Figure 4.2 illustrates the charging sequence we will employ which is divided into four regimes. For reference, the key charging parameters (voltages and currents) are listed in Table 4.2 with typical per-cell values for a 10A lead-acid battery.

As shown in Figure Figure 4.2, if the battery voltage is below V_T we first apply a gentle *trickle* current I_T to bring the battery up to the bottom of its

Parameter	Value	Units	Description
V_{cell}	2.25	V	Fully-charged voltage at 0C
V_Q	0.50	V	Cell discharge voltage at 0C
K_T	-3.9	$\frac{mV}{\text{deg}C}$	Temperature coefficient
Q_C	5×10^5	C	Battery capacity
R_B	0.1	Ω	Series resistance
L_B	10	μH	Series inductance
N	6		Number of series cells

Table 4.1: Parameters for a typical 12V lead-acid battery

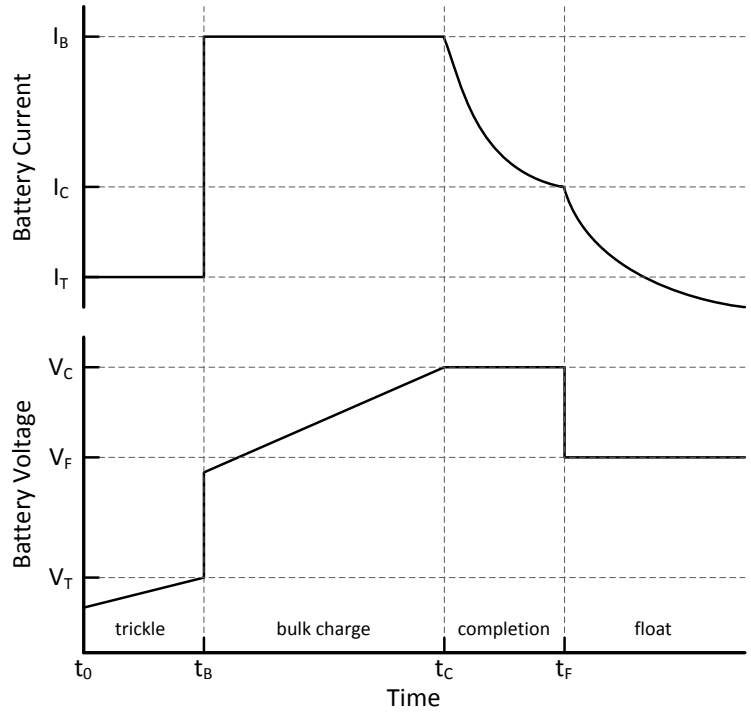


Figure 4.2: Charging a battery involves four steps. A *trickle* phase ($t_0 - t_B$) brings the battery up to its minimum charging voltage. A *bulk charge* ($t_B - t_C$) phase charges the battery at a constant current I_B. A *completion* phase ($t_C - t_F$) holds the battery at a constant voltage while the charging completes. Finally, a *float* phase ($t > t_F$) maintains the charge on the battery.

Parameter	Value	Units	Description
I_T	0.5	A	Trickle charge current
I_B	10	A	Bulk charge current
I_C	1	A	Fully-charged current (at $V = V_C$)
V_T	1.75	V	Minimum voltage for bulk charge
V_C	2.35	V	Fully-charged voltage (at $i = I_B$)
V_F	2.25	V	Float charge voltage

Table 4.2: Charging Voltages and Currents. All voltages are per cell.

linear region. Next, at time t_B, a bulk current I_B is applied to charge the battery to near capacity as rapidly as safely possible until the fully-charged voltage, V_C is reached. The battery voltage jumps up when the bulk current is applied because of its series resistance (R_B in Figure 4.1). Once the V_C is reached at time t_C, voltage is held constant at V_C until the current falls below I_C at time t_F signaling that the charge is complete. Finally, the battery voltage is held at a *float* value, V_{F} to maintain the charge. The battery voltage must not be allowed to remain above V_F for extended periods or the overvoltage will boil away the battery's electrolyte.

Different batteries have different optimal charging profiles. For a battery with different chemistry or a different type of lead-acid battery, you should consult the battery manufacturer for the proper charging sequence. An incorrect charging sequence can lead to premature battery failure, or, in the case of some battery types, fire or explosion of the battery.

4.3 Power Path of the Charger

As with most green-electronic systems, our charger consists of a *power path* and a controller. The power path handles the high-power and high current portions of the circuit. The controller manipulates signals to make the decisions required to control the power path. In this case the power path delivers power from an AC supply to the battery. The power path is described in this section. We discuss the controller in Section 4.4.

The power path, shown in Figure 4.3 consists of an input stage, a switched inductor, and an output filter in a configuration called a *buck* converter.

4.3.1 Input Stage

The input stage, consisting of fuse F_1, diode D_1 and capacitor C_1, rectifies and filters the AC input. This stage converts the 120V AC input to 167V DC. A fuse protects the input from over-current conditions. The diode rectifies the AC input — conducting only during the part of the AC cycle when $V_{\mathrm{ac}} > V_i$, charging C_1 to the peak of the AC cycle. The capacitor supplies current during the remainder of the AC cycle.

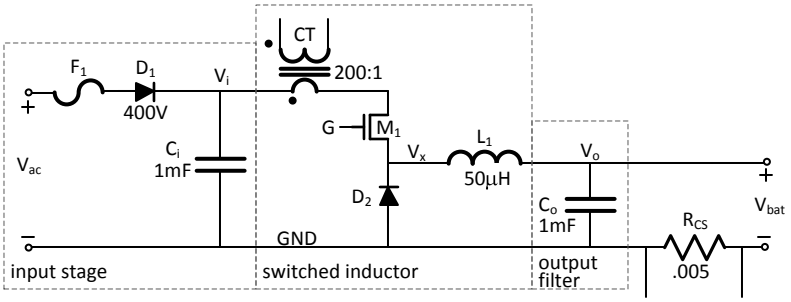


Figure 4.3: The power path of the charger.

An AC input cord has three conductors: *hot* (black), *neutral* (white), and *ground* (green). Current is only carried by the first two. The neutral (white) wire is grounded back at the breaker box and is nominally the safer of the two conducting wires. The hot wire oscillates about the neutral wire with an RMS value of 110V nominal (120V max).

The ground conductor of the AC line is just for circuit protection and should normally carry no current. If you build a circuit in a metal chassis, the ground conductor should be connected to the chassis. However it should not be connected to your circuit ground — or it will start to carry current. Do not confuse the ground conductor of the AC line (chassis ground) with the ground point in your circuit (circuit ground). Despite the similar names, the two are different and should not be connected together. If you accidentally short part of your circuit to the grounded chassis, current will flow in the ground conductor of the AC line. This is called a *ground fault* and if the circuit is protected by a ground-fault interrupter, the breaker will open.

Figure 4.4 shows the waveforms for the input stage. The top panel shows the AC input, $V_{\rm ac}$, (dotted) and V_i (solid). The 120V RMS AC input is a sinusoid with an amplitude of 170V and a frequency of 60Hz. The diode causes V_i to never fall below $V_{\rm ac}$. Each AC cycle V_i follows $V_{\rm ac}$ briefly, until the peak of the AC cycle. During the remainder of the cycle, $V_{\rm ac}$ slowly ramps down as the input capacitor is discharged driving the output.

At about 8ms, the voltage across diode D_1 (the distance between the two lines) is about 340V. To withstand this voltage, a diode with a reverse voltage V_R of at least 400V must be used for D_1.

The lower panel of Figure 4.4 shows the input current. This current is zero except during the brief period where V_i follows $V_{\rm ac}$. During this period the current is the superposition of the current $I = C_i \frac{dV_i}{dt}$ charging C_i and the inductor current. The peak value reached is over 30A. Thus, in addition to having an average forward current of 5A, diode D_1 must be able to withstand

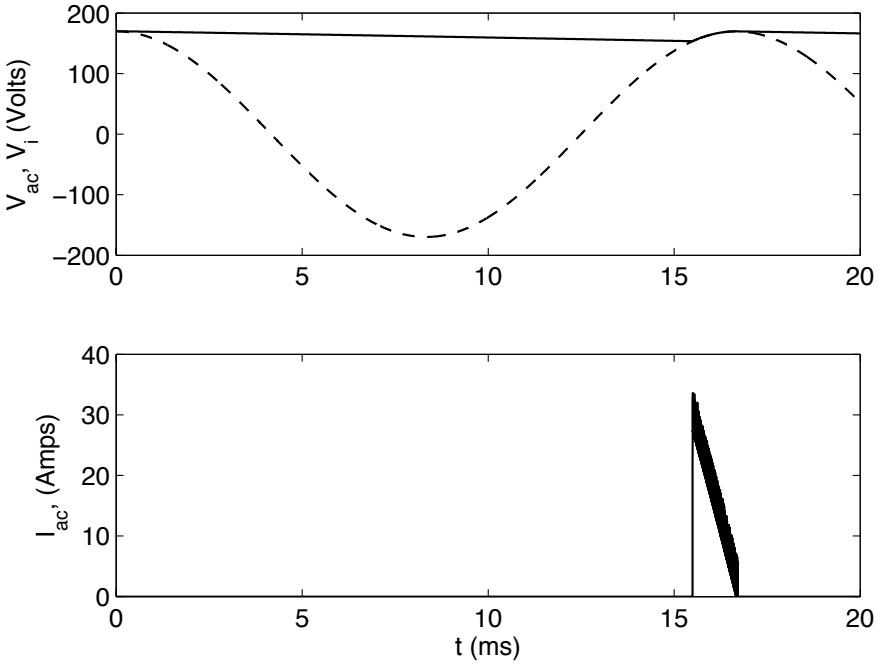


Figure 4.4: Waveforms for the AC input section. The top panel shows voltages: AC input voltage $V_{\rm ac}$ (dotted) and rectified voltage V_i (solid). The bottom panel shows input current.

a repetitive peak current of at least 40A.

Note that even though the input current is instantaneously over 30A, this circuit will operate without tripping a 10A circuit breaker or a blowing a 10A fuse. This is because most circuit protection elements respond slowly, averaging the current over a period of time, and only trip when the current overload is sustained for 10s of milliseconds — several AC cycles.

In practice, more sophisticated input circuits are used to avoid putting such a large current transient on the AC input. These circuits typically shape the AC input current to track the AC input voltage - making the overall circuit look like a resistor to the AC line. These input current conditioning circuits are often called *power-factor correction* circuits.

Also in practice, a transformer would be included at some point in the charger to isolate the negative terminal of the output (GND) from the AC neutral voltage.

Both power factor correction and isolation are required by regulation for equipement offered for sale.

4.3.2 Switching Stage and Output Filter

A switching stage consisting of FET M_1 and diode D_1 modulates the DC voltage across C_i into a series of pulses on V_x. The width of these pulses controls I_L. The operation of this *pulse-width modulator* is shown in the waveforms of Figure 4.5. The top waveform shows the 200kHz pulse train controlling the gate of FET M_1. A pulse occurs every 5μs. The width of the pulse controls the inductor current, and, indirectly, the output voltage.

The second trace shows the waveform on V_x. when $a = 1$, FET M_1 switches on, and V_x is connected to V_i. As shown in the third trace, this causes I_L to ramp up at a rate given by $\frac{dI_L}{d_t} = \frac{V_i - V_o}{L_1}$. When $a = 0$, M_1 turns off, diode D_2 clamps V_x to ground, and I_L ramps down at a rate given by $\frac{dI_L}{d_t} = \frac{-V_o}{L_1}$. If I_L reaches zero (which does not happen in the figure), D_2 turns off and $V_x = V_o$[1]. Output capacitor, C_o, filters the sawtooth inductor current providing a steady output voltage as shown in the bottom trace of the figure.

The two remaining components in Figure 4.3 are used for current sensing. The current transformer CT senses the instantaneous source-drain current in FET M_1. This measurement is used for current-mode control of the charger (Section N.4.2). Sense resistor R_{CS} is used to sense the battery current.

The buck converter topology of Figure 4.3 is very versatile and can be used for any voltage conversion application where the output voltage is lower than the input voltage. With different control, for example, the exact same power path could be used as a switching power supply or to step down an AC input to drive low-voltage LED lighting.

When the difference in voltages between input and output is large (more than about 4:1), however, the buck converter becomes costly and inefficient. This is because the switching stage must simultaneously handle the high input voltage

[1] Actually, due to stray capacitance, V_x oscillates about V_O.

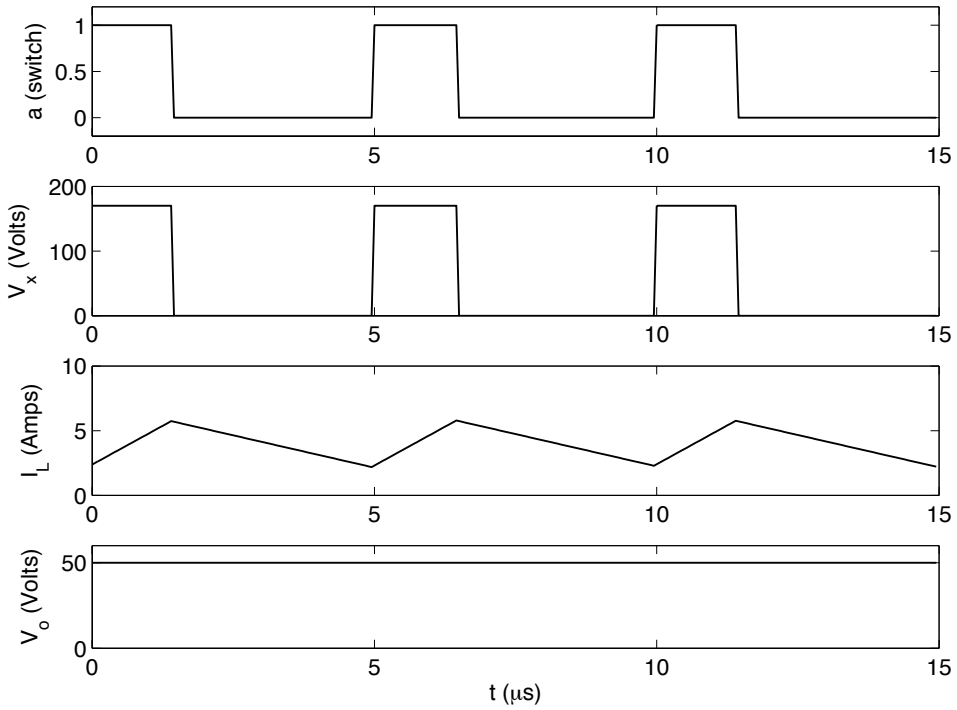


Figure 4.5: Pulse-width modulation waveforms.

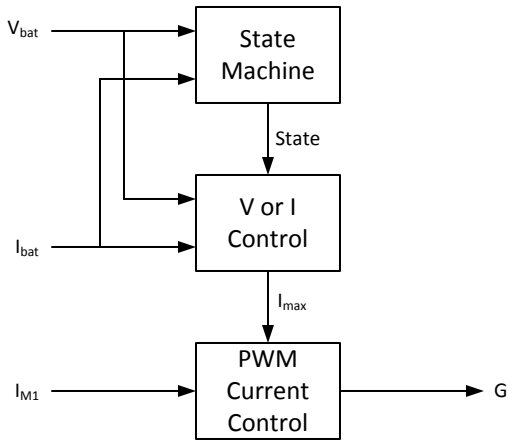


Figure 4.6: Layered controller for the battery charger

and the high output current. When a large step-down (or step-up) is required, a full-bridge or half-bridge converter, using a transformer is more efficient.

The actual buck converter is a bit more complicated than what we have shown here. It includes a *high-side driver* to provide the voltage levels needed to drive the gate G of M_1, a power supply for this high-side driver including a *start-up supply*, and a snubber on the drain of M_1 to avoid high voltage-spikes due to the inductance of the current transformer when M_1 switches off. For details of these circuits see Appendix E.

4.4 Control of the Charger

4.4.1 Layered Control

The battery charger uses a *layered* controller, illustrated in Figure 4.6. Layered control is a common theme in green electronics systems. Each level of the controller monitors its inputs, performs a control function, and sends a command to the next lower layer. Layered control partitions a complex controller in a manner that each individual layer and box is fairly simple. Thus, both the individual modules and the controller as a whole are easily understood, designed, and modified.

Our controller has three layers. The top layer is a state machine that deter-ines — by monitoring battery voltage and current — which of the four charging phases of Figure 4.2 the controller is in. The middle layer is a voltage or current regulator. Depending on the state, it either holds current constant (in

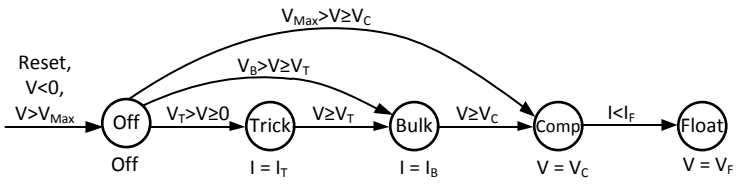


Figure 4.7: State diagram of the battery charger.

the trickle and bulk charge phases), or voltage constant (in the completion and float phases). This layer holds the appropriate variable constant by setting $I_{\max}$, the maximum inductor current. The bottom layer, in this case the PWM controller, controls the power path, in this case by driving G, the gate of FET M_1 in Figure 4.3. It monitors the current in FET $M1$ (via the current transformer) and terminates each PWM pulse when the desired maximum current, $I_{\max}$, is reached.

4.4.2 State Machine

Figure 4.7 shows the state diagram of the top level of the controller of Figure 4.6. The system has five states, *Off, Trick, Comp, Top*, and *Float* the last four of which correspond to the four phases of charging shown in Figure 4.2. Transitions between the states are made when various voltage and current thresholds are reached. The label below each state indicates the function of the controller in that state. The controller regulates current in states *Trick* and *Bulk* and regulates voltage in states *Comp* and *Float*.

The system starts in the *Off* state with no current flowing. The system will stay in the *Off* state, and will return to this state from any other state, if reverse polarity is sensed or if the battery voltage exceeds a maximum value V_{Max}.

The transition out of the *Off* state depends on the battery voltage. If the battery voltage is below threshold V_T, the controller enters the *Trick* (trickle charge) state to trickle charge the battery at a relatively low current I_T until it reaches threashold. If the battery starts above threshold — or after it has been trickle charged to this state — the controller enters the *Bulk* state to bulk charge the battery by holding the current at the bulk charge level I_B. If the battery is at a voltage at or above the completion voltage V_C, the charger enters the *Comp* state. The charger transitions from each state to its right neighbor when the appropriate voltage or current threshold is reached.

4.4.3 Buck Regulator

The second level of the controller of Figure 4.6 is a current-mode buck controller (Section N.4.2) that regulates the battery voltage or current depending on the current state. The quantity regulated is depicted under each state in Figure 4.7. For example, in the *bulk* state current is regulated so that $I = I_B$ and during the *comp* state, voltage is regulated so that $V = V_C$.

The current mode controller commands the target inductor current $I_{\max}$ as needed to achieve the regulation goal. As described in Section N.4.2, we use a simple PI controller to compute $I_{\max}$ from an error term e.

$$I_{\max} = pe + q \int e dt \tag{4.2}$$

Following the derivation in Sections N.2 and N.4.2, we choose $\omega = 62.8$ (10Hz) to allow software to close the loop with a 1kHz sample rate and a 100Hz command bandwidth, and we choose $\zeta = 1$ to provide critical damping. From these two choices, we use Equation (N.45) to derive $q = 3.95$ and (N.46) to derive $p = 0.125$.

These values of p and q are used when we are controlling voltage. When we are controlling current, we must compensate for the gain represented by the series resistance of the battery, R_B. A differential change in battery voltage Δv produces a differential change in batter current $\Delta i = \frac{\Delta v}{R_B}$. Thus, from the point of view of the controller, the battery is a stage with gain $\frac{1}{R_B}$ that converts voltage to current. To compensate, we multiply our controller gains by R_B when regulating current. Assuming the value $R_B = 0.1\Omega$ from Table 4.1 we have $q_i = qR_B = 0.395$ and $p_i = pR_B = 0.0125$.

To check our choice of p and q, we use Matlab to compute the open-loop frequency response of our controller (Figure 4.8). The figure shows a unity-gain frequency of 20.6Hz and a phase margin of 76°.

A controller with a unity-gain frequency of 20Hz may seem slow. However, this is plenty fast to charge a battery. Choosing a low ω allows us to implement the middle level of our controller in software and to use a relatively slow PWM output on our microprocessor to generate the voltage that controls $I_{\max}$.

4.4.4 PWM Current-Mode Control

The bottom layer of our controller controls the pulse-width applied to MOSFET $M1$ in Figure 4.3 to maintain the desired maximum inductor current $I_{\max}$. This layer of the controller uses a UC3843 current-mode PWM controller integrated circuit connected as shown in Figure 4.9. The UC3843 combines an oscillator, an op-amp, a comparator, a flip-flop, an output driver, an undervoltage protection circuit, and a flip-flop.

The PWM waveform is generated to regulate current by setting the flip-flop with the oscillator and resetting it with the comparator. The oscillator generates a *sawtooth* waveform with a frequency given by $f = \frac{1.64}{R_5 C_3}$. Once each cycle, the flip-flop is set, causing a rising edge on the PWM output which turns

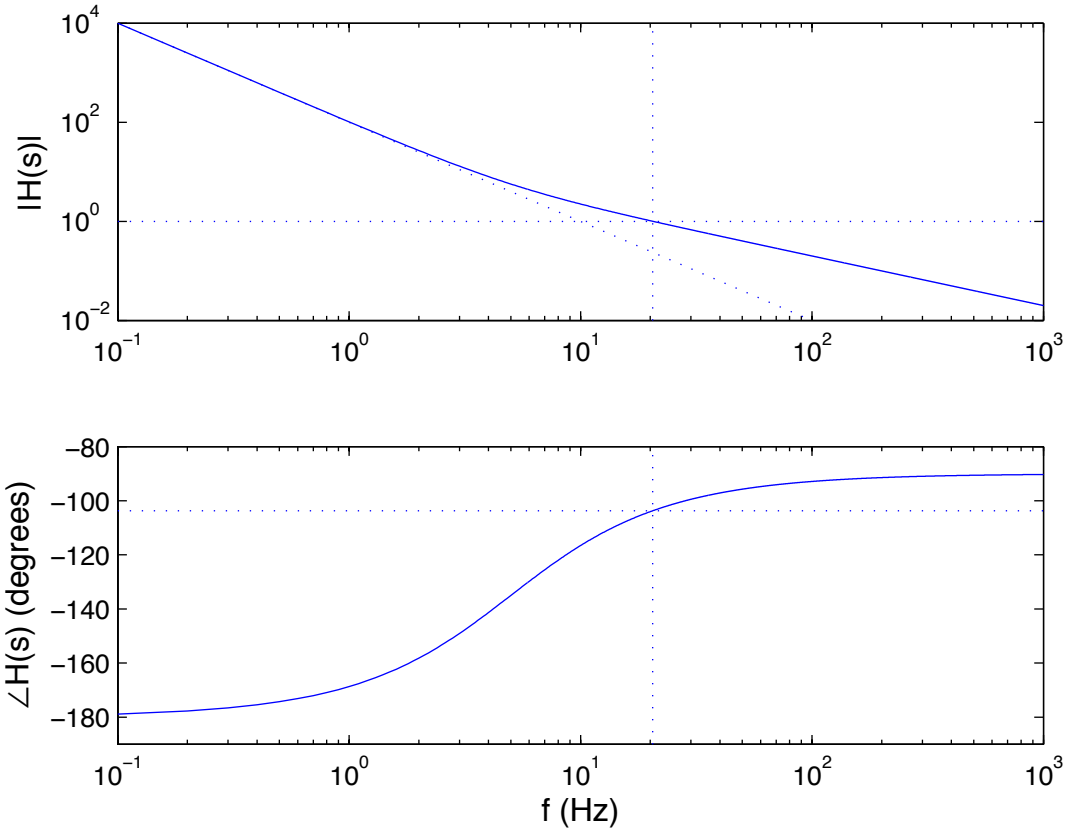


Figure 4.8: Open-loop frequency response of controller

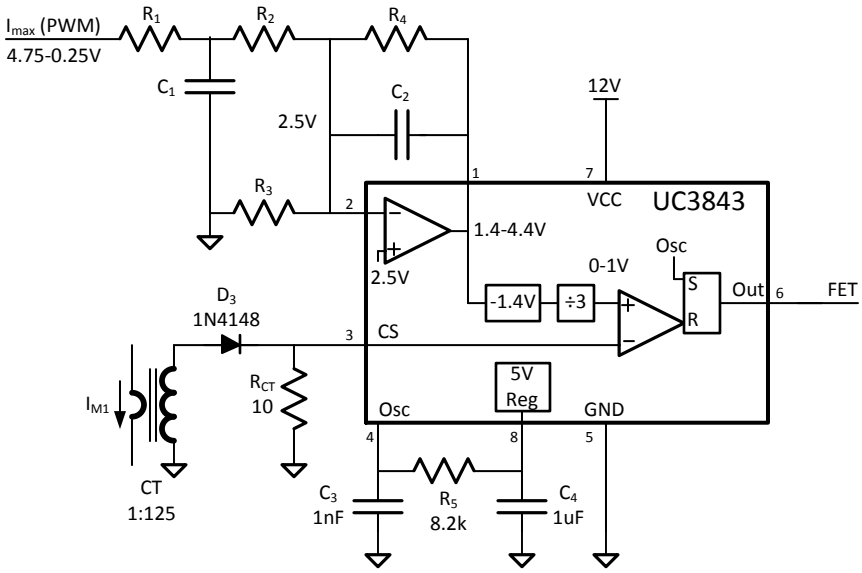


Figure 4.9: PWM current controller

on FET $M1$. As current builds in inductor $L1$, the current transformer senses this current and resistor R_{CT} generates a *current-sense* voltage. When this current sense voltage (on pin 3 of the UC3843) exceeds the value at the positive terminal of the internal comparator (which reflects $I_{\max}$) the flip-flop resets, causing a falling edge on the PWM output and limiting the inductor current to the specified $I_{\max}$.

Current Sense

To provide a maximum inductor current of 10A, the current sense circuit needs to provide a 1V signal at pin 3 of the UC3843 when the inductor current reaches about 10A. The current transformer has a turns ratio of 1:125. Thus, a 10A inductor current is reduced to a 80mA current in the secondary. A 10Ω resistor provides a signal that reaches 1V at an inductor current of 12.5A.

The diode in series with the resistor rectifies the current — providing the proper offset. Because the transformer only couples the AC component of the MOSFET current, the signal out of the current transformer is offset — going below ground when the actual MOSFET current is zero. The diode cancels this offset.

Low-pass Filter

The $I_{\max}$ signal is generated as a pulse-width modulated digital signal from one of the microcontroller's timer/counters. We use the op-amp (pins 1 and 2) of the UC3843 as a low-pass filter and level translator to convert this pulse train to a level signal on pin 1 of the UC3843. Resistors R_1 to R_4 and capacitors C_1 and C_2 configure the op-amp for this functionality[2].

The DC response of this circuit is set by the four resistors to translate an input range of 4.75-0.25V to an output range of 1.4-4.4V. (For arcane reasons, internal block of hte UC3843 apply the function $f(x) = \frac{x-1.4}{3}$ to the voltage on the output of the op-amp before applying it to the positive input of the comparator, so the 1.4-4.4V range is ultimately converted to a 0-1V range at the comparator.) We leave it as an exercise for you to calculate the values for R_1 to R_4 and C_1 to C_2. Note that for R_1 and C_1 to be an effective low-pass stage, R_1 must be at least an order of magnitude smaller than R_2.

To smooth the PWM $I_{\max}$ pulse train into a DC level, we use a two-pole low-pass filter. The first pole is provided by R_1 and C_1 and the second pole is provided by R_4 and C_2. The cutoff frequencies $f = \frac{1}{2\pi RC}$ for these two filter stages should be chosen small enough so that the AC component of the PWM signal is attenuated by at least 40dB (100×), but high enough to give a *quick* response to changes in this signal — quick compared to the speed of the middle-level control loop which has a bandwidth of 10Hz.

[2]The UC3843 provides this op-amp to implment the PI controller (the middle-level of our layered control) we implement the PI controller in the microcontroller and choose to use the op-amp as a level translator and filter.

4.5 Simulation

In this section we simulate our battery charger using Matlab. Like any model, a simulation model is tailored to the questions that need to be answered. In this section we are concerned with the overall behavior of our charger — what the general shape of the waveforms are and how our control law operates. We choose not to model the details of the switching transient — which would be required to calculate switching losses for example. As with any engineering model, the principle here is to keep things as simple as possible, but no simpler. Nothing is gained by adding unnecessary complexity to a model, and a lot of time is wasted.

We start our simulation model by writing a Matlab function `bcupdate.m` (Figure 4.10) that advances the state of our power path by one timestep. The state of our power path is to first order described by the state variables of the energy storage devices: the voltage across the input capacitor V_i, the voltage across the output capacitor V_o, and the current through the inductor, I_L. We model these three state variables as Matlab variables `v_i`, `v_o`, and `i_l` respectively. The code in Figure 4.10 computes difference variables `dv_i`, `dv_o`, and `di_l` which reflect how much each of the state variables changes in one time step, dt. For example, if the switch is closed (`(a==1)`), we compute `dv_i = -i_l*dt/c_i`. This is directly from the constituent equation of the capacitor $i = C\frac{dv}{dt}$. Which can be rewritten as $dv = \frac{idt}{C}$.

After the difference variables are computed using the old values of the state variables, we update the state variables by adding the difference variables, e,g., `v_i = v_i + dv_i`.

The last portion of the code in Figure 4.10 simulates the input stage. This code updates `v_i` to follow the AC input if `v_ac > v_i` and calculates the resulting input capacitor current. When the switch is closed, `v_i` has already been decremented to account for the inductor current, so the value computed for `i_ac` includes the inductor current.

Figure 4.11 shows the Matlab code to simulate our battery charger power path for one PWM cycle t_c. The first part of the code simulates the battery, computing the battery voltage (`v_bat`) and current (`i_bat`), and updating the charge state (`q`).

The bottom portion of the code loops over the timesteps within a PWM cycle (`steps`) calling `bcupdate` for each timestep. The switch starts out closed at the beginning of each PWM cycle (`a = 1`). Then, each step we check if the inductor current has reached the set value (`i_l >= i_a`) and if so, open the switch. Once the switch is set to the appropriate state, `bcupdate` is called to simulate one timestep. The final line of the loop logs per-timestep values to be used in plots.

What remains is to simulate the upper two layers of our three-layer control system. The code for this is shown in Figure 4.12. This code also runs once each PWM cycle. The case statement simulates the state machine, advancing to the next state when the specified conditions are met. The PI controller is then simulated by computing the error (`err`) integrating it to compute the integral

```
% bcupdate.m update vi, vo, il for one timestep

% simulate switching stage
if (a==1) % switch closed
  dv_i = -i_l*dt/c_i ;      % input capacitor
  di_l = (v_i-v_o)*dt/l ;   % inductor
else      % switch open
  dv_i = 0 ;                % input capacitor
  di_l = -v_o*dt/l ;        % inductor
end
dv_o = (i_l-(v_o-v_bat)/r_b)*dt/c_o ; % output capacitor

% update v_i, v_o, and i_l by adding deltas
v_i = v_i + dv_i ; % this gets i_l incorporated automatically
v_o = v_o + dv_o ;
i_l = max(0, i_l + di_l) ; % diode keeps i_l non negative

% simulate input rectifier
v_ac = v_acmax*cos(t*omega) ;
if (v_ac > v_i) % rectifier diode is on
  dv_i = v_ac - v_i ;
  v_i = v_ac ;
  i_ac = dv_i*c_i/dt ;
else % rectifier diode is off
  i_ac = 0 ;
end
```

Figure 4.10: Matlab simulation model to advance the state of the battery charger power path by one timestep, dt.

```
% The battery
v_bat = n_cells*(v_cell+v_q*(q/q_c - 1)) ;
i_bat = (v_o - v_bat)/r_b ;
q = q + i_bat*t_c ;
% leak off an overvoltage battery
if (v_bat > v_fl) q = q-t_c*(v_bat-v_fl)/r_fl ; end

a = 1 ;% switch a closed until current reaches i_a
for step=1:steps ;
  if(a == 1) step_a = step ; end  % remember last step with switch on
  if(i_l >= i_a) a = 0 ; end      % open switch when current reached
  bcupdate ;                      % simulate one timestep
  iaax(x) = i_a ; bclog ;         % log values for plots
end
t_a = step_a * t_c/ steps ;
```

Figure 4.11: Matlab code to simulate one PWM cycle t_c of our battery charger power path.

error (`ierr`) and computing the target current as a sum of these errors with weights `kp` and `kq` respectively. The target current is constrainted to remain in the interval $[0, i_{\text{amax}}]$.

The simulation files described in this section were used to generate the plots of Figures 4.4 and 4.5 showing the behavior of the input stage and a single cycle of the PWM modulator respectively. This simulation model was also used to simulate a complete charging cycle as shown in Figure 4.13. To accelerate the simulation, the battery capacity was modeled as $Q_C = 5 \times 10^{-2}$Coulombs — about 7 orders of magnitude lower than typical.

The plot shows the charger starting in the trickle state, charging with a trickle current of 0.5A until the voltage reaches 43V at about 8ms. At that point bulk charge starts with a constant current of 5A. There is a small overshoot as the controller settles to this value. Bulk charge continues until the voltage reaches 56.4V at 19ms. The controller enters the completion phase and holds the voltage at 56.4V while the current ramps down to 1A at about 20ms. The controller then enters the float state and sets the target voltage to 53.9V. Because of our incomplete battery model, the voltage does not drop to this level within the simulation time, so current goes to zero.

4.6 Software

The software for our battery charger is very similar to that of our energy meter (shown in Figure 2.4) and motor controller. Like the energy meter we use a real-time clock interrupt to schedule periodic current and voltage samples. Also like

```
% the state machine
switch state
  case 'trick' % trick
    mode = 'current' ;
    i_t = i_trick ;
    if(v_o > v_trick) state = 'bulk' ; end
  case 'bulk' % bulk
    mode = 'current' ;
    i_t = i_bulk ;
    if(v_o > v_c) state = 'top' ; end
  case 'top' % top
    mode = 'voltage' ;
    v_t = v_c ;
    if(i_bat < i_c) state = 'float' ; end
  case 'float' % float
    mode = 'voltage' ;
    v_t = v_float ;
end

% control  law
if(mode == 'voltage') err = v_t - v_o ;
else err = (i_t - i_bat)*r_b ;
end
ierr = ierr + err*t_c ; % integrate

i_a = max(0, min(i_amax, kp*err + kq*ierr)) ;
```

Figure 4.12: Matlab code to simulate one PWM cycle t_c of the control law.

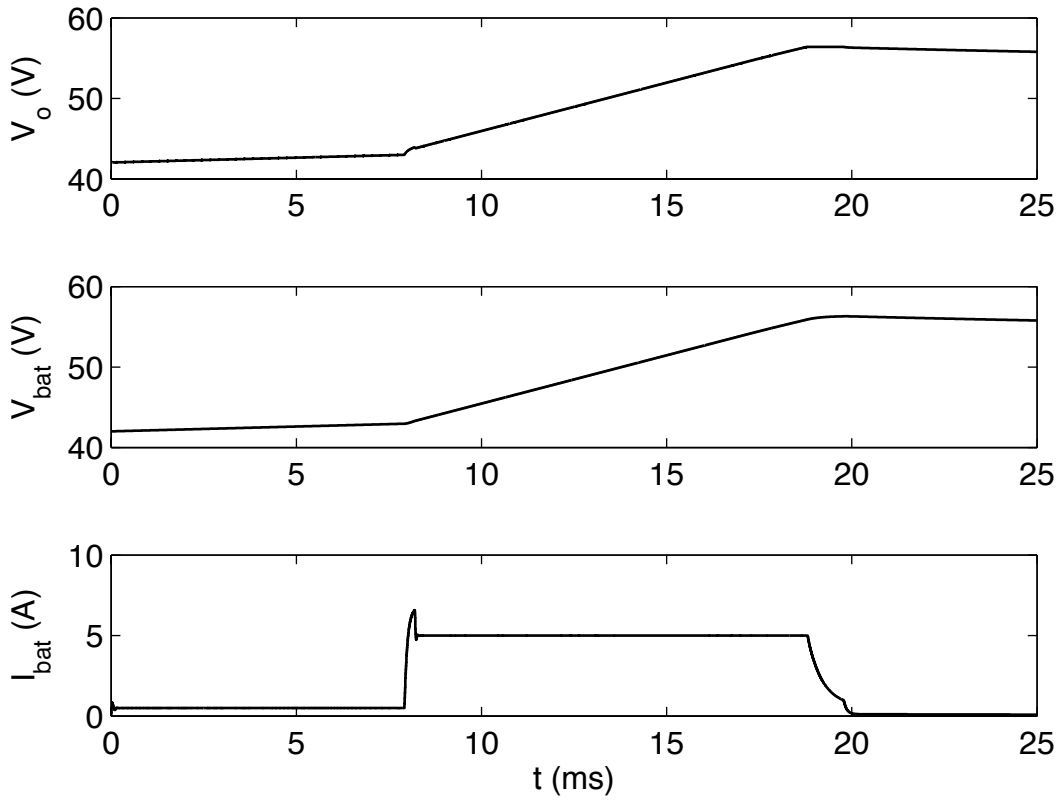


Figure 4.13: Result of simulating an entire charging cycle: trickle, bulk, completion, and float. The battery capacity Q_C has been artificially reduced by several orders of magnitude to accelerate the simulation.

the energy meter, we calibrate the raw current and voltage measurements. Like the motor controller, we include a control law block that executes periodically (once per millisecond). Once each sample period the control law block executes the upper two layers of our layered control system. The resulting control value is used to control the duty factor of the PWM output that controls the target current setting in Figure 4.9.

The control law is implemented in two subroutines. The top-level state machine is executed by routine `do_control()` (Figure 4.15).

4.7 Circuit Simulation

A circuit simulator is often used for debug and optimization of complex circuits. Like Matlab, the circuit simulator (doing a transient simulation) advances a model of the circuit from one timestep to the next. Unlike Matlab it does this from just a netlist of the circuit and easily handles complex device models and non-linearities. For example, a circuit simulation will predict the current-spike due to the finite reverse-recovery time of diode D_2 while the Matlab simulator treats this diode as an ideal rectifier.

Circuit simulations are particularly valuable for green electronic systems (and power electronics in general), because they make it easy to observe circuit failures without destroying the circuit.

With appropriate models, a circuit simulator can also predict the effects of non-ideal device behavior. The effects of the finite turn-on and turn-off times of diodes and FETs and the finite on-resistance of FETs can all be studied. With accurate models, a circuit simulator can accurately predict the losses due to these effects and hence the efficiency of a green electronic circuit.

In this section we will develop an LT-SPICE model of our battery charger. SPICE is one of the most commmonly used circuit simulators and LT-SPICE is a version that is freely available from Linear Technology.

Figure **??** shows a spice subcircuit model for the PWM controller of Figure 4.9. The first line declares a subcircuit called `cm_pwm_a` with *pins* `out, isense, imax, v12,` and `GND`. The next non-comment[3] line instantiates a copy of the UC3843 chip which is itself modeled as a subcircuit[4]. Each of the eight *pins* connected to subcircuit `xa` corresponds to one pin of the UC3843 in pin-number order.

The remaining non-comment lines of Figure 4.9 connect the passive components $R_1 - R_5$ and $C_1 - C_4$. A line beginning with `r` declares a resistor, specifies the nodes at the two ends of the resistor, and gives the resistor's value. For example, the line `r5 ref osc 8.2k` connects the 8.2kΩresistor R_5 between nodes `ref` and `osc`. Lines beginning with `c` declare capacitors in a similar manner.

The value of a component can either be specified as a numerical value like `8.2k` or as a symbolic parameter, like `r1`. The value of parameters is specified

[3] lines beginning with an asterisk "`*`" are comments and ignored by SPICE.

[4] This line instantiates an LT1247 which is pin-compatible with teh UC3843 because LT SPICE includes a model of the LT1247.

```
// the control algorithm - runs at 1kHz
void do_control() {
  // FSM
  if((voltage <= 0)||(voltage > vmax)) state = S_OFF ;
  switch(state) {
  case S_OFF: // off initially or if voltage out of range
    if((voltage > 0) && (voltage < vmax)) {init_pi() ; state = S_TRICK ;}
    OCR1B = 0xFFF ;
    break ;
  case S_TRICK: // trickle charge until at VT
    if(voltage > vt) {init_pi() ; state = S_BULK ;}
    do_pi_control(SET_I, it) ;
    break ;
  case S_BULK: // bulk charge until VOC
    if(voltage > voc) {
      tcount++ ;
      if(tcount >= S_COUNT) {tcount = 0 ; init_pi() ; state = S_TOP ;}
    } else tcount = 0 ;
    do_pi_control(SET_I, ib) ;
    break ;
  case S_TOP: // hold voltage at VOC until IOC
    if(current < ioc) {
      tcount++ ;
      if(tcount >= S_COUNT) {tcount = 0 ; init_pi() ; state = S_FLOAT ;}
    } else tcount = 0 ;
    if(current > (ib+IFUZZ)) {
      bcount++ ;
      if(bcount >= S_COUNT) {bcount = 0 ; init_pi() ; state = S_BULK ;}
    }  else bcount = 0 ;
    do_pi_control(SET_V, voc) ;
    break ;
  case S_FLOAT: // then float
    if(current > ioc+IFUZZ) {
      bcount++ ;
      if(bcount >= S_COUNT) {bcount = 0 ; init_pi() ; state = S_BULK ;}
    } else bcount = 0 ;
    do_pi_control(SET_V, vfloat) ;
    break ;
  default:
    state = S_OFF ;
    break ;
  }
}
```

Figure 4.14: "C" code that implements the charging state machine. This code executes at 1kHz (once per millisecond).

```
// do_pi_control - one time step of PI controller
void do_pi_control(int mode, int16_t target) {
  int16_t value ;

  // select current or voltage mode
  if(mode == SET_I) { value = current ; p = pi ; q = qi ;} // current mode
  else { value = voltage ; p = pv ; q = qv ; }// voltage mode

  // compute errors
  err = target - value ;
  ierr = ierr + err ; // scaled by 1/dt (factor into Q)
  if(ierr > IERR_MAX) ierr = IERR_MAX ;
  if(ierr < -IERR_MAX) ierr = -IERR_MAX ;

  // the control law
  control = p*err + q*ierr ;

  // scaling
  control = control>>SHIFT ; // to normalize - 1 = 11bits.
  control = 3000L - control ;  // offset to middle of range and invert
  if(control > 4095L) control = 4095L ; // saturate
  if(control < 0L) control = 0L ;
  OCR1B = control ;
}
```

Figure 4.15: "C" code that implements the PI controller. This code executes at 1kHz (once per millisecond).

```
.subckt cm_pwm_a out isense imax v12 GND

* instantiate UC3843 (same as LT1247)
x2 comp vfb isense osc GND out V12 ref LT1247

* oscillator
r5 ref osc 8.2k
c3 osc GND 1n
c4 ref GND 1u

* input conditioning
r1 imax r1r {r1}
r2 r1r vfb {r2}
r3 GND vfb {r3}
r4 vfb comp {r4}
c1 r1r GND {c1}
c2 vfb comp {c2}

.ends cm_pwm
```

Figure 4.16: SPICE Subcircuit Model for the PWM Controller of Figure 4.9.

in a `.param` statement elsewhere in the SPICE deck.

Figure 4.17 shows a SPICE model for the entire battery charger except for the AC input circuit. This SPICE model instantiates three subcircuits, our PWM controller from Figure **??**, `cm_pwm_a`, a current transformer, `cur_xform_a`, and the green-electronics *buck* module, `gel_buck`. The model also includes the inductor L_1 and output capacitor C_O from Figure 4.3 along with a simple battery model.

Figure 4.18 shows waveforms from a SPICE simulation of our battery charger. The figure shows six signals over two PWM cycles, a total of 10μs of simulated operation. The top trace shows the output of the buck module, the node labeled V_x in Figure 4.3. This PWM signal has an amplitude of 170V and pulse width of 1.5μs — corresponding to a duty factor of 30%.

The second trace shows I_{L1}, the inductor current which ramps from 3A to 11A when the output of the buck module is high, putting about 120V across the inductor, and then ramps back down to 3A when the output is low — with -50V across the inductor.

The next two traces show the current through D_2 and M_1 respectively. The sum of these two currents is equal to the inductor current above. The waveform shows a 30A current spike when M_1 turns on. This is due to the reverse recovery time of diode D_2. Diode D_2 takes time t_{rr} to turn off after M_1 turns on. During this period current flows directly from the supply to ground through M_1 and D_2. In a more polished charger, an inductive *snubber* on D_2 or *soft switching* of

```
* instantiate the UC3843 and components
xpwm pwm isense imax V12 GND cm_pwm_a

* current sense
xct vd vdd isense GND cur_xform_a

* buck module
xbuck vdd out pwm GND V12 gel_buck

* Inductor
L1 out ld 22uH

* output capacitor
CO ld GND 1000uF

* battery model
VL bat GND 48
RL ld bat 0.1

* supplies
VD vd GND 170
V12 V12 GND 12V
Vin imax GND 2.6

.ic v(pwm:comp)=2.8
.tran {tcy*ncy}
```

Figure 4.17: SPICE Model for the battery charger.

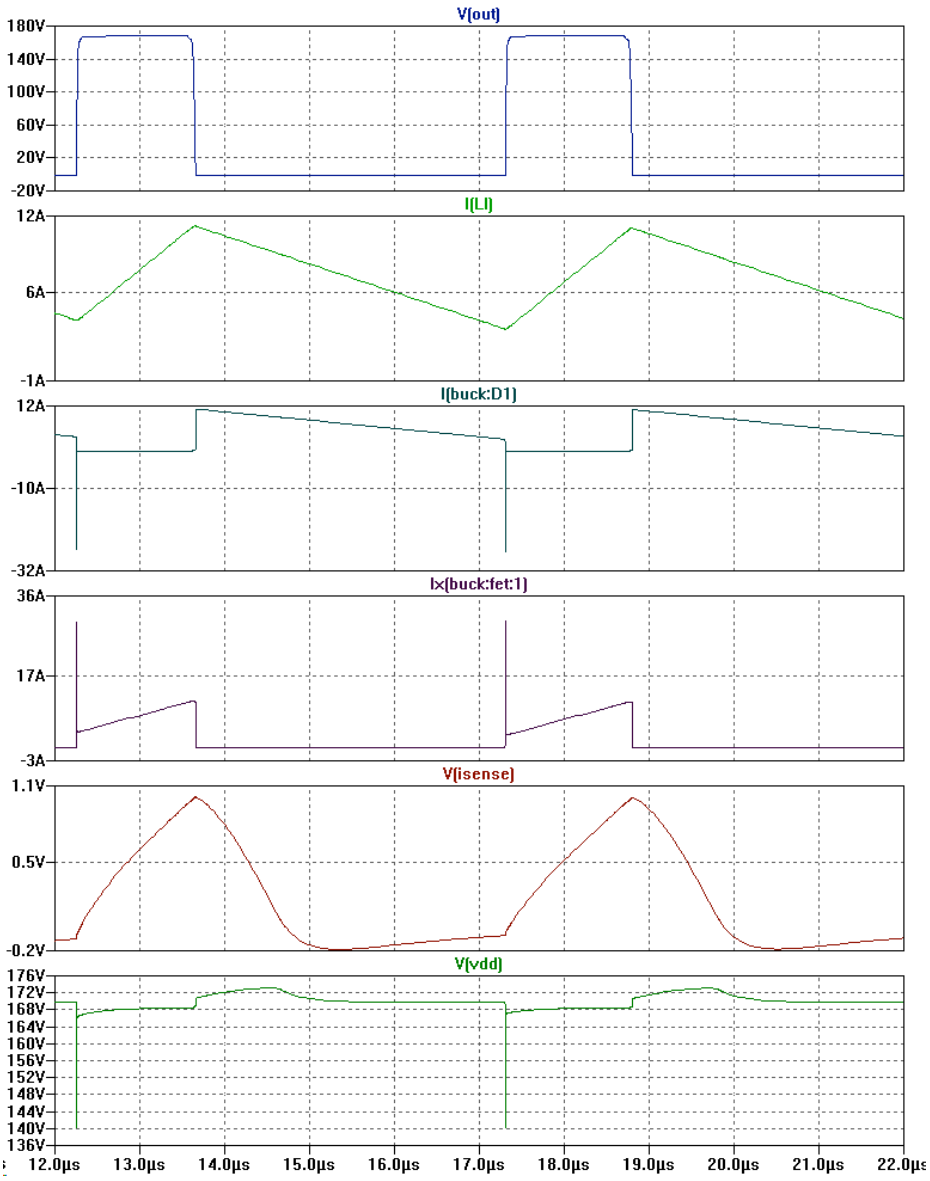


Figure 4.18: Waveforms from SPICE simulation of battery charger.

M_1 would be employed to avoid this *shoot-through* current.

Ignoring the shoot-through current for the moment, these two waveforms show that M_1 carries the inductor current during the upward ramp and D_2 carries the current during the downward ramp as expected.

The fifth waveform is the voltage across the current sense resistor on the secondary of the current transformer. This signal shows that the transformer current ramps up with the current in M_1 but then continues after M_1 turns off. This is due to the inductance of the current transformer. A capacitive snubber on the drain of M_1 in the buck module prevents the energy stored in this inductor from causing a damaging voltage spike on the drain of M_1 during turn-off.

The bottom trace of Figure 4.18 shows the voltage on the drain of M_1. This voltage drops briefly to 140V as M_1 turns on as the shoot-through current is dropped across the resistor of the drain snubber in the buck module. It then settles just slightly below the 170V input voltage as the current in M_1 ramps up. It goes a few volts above the input voltate when M_1 turns off as the energy stored in the inductance of the current transformer is absorbed by the drain snubber.

SPICE simulation is particularly useful for studying switching losses. Figure 4.19 shows the current through, voltage across, and power dissipated in MOSFET M_1 during the turn-on transient. As the figure shows, the shoot-through current that occurs before diode D_2 recovers causes the current to ramp to 30A while the voltage across the device is over 140 V. The result is an instantaneous power of over 4kW for a few nanoseconds. The current settles down to 4 A while the voltage ramps from 130V down to zero. This results in the power ramping from about 400W down to zero over about 25ns. The energy dissipated during the turn-on switching event, the area under the power curve, is measured (using a SPICE `.meas` statement) to be 13.1μJ.

The turn-off transient is shown in Figure 4.20. Here the current holds steady at 11A while the voltage ramps up to 170V over 25ns resulting in the power ramping up to 1.7kW. The energy dissipated during the turn-off event is measured to be 31.2μJ. Even though there is no shoot-through current, the turn-off loss is substantially larger than the turn-on loss because the inducotr current at turn-off is almost four times larger than the current at turn-on. The total switching energy is 44.3μJ each 200kHz cycle giving a switching power of 8.86W. The 0.9W conduction loss due to the FET's 30mΩ R_{on} is small in comparison.

Of course these predictions of switching loss, like any SPICE results, are only as good as the models used in the SPICE simulation. This simulation uses MOSFET and diode models provided by the manufacturers but ignores almost all layout parasitics — the stray inductance and capacitance due to wires and printed-circuit board traces. In practice these parasitic circuit elements can be quite important and may cause actual circuit behavior to deviate from the simulation results in significant ways. For truly accurate simulation results these circuit elements need to be accurately modeled.

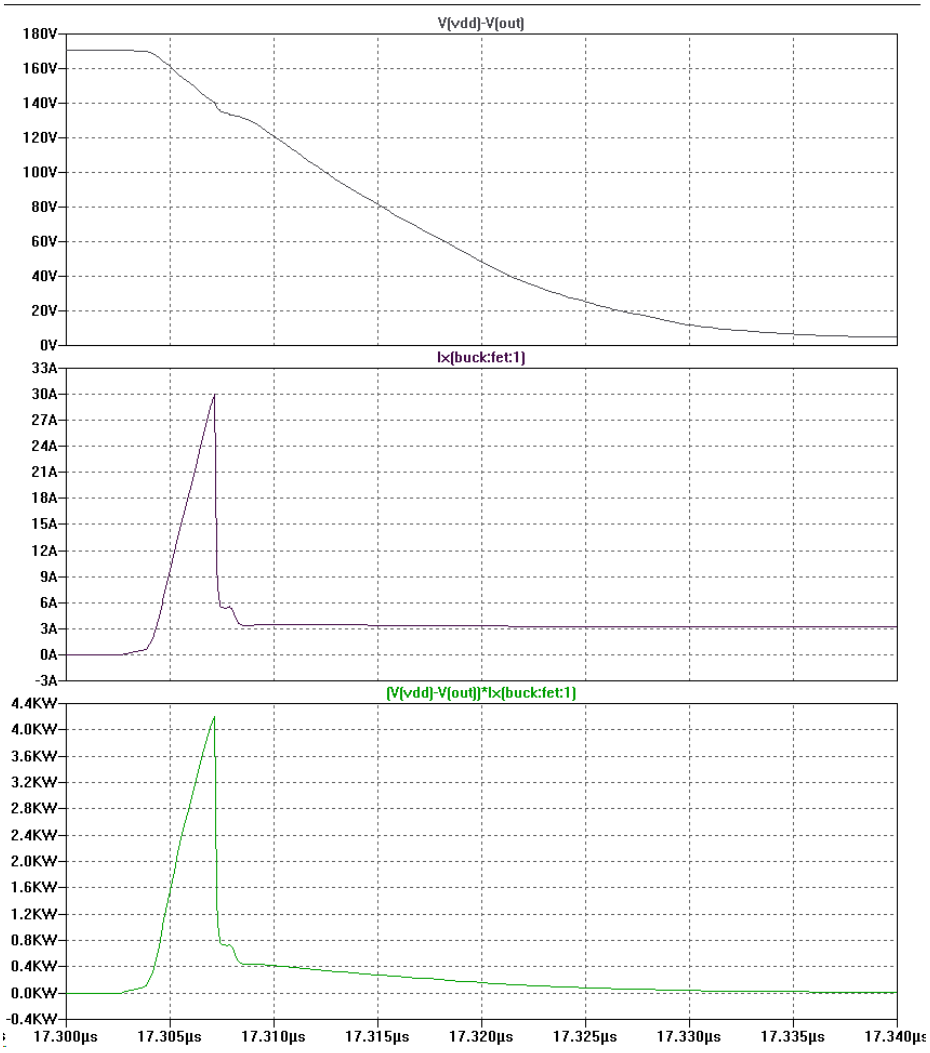


Figure 4.19: SPICE waveforms showing switching loss in M_1 during turn-on transient.

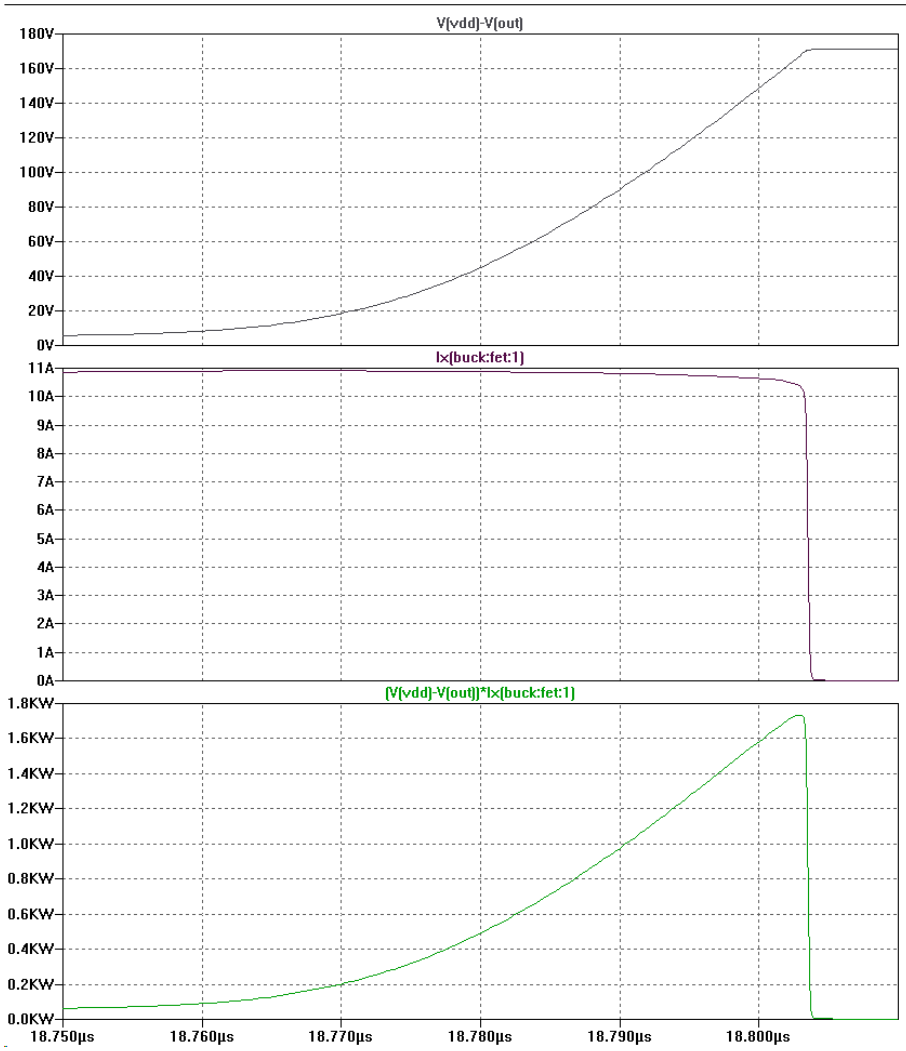


Figure 4.20: SPICE waveforms showing switching loss in M_1 during turn-off transient.

4.8 The Lab

For the laboratory you should do the following:

1. Using a lab power supply and power resistor to source and sink current from the battery, estimate the series resistance of the battery, R_B in Figure 4.1.

2. Calculate appropriate values for R_1 to R_4, C_1, and C_2 and, using these values, build the PWM controller illustrated in Figure 4.9.

3. Run a SPICE simulation of your charger using the resistor values you calculated.

4. Build and incrementally verify your PWM controller. Start by checking that the oscillator (pin 4) has a 200kHz sawtooth waveform. Next, apply a PWM waveform to the left side of R_1 and verify that the OpAmp output has a smooth DC voltage that can be varied from 1.4 to 4.4 V as the PWM duty factor is varied.

5. Assemble the power path of your charger (Figure 4.3). Use a lab power supply to supply voltage to your charger, and a resistive load in place of the battery during verification to limit damage if something goes wrong.

6. With the lab supply connected across the input capacitor verify the operation of your charger. Verify one layer of your controller at a time. Start by putting your charger in a mode where you can manually set the PWM duty factor and verify that the controller behaves as expected. Next, put the charger in a mode where you can manually specify the current or voltage the charger is to produce and verify that the charger regulates to this value. Finally put the charger into its normal operating mode and verify that it sequenes through the charging states.

   Limit your charging current to no more than 4A to avoid overheating the inductor on the component board.

7. Repeat your verification with your charger connected to the battery — but still powered from the lab supply. You can determine the battery inductance L_B at this time.

8. Finally, connect your charger to the AC line and verify that it operates as expected.

4.9 Extensions

More Accurate Battery Model: Our battery model is too simple to capture many important electrical characteristics of a lead-acid battery. Determine the shortcomings of our model and develop a more accurate model that overcomes these shortcomings.

Slope Compensation: Regulating maximum PWM current — as performed by our UC3843 can result in unstable operation when the duty factor exceeds 50%. At high duty factors the controller can alternate between a fat pulse and a skinny pulse. A technique called *slope compensation* can elminate this unstable behavior. Implement slope compensation in this lab.

Synchronous Rectification: Losses can be reduced by replacing the diode D_2 in the power path with a second MOSFET. Implement a battery charger using this techqnique, which is called *synchronous rectification*.

Soft Switching: Our battery charger has high *switching losses* because MOSFET M_1 turns on with high voltage (170V) across it and turns off with high current (up to 10A). The turn-on transient is particularly costly because diode D_2 does not turn off instantly. During the *reverse-recovery time* of D_2 M_1 shorts 170V to ground and can draw a very large amount of current until the diode stops conducting.

Redesign the converter to use *soft switching* so that the MOSFET only switches when it has either zero voltage across its terminals (ZVS) or zero current flowing through it (ZCS) or both.

Power Factor Correction: To comply with regulations, products offered for sale must have a power factor close to unity. Design a pre-regulator that comes before our buck regulator and performs power-factor correction. Most often such PFC stages are implemented as boost regulators that step-up the variable AC voltage to a higher intermediate voltage (between 200 V and 400 V) while shaping the input current to be proportional to the rectified AC volage. A bulk capacitor on this intermediate voltage stores energy across the null in the AC input with this design — replacing V_i.

Isolation: In many applications our charger must also be completely isolated from the AC line. By adding a transformer to our buck converter we can provide this isolation. Such an organization is called a full-bridge or half-bridge converter — depending on how the transformer is connected. Design such an isolated charger.

Highly-efficient charger: Because of losses due to rectification, switching, and conduction most chargers are not particularly efficient — perhaps averaging 80 to 90% efficiency. Design a charger with the goal of making it as efficient as possible. A 97% efficient charger shoudl be achievable and higher efficiencies may be possible.

Battery Management and Equalization: Our 48 V battery is composed of 24 2 V cells. Over time the voltages across these cells may become imbalanced. While we don't have access to the individual cells, we can monitor the voltage across each 12V group of six cells to determine when imbalance occurs.

Design a *battery management* system that monitors the voltage across each group of cells and turns on switches that connect *bleed resistors* across cell groups that have a higher voltage than their peers. Integrate this system with teh charger so this *equalization* occurs near the end of charging.

Your battery management system might also display the current state of the battery and maintain a history of its charging and discharging.

Charge Reistribution: Design a battery management system that equalizes cells without losses. In place of a bleed resistor, use a switched inductor to bypass current around overcharged cells during charging and around undercharged cells during discharge to equalize voltages.

Chapter 5

Build a Photovoltaic Controller

Photovoltaic cells are a great source of renewable energy. With the sun directly overhead, there is about 1kW of solar energy (energetic photons) per square meter of area. A photovoltaic panel converts this solar energy into electricity. A photovoltaic controller optimizes the operating point of a panel, or string of panels, for peak efficiency and converts the raw electrical energy out of the panel into a useful level.

This chapter describes a laboratory where you will build a photovoltaic controller that controls a single panel and optimizes its operating point driving either a resistive load or charging a battery pack. The the course of this laboratory you will learn:

Photovoltaic Cells: The basic electrical characteristics of photovoltaic cells and panels.

Optimization: You will learn how to use dithering and a gradient search to optimize a convex function.

Boost Converter: You will become familiar with details of the *boost* converter, introduced in Section 3.3, as it is applied to converter a lower voltage (from the solar panel) to a higher voltage (to charge the battery).

5.1 Photovoltaic Panel

For our laboratory we will be using a CS6P-235PX solar panel manufactured by Canadian Solar. The I-V curves for this panel are shown in Figure 5.1 and the key parameter for the panel are shown in Table 5.1. At full irradiance, the panel has a short-circuit current (the current that flows if you short its leads together) of $I_{SC} = 8.46$A. And this current falls off linearly as irradiance is reduced. The panel has an open circuit voltage of (the voltage that appears

Symbol	Value	Units	Description
N_C	60		Number of Cells
V_{OC}	36.9	V	Open Circuit Voltage
I_{SC}	8.46	A	Short-Circuit Current
V_{MP}	29.8	V	Maximum Power Voltage
I_{MP}	7.90	A	Maximum Power Current
eta	14.6	%	Efficiency

Table 5.1: Key parameters of the CS6P-235PX solar panel at 1kW/m^2 irradiance and 25°C.

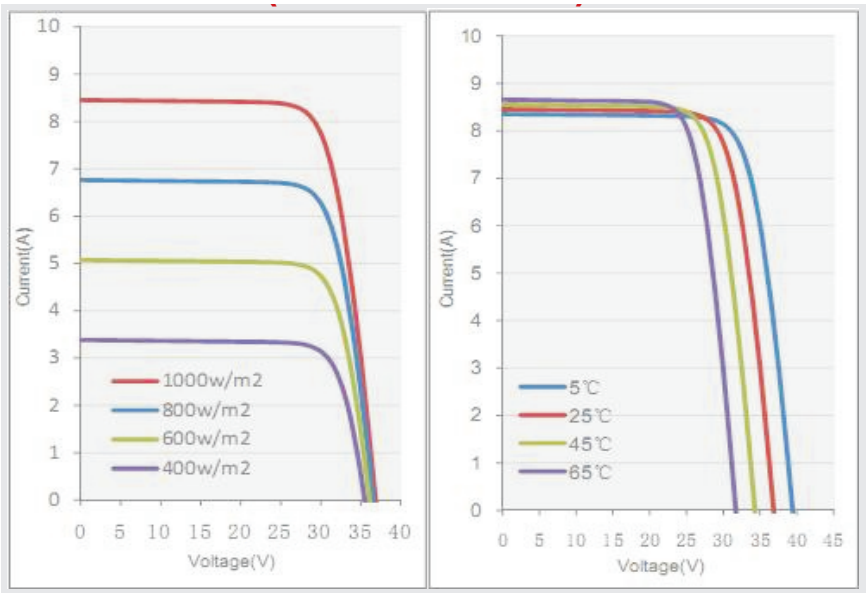


Figure 5.1: IV curves for the CS6P-235 235W Solar Panel.

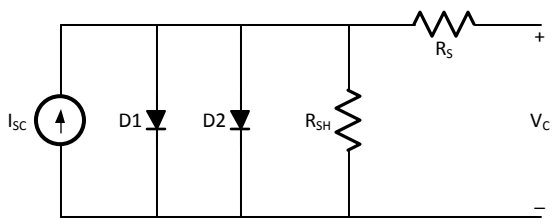


Figure 5.2: Model of a single photovoltaic cell.

across its terminals with no current flowing) of $V_{OC} = 36.9$V. The open circuit voltage is reduced slightly as irradiance is reduced but is a strong function of temperature.

The current is initially very flat, holding relatively steady as voltage is increased until about 30V. At that point the current is drops exponentially with voltage. The peak power point, the point where the power (the product of voltage and current) is a maximum is at the knee of this curve.

Figure 5.2 shows a circuit model of a single photovoltaic cell. We model the cell as a current source — that represents the photocurrent generated by energetic photons — in parallel with two diodes. The diodes have different characteristics, one is nearly ideal with an emissivity constant $n = 1$ and $I_{\text{Sat}} = 3\times10^{-10}$ while the other represents losses due to recombination and other effects and hence has an emissivity constant of $n = 3$ and $I_{\text{Sat}} = 5 \times 10^{-4}$. The shunt resistor R_{SH} sets the slope of the horizontal part of the current curve at low voltages. The series resistor R_S sets the slope of the vertical part of the current curve at low currents.

5.2 Power Path

Figure 5.3 shows the power path of the PV controller, a simple *boost* converter (Section 3.3). The PV panel provides energy at its operating voltage V_{PV} and current I_{PV}, which is sensed across resistor R_S. Input capacitor C_i stores this energy to smooth the inductor ripple current. Current in inductor L_1 ramps up when MOSFET M_1 is on and ramps down — transferring its energy to the higher load voltage V_L — when M_2 is on. Output capacitor C_O smooths the output load current.

As derived in Section 3.3 the duty factor D of M_1 determines the voltage ratio.

$$\frac{V_L}{V_{PV}} = \frac{1}{1-D} \tag{5.1}$$

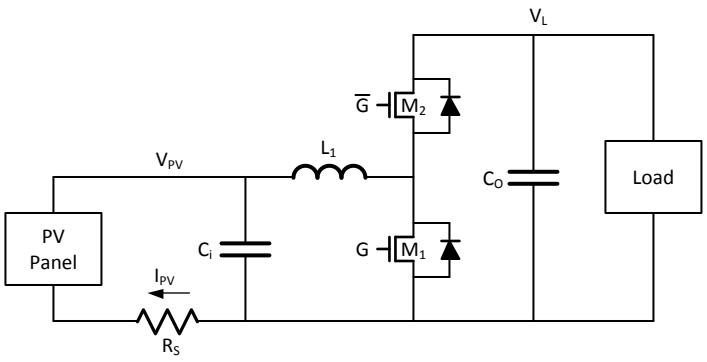


Figure 5.3: Power Path of the PV Controller.

5.3 Controller

To find the peak-power point of the photovoltaic panel, our controller uses a technique called *gradient search* or *hillclimbing*. Starting from the current operating point, we vary the operating voltage of the PV panel slightly (this is called *dithering*) and measure the power at the new operating point. If the power goes up, we accept the new operating point. If the power goes down, we go back to our old operating point and try searching in the other direction.

While we could search by varying V_{PV} or I_{PV} and then, using a layered controller, determine the duty factor D that corresponds to the setpoint, it is simpler to remove the indirection and perform gradient search directly on duty factor.

Our controller is summarized in the pseudo-code of Figure 5.4. Starting from an initial point, the loop repeatedly dithers from the current operating point, waits for the system to stabilize, and then remeasures power. If the new power is an improvement, the new operating point is accepted and the search continues. Otherwise the controller returns to the old operating point and reverses the direction of search.

5.4 Matlab Simulation

To verify the design of our photovoltaic controller — both the power path and the controller — we develop a Matlab model of the solar panel, the power path, and the controller.

Figure 5.5 shows the Matlab model of our solar panel. This file is a Matlab function that returns the cell current i as a function of cell voltage v, the degree of illumination m, and the temperature in degrees Kelvin *temp*. This model

```
df = INITIAL_DUTY_FACTOR ;
run_until_stable() ;
power = measure_power() ;
delta_df = DELTA_DF ;
old_df = df ;
old_power = power ;
do {
  df = df + delta_df ;
  run_until_stable() ;
  power = measure_power() ;
  if(power > old_power) {
    old_power = power ;
    old_df = df ;
  } else {
    df = old_df ;
    delta_df = - delta_df ;
  }
}
```

Figure 5.4: Pseudo-code for a photovoltaic peak-power tracker that operates by performing gradient search on duty factor.

omits the series resistor R_S to avoid the need to solve for the internal voltage. The main equation is in the last line of the function and has four terms that correspond to the four elements of Figure 5.2.

Matlab code that simulates one timestep of operation of the power path of Figure 5.3 is shown in Figure 5.6. This code computes the input and output current, `ipv` and `iout`, and then integrates the state of the three energy storage elements `vin`, `vout`, and `il`. An `if` statement conditions the state of `vout` and `il` on the state of switch M_1.

Figure 5.7 shows the Matlab code that implements the gradient search. In the simulation, this code is run each 40 PWM cycles to compute a new value for duty factor `df`. The code implements a simple gradient search similar to that shown in Figure 5.4.

The result of running the Matlab simulation for three cycles is shown in Figure 5.8. The simulation shows a state where the controller has a duty factor of $D = 0.2$ and is stepping up a 35V v_{in} by $\frac{1}{1-D} = 1.25$ to a 44V v_{out}. Each cycle, inductor current i_L ramps up from 4A to 7A for 2μs while M_1 is closed, and then ramps down from 7A to 4A over 8μs while M_2 is closed. This gives an average inductor current of 5.5A. The ripple on v_{in} and v_{out} are the result in integrating this sawtooth inductor current on the input and output capacitors, C_i and C_O (10μF each).

Figure 5.9 shows a Matlab simulation of our maximum-power-point tracking algorithm. The simulation starts with full irradiance and a duty factor of $D =$

```
function [i] = pv(v,m,temp)
% model of a PV cell
% i is current
% v is voltage
% m is illumination - 0-dark, 1-full sunlight
% temp is temperature in degrees K

% ignore Rs and find current from diodes and Rsh
rsh = 300 ;  % shunt resistance
is1 = 5e-4 ; % diode 1 parameters
dn1 = 3 ;
is2 = 3e-10 ; % diode 2 parameters
dn2 = 1 ;
cells = 60 ; % number of cells in series
k = 1.38E-23 ;
q = 1.6E-19 ;
ve = k*temp/q ;

igen = 8.46*m ;

i = igen - v/rsh - is1*(exp(v/(cells*dn1*ve))-1) - is2*(exp(v/(cells*dn2*ve))-1) ;
```

Figure 5.5: Matlab model of our solar panel.

```
% simulate one step, dt of PV power path
% if a switch is down, otherwise switch is up
ipv = pv(vin,illum) ;
iout = vout/rout ;
if(a)
  dil = vin*dt/l1 ;
  dvout = -iout*dt/cout ;
else
  dil = (vin-vout)*dt/l1 ;
  dvout = (il - iout)*dt/cout ;
end
vin = vin + (ipv - il)*dt/cin ;
vout = vout + dvout ;
il = il + dil ;
```

Figure 5.6: Matlab code to advance the state of the power-path of Figure 5.3 one time step.

```
p = measure_power ;
if(p >= pold)
  pold = p ;
  dfold = df ;
  df = df + ddf ;
else
  ddf = -ddf ;
  df = dfold ;
  pold= 0 ;
end
```

Figure 5.7: Matlab implementation of the gradient-search controller.

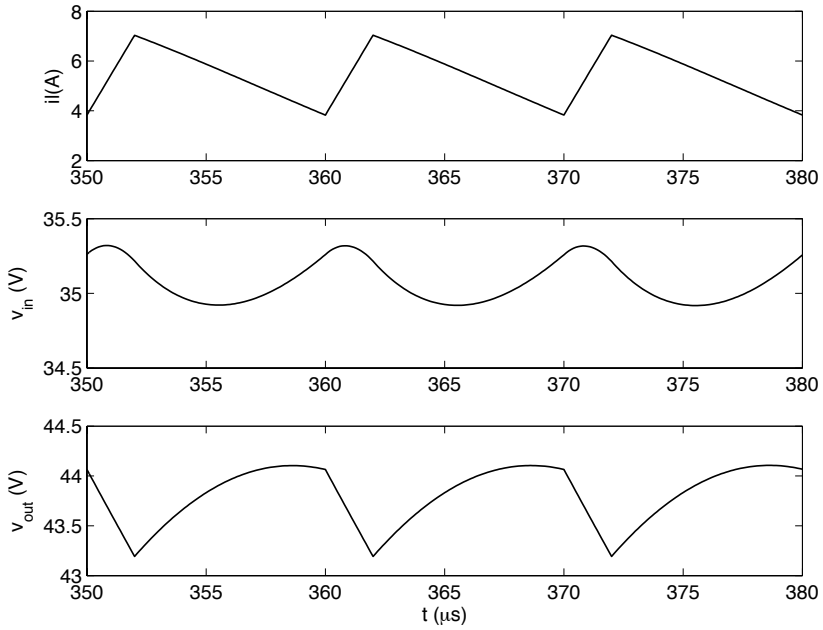


Figure 5.8: Matlab simulation of 3 cycles of the PV controller. Traces are i_L, v_{in}, and v_{out}.

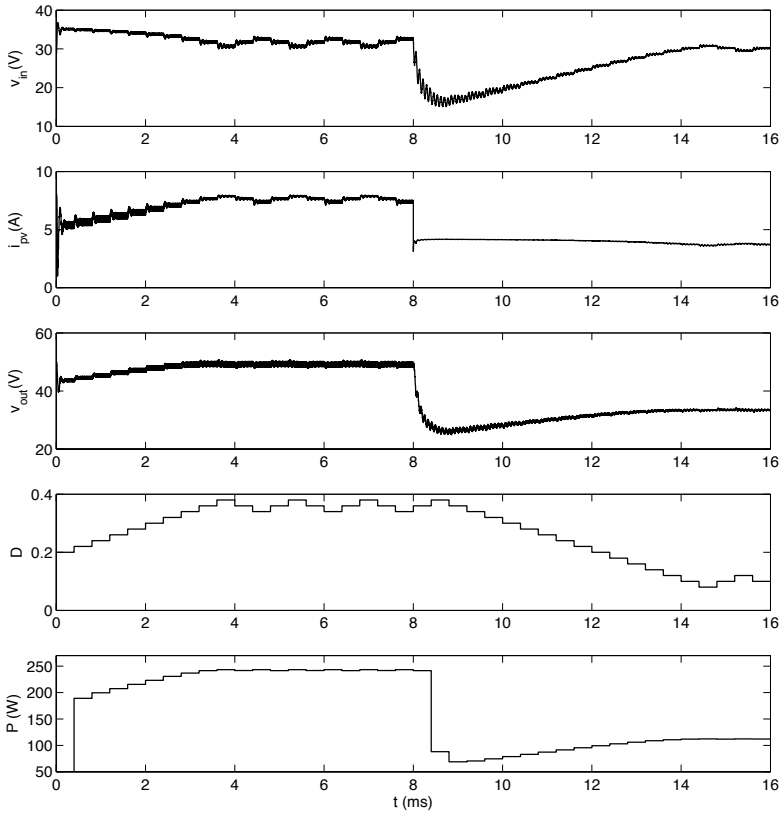


Figure 5.9: Matlab simulation of the maximum-power-point algorithm. Traces are $v_{\rm in}$, $i_{\rm pv}$, $v_{\rm out}$, duty factor D, and power P.

```
* SPICE model of the power path
x1 an ca pv_panel

RS ca 0 0.01
CI an 0 {ci}
L1 an x 22uH

xf2 l g2 x irlb3036pbf
xf1 x g1 0 irlb3036pbf

CO l 0 {co}
RL l 0 10
```

Figure 5.10: SPICE model of the power path of Figure 5.3.

0.2. Half-way through the simulation, at 8ms, irradiance is abruptly reduced to 50%.

During the first 4ms of the simulation, D ramps from its initial value of 0.2 which yeilds a power of about 190W to a value of 0.36 where power reaches a peak of about 240W. The simulation spends the next 4ms dithering about the optimal value with duty factor varying between 0.34 and 0.38. Because the power curve is fairly smooth in this area, power holds steady at about 240W.

At 8ms the irradiance is abruptly cut in half. This cuts the panel current roughly in half - from about 8A to about 4A and both the input and output voltage fall in response. This puts the panel at a very non-optimal operating point — due to its low input voltage — about 18V.

The peak-power tracking algorithm responds to the transient by ramping D down to 0.1 at about 14ms. This ramps the input voltage back up to about 30V giving a power P of about 110W. The controller then dithers about this new optimal point with D varying between 0.08 and 0.12.

5.5 SPICE Simulation

Matlab is useful for understanding the high-level operation of the PV controller and for simulating the control algorithm along with the circuit. SPICE, on the other hand, is useful for exploring the details of circuit operation including switching losses, transient currents, and stresses on components.

A SPICE model of the power path of Figure 5.3, including the photovoltaic panel and a resistive load, is shown in Figure 5.10. Subcircuit `x1` is the photovoltaic panel and subcircuits `xf1` and `xf2` are M_2 and M_1. The model also includes the inductor `L1` input capacitor `CI`, output capacitor, `CO`, load resistor `RL`, and sense resistor `RS`.

Figure 5.11 shows a SPICE simulation of two cycles of the power path of Figure 5.10. The waveforms closely match those of Figure 5.8 although the

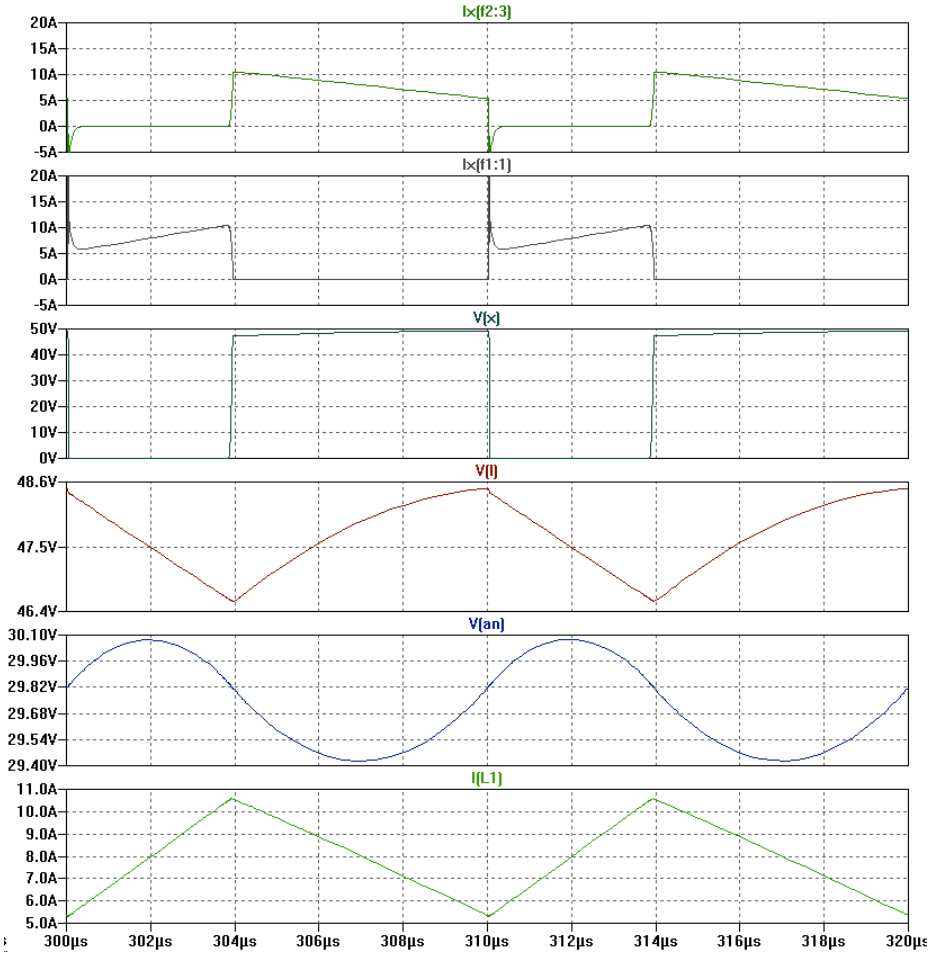


Figure 5.11: Waveforms from SPICE simulation of model of Figure 5.10. From top to bottom, the traces show i_{M2}, i_{M1}, v_x, $v_{\rm L}$, v_{rmPV}, and i_{L1}.

operating point here has a higher duty factor (0.4). During the first portion of each cycle inductor, v_X is zero and inductor current ramps up through M_1. During the second portion of each cycle, v_X is equal to v_L and inductor current ramps down through M_2.

The top two traces show a large current transient when M_1 turns on. This effect is due to the reverse recovery time of the body diode in M_2. When M_2 turns off, the body diode of M_2 carries the inductor current until M_1 turns on. When M_1 turns on, it sinks the inductor current to ground and reverse biases the body diode of M_2. However, it takes several nanoseconds for this diode to turn off. During these few nanoseconds, M_1 shorts v_L to ground and draws a very large current.

Figure 5.12 shows a close-up of the M_1 turn-on transient. The figure shows that because of the slow diode reverse-recovery, current in M_1 (and M_2) ramps to over 150A while the voltage across M_1 is still over 44V. This *shoot-through* current occurs because M_1 and the body diode of M_2 are both *on* at the same time — effectively shorting v_L to ground. At the peak of the current transient, the instantaneous power dissipated in M_1 reaches nearly 7kW. Fortunately this lasts for only a few nanoseconds. The total energy of this transient E_{on}, the area under the power curve, is 29μJ (as measured with a SPICE `.measure` statement. With a switching frequency of 100kHz, this gives a switching loss due to the M_1-on transient of 2.9W. There is also a 7.4μJ loss in M_2.

The M_2 turn-on-transient, shown in Figure 5.13 is much less dramatic. As M_1 turns off, and the current in M_1 (bottom trace) ramps down, the body diode of M_2 turns on instantly and picks up the remainder of the inductor current i_L. Thus, as i_{M1} ramps from 11A to 0A i_{M2} ramps from 0A to 11A. Voltage v_x swings from 0V to 48V during this 70ns period. The P_{M1} peaks at about 180W and P_{M2} peaks at about 90W. The energy during this transient E_{off} is 9.2μJ for M_1 and 4.6μJ for M_2 for a total of 13.8μJ or 1.3W at 100kHz.

The total switching losses over both M_1 and M_2 over both the on transition of Figure 5.12 and the off transition of Figure 5.13 is 49μJ or 4.9W at 100kHz. These switching losses can be largely eliminated by using *soft-switching* techniques. There are also conduction losses in the transistors and inductors that we are not considering here.

Of course, the results of this SPICE simulation are only as good as the models that went into it. This simulation used manufacturers models for the MOSFETs. However parasitic inductance due to board layout was ignored. If this inductance is significant, it can make a very large difference to the switching losses. A truly accurate simulation would include parasitic circuit elements calculated from layout geometry and validated via experiment.

5.6 The Laboratory

For the laboratory you are to do the following.

1. Build a Matlab model of the PV controller and simulate operation of the

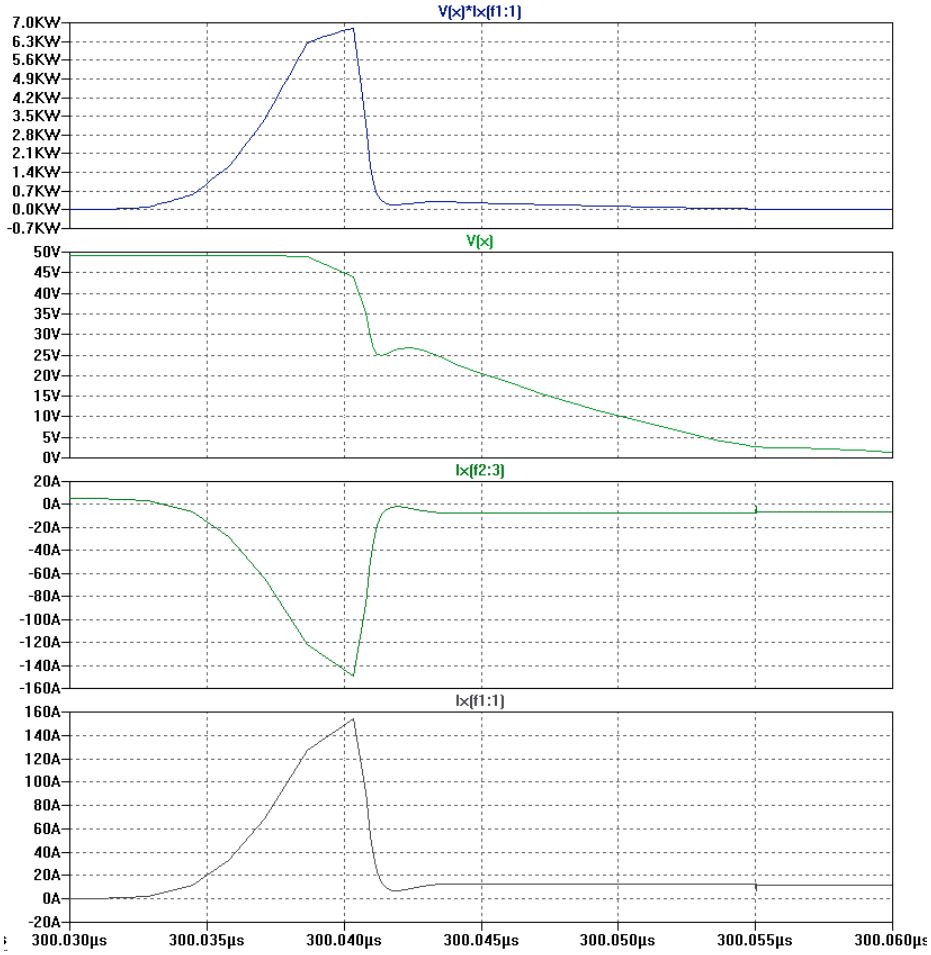


Figure 5.12: Close-up of the M_1 turn-on transient. Traces are P_{M1}, v_x (voltage at the drain of M_1), i_{M2}, and i_{M1}.

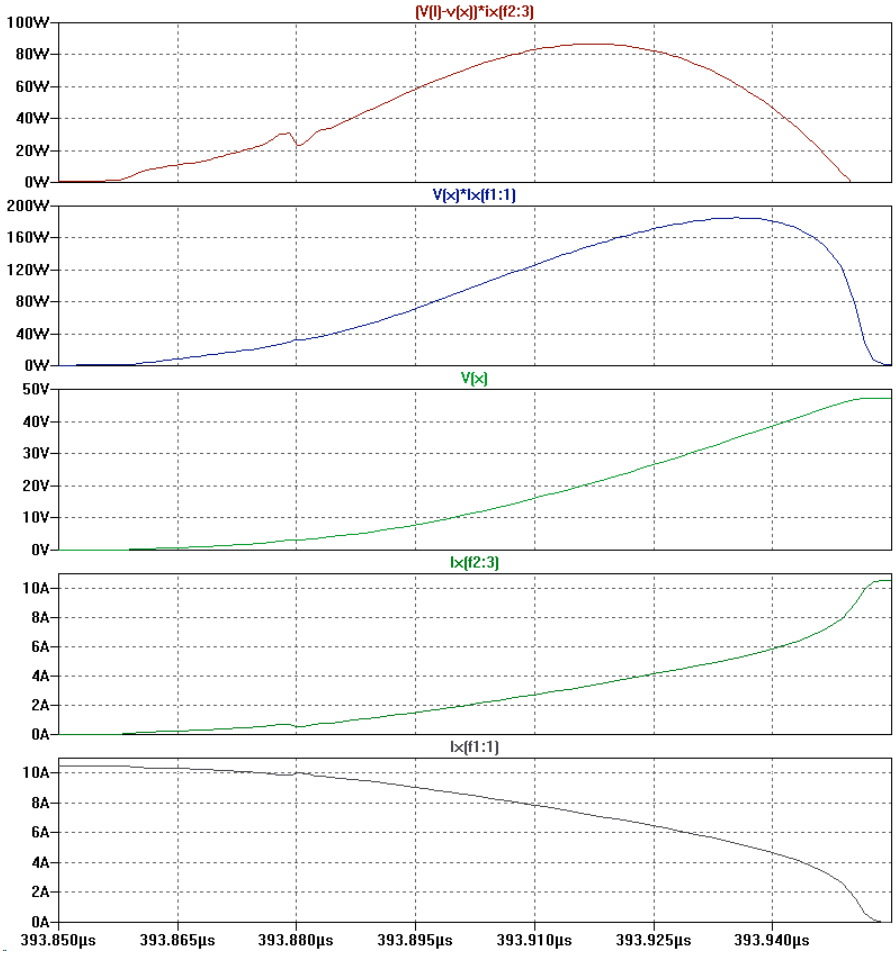


Figure 5.13: Close-up of the M_2 turn-on transient. Traces are P_{M2}, P_{M1}, v_x (voltage at the drain of M_1), i_{M2}, and i_{M1}.

controller including finding the maximum-power point and dealing with transients in irradiance and temperature.

2. Buld a SPICE model of the PV controller and simulate five cycles of operation in a steady state conditions. Measure the energy of the turn-on and turn-off transients.

3. Wire up the power path of Figure 5.3 and control it using a 100kHz PWM signal from teh processor module. Write the embedded code to perform the maximum-power point tracking.

4. Test your prototype PV controller using a lab power supply set for 30V maximum voltage and 3A maximum current to simulate a PV panel (albeit with a rather *square* I-V curve. Your maximum-power point tracker should find the pulse width needed to stay at this maximum power point.

5. Test your prototype PV controller using an actual solar panel. Verify that it finds a new maximum-power point when you shade half of your panel with a piece of cardboard.

6. (Optional) Measure the efficiency of your PV controller by placing energy meters before and after the controller.

5.7 Extensions

Inverter: Modify your PV controller to include an inverting output stage to interface with a power line. The boost stage of Figure 5.3 should be modified to produce an intermediate voltage that is larger than the maximum AC voltage (170V for 110V AC and 340V for 220V AC). A full-bridge stage should then be added to synthesize a sine wave via pulse-width modulation. An LC output filter is needed to filter out the PWM frequency noise.

Panel Balancing: If two PV panels (or two PV cells) connected in a string are not identical or do not have identical irradiance, it will not be possible to simultaneously find the maximum-power point for both. The current i_{PV} which is common to all of the panels in a string will be a compromise between the maximum-power current of the two panels.

Using our Matlab model for a solar panel, quantify the amount of power lost when two panels must operate at the same current point when one has an full irradiance (1000W/m^2) and the other has half irradiance (500W/m^2) compared to the power when both are operated at their individual maximum-power points.

The current of each panel (or cell) in a string can be made individually adjustable by using a pulse-width-modulated inductor to losslessly bypass current around panels requiring a lower current. Design such a panel bypass circuit and simulate it in SPICE on a two-panel configuration.

(Optional) Implement your circuit and demonstrate it on a real two panel configuration with one panel half shaded.

Variable-Step Gradient Search: Our gradient-search algorithm has a fixed step size. This causes it to take a long time to track a large transient and at the same time results in a fairly large dithering region once we are converged on the maximum-power point. Develop a better search algorithm that varies the step size for faster tracking of transients and smaller dithering intervals when converged.

Soft Switching: As illustrated in Figure 5.12 and Figure 5.13, our PV controller power path performs *hard* switching, resulting in significant switching losses. By switching our MOSFETs only when the current through them is zero (zero-current switching or ZCS), or the voltage across them is zero (zero-voltage switchign or ZVS), or both, we can reduce the switching losses to zero.

Design a power path that uses soft switching to eliminate switching losses and simulate this power path using SPICE. Measure the total losses — switching, MOSFET conduction, and inductor conduction and core losses — for both the original configuration and your new design.

(Optional) Implement your design and demonstrate it operating with a PV panel. Measure its efficiency using an energy meter before and after the PV controller.

Stability Analysis: We use a 10μF film capactior for C_i because it reduces voltage ripple due to the inductor current ripple to acceptable levels. It turns out that if this capacitor is made very large, say 1mF, the PV controller will oscillate. Develop a model of the PV controller that allows you to study its stability. Using your model, predict the value of capacitance that causes the controller to oscillate. Develop a strategy to stabilize the controller when using a large capacitor.

Appendix A

Safety

If a few basic rules are followed a power electronics laboratory exercise can be completed in a safe manner — with little risk of injury or damage to equipment. However, because of the high voltages, currents and power levels involved deviation from the rules or careless activity has the potential to cause harm.

This appendix outlines the risks associated with a power electronics laboratory and a set of rules and procedures to manage these risks without incident.

A.1 Risks

The risks associated with a power electronics lab an inherent to working with high-voltage, high-current electronics and include:

Electric Shock: Some conductors in the circuits you will build will be at high voltages (up to several hundred volts). Touching these conductors with your bare hands or with a conductive tool can result in a severe electric shock.

The capacitors in the circuits you will build store many Joules of energy and can retain this energy even when the circuit is powered down. Do not assume it is safe to touch a circuit just because the power is turned off. Always make sure high energy capacitors have been drained.

Burns: Some components in power electronic systems dissipate large amounts of power in the form of waste heat. Because of finite thermal resistance, these components get hot — often as much as 100-200°C. It can take many minutes after a circuit is powered down for hot components to cool off. Touching a hot component can result in a severe burn.

During failures of power circuits wires and other conductors may get so hot that they vaporize. If your hand — or other body part — is near these conductors when they vaporize you can suffer a plasma burn.

Projectiles: An incorrectly operating power circuit can overheat a component causing it to explode and eject fragments with high velocity. Power semiconductors, power resistors, capacitors, and inductors can all exhibit this behavior if they are overloaded. If a part of your body gets in the path of a fragment of an exploding component you can suffer an injury. Your eyes are particularly vulnerable.

A.2 Rules and Procedures

To manage the risks associated with a power electronics lab, we follow a few simple rules and procedures. Please take these seriously — ignoring the rules or taking shortcuts can result in someone getting hurt — possibly you.

Don't work alone: A minor injury can become life threatening if there is no one present to lend aid and call for help. Never work in the lab by yourself on high-power circuits.

For EE26N we carry this one step further by opening the lab only when it is supervised by an instructor or section leader.

Don't work if you are not at your best: If you are tired, sick, distracted by a situation outside the lab, inebriated, or otherwise operating at less than 100% of your mental capacity, don't work in the lab. Accidents happen when people aren't completely focused on what they are doing with all of their faculties.

Don't "operate" on a powered circuit: Do not attempt to rewire a circuit, apply or move a prove, or any other manipulation of a power circuit while it is powered on. With the lid closed while power is applied you shouldn't be able to do this anyway. Moving a probe or a wire is a good way to cause a momentary short that can vaporize the wire and give you a nasty plasma burn.

Close the lid before applying power: To protect you from vaporizing wires and exploding components, our Green Electronics lab kits have plexiglass lids. These lids let you see your circuit while protecting you from flying debris. However, they will only work if they are down when the explosion happens, so lower the lid before applying power.

Wear eye protection when power is applied: On the off chance that a particularly energetic explosion escapes the lid, wear eye protectin when circuits are powered on — either regular eyeglasses or safety glasses.

Check component temperatures before touching: Before touching a power component, use the infrared thermometer — not your finger — to check its temperature. Wait until all components have cooled to less than 40°C (104°F) before touching anything.

Check capacitor voltages before touching: Large capacitors can hold a significant charge for weeks. The Green Electronics modules have *bleeder* resistors on these capacitors to discharge them in a few minutes, but don't assume they are functioning. Either check a capacitor with a voltmeter, or short it to ground with a 25W 1K resistor (which will discharge a 1mF capacitor in a few seconds) before operating.

While less important from a safety perspective, the following guidelines will avoid uneccesary destruction of lab equipment:

Low-power first: Before hooking a lab to the 48V 100A (4.8kW) battery pack or to the 110V 15A (1.65kW) AC mains hook it up to a lab supply that can provide moderate voltage with a current limit. This will let you find a circuit problem without having the problem cause your circuit to destroy itself — leaving few clues of what went wrong.

Pulsed power second: Once your circuit is working on a lab power supply hook it to the high-power supply (battery or AC mains depending on the lab) and pulse the supply on and then quickly off. You should power the circuit just long enough to take a measurement with the oscilloscope. After powering down, check component temperatures. If everything is working as expected, power it on for a bit longer and check again. Repeat with longer time intervals until you are comfortable with steady-state operation.

Understand the startup transient: Many problems with power circuits happen during the transient when either the input power or the load is applied. An inrush of current when input power is applied can overload conductors or components and with parasitic inductance in the input path can lead to over-voltage conditions as well. Control systems are sometimes pegged in a *full-on* state when the input power is off leading to exessive current flow when power is first applied — particularly if the controller is slow to respond. Make sure you understand the transient behavior of your circuit during key transitions. It is a good idea to trigger a scope on each transient of interest to capture a "picture" of your circuit's behavior.

Make sure your controller is stable: Many other problems with power circuits stem from control problems. An unstable controller can easily drive a circuit to destruction. Make sure your controller is stable. Its also a good idea to put in one or more *fail-safe* checks that shut the circuit down if an out-of-limit condition is detected.

Understand a failure before trying again: Repeating an action and expecting a different response is the operational definition of insanity. If you power on your circuit and it explodes, don't just replace the burned-out parts and try again. Think. Understand what caused the failure and redesign (or reprogram) your circuit to fix the problem. If you don't understand the failure — and there's not enough of your circuit left to figure out what failed first — plan a low-power experiment to verify operation before attempting another high-power trial.

Appendix B

Processor Module

The Green Electronics Lab (GEL) processor module provides control and input/output functions for our Green Electronics labs. A photo of the module is shown in Figure B.1. The module is based on an Atmel ATMega1284 microcontroller.

B.1 Schematic

A schematic diagram of the GEL processor module is shown in Figure B.2. At the center of the diagram is the microcontroller, U1. The processor provides four 8-bit I/O ports A, B, C, and D some of the pins of which have special functions. 12V power is input through connector N3 and regulated to 5V by linear regulator U2. Capacitors C_1 through C_4 and inductor L_1 provide supply filtering. A crystal, X_1 provides a precision 12MHz reference that is needed when the USB port is used. For normal operation we operate the microcontroller using an internal 8MHz reference.

On-module display is provided by three seven-segment LEDs (LED1-LED3) and eight individual LEDs (D_1 through D_8). These four units (the three digits and the discrete LEDs) share a common 8-bit anode drive (via resistor arrays RA1 and RA2). Only one is turned on at a time via digit driver U3.

The module also provides 8-pin connector N4 to communicate with a standard two-line or four-line LCD module when more than three digits must be displayed.

Input is provided by four push-button switches (SW1-SW4) that are scanned by the digit drive, and a hexadecimal rotary switch SW5.

Header N1 plugs into a solderless breadboard and provides connection to

Figure B.1: The Green Electronics Lab Processor Module provides control and I/O functions.

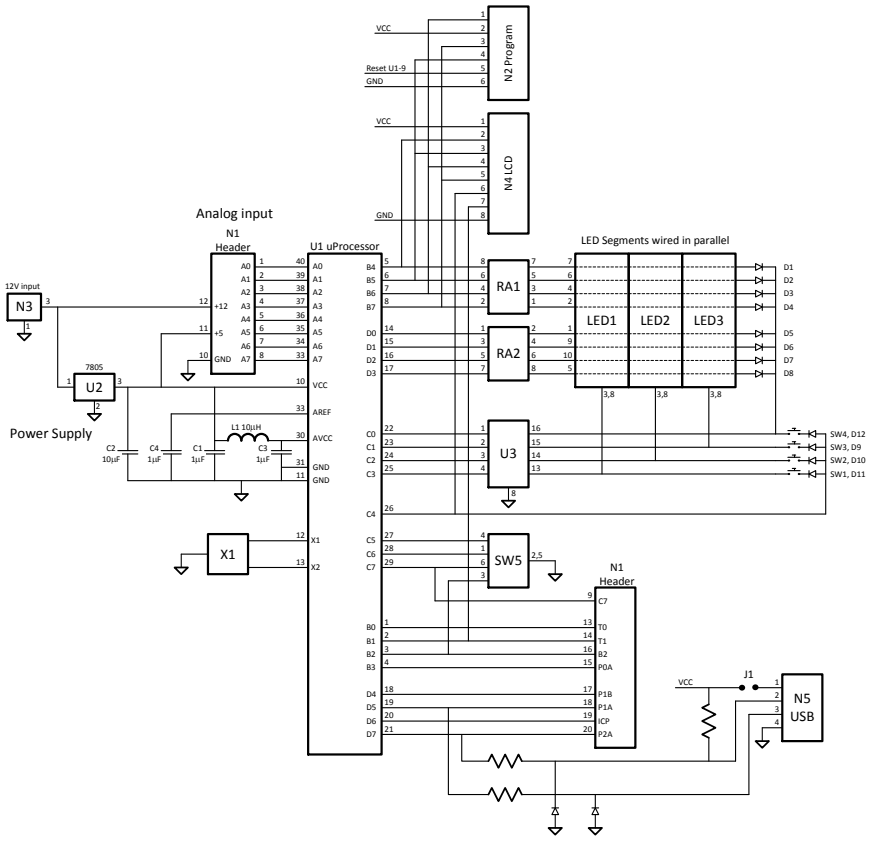


Figure B.2: Schematic of the GEL Processor Module.

selected port pins of the microprocessor. The first eight pins of the header connect to processor port A (A_0 to A_7). These pins can (among other functions) be configured as analog inputs to an A/D converter. Pins 10-12 of the header bring out the three power supplies on the module (GND, +5V, and +12V). The remaining pins bring out selected pins of other ports including the low Nybble of port B and the high Nybble of port D. Some of these pins are shared with other functions.

B.2 Software

A software library is provided to simplify access to the functions of the processor module. The following sections give an overview of the library. For details the reader should consult the source code.

Many of the library functions provide access to registers of the ATMega1284 processor. For details of these registers and the functions of the processor's A/D and timer/counters, refer to the processor manual available online. [1]

B.2.1 Display and Switch Input

To use the displays call `lcd_init()` and `led_init()` at the beginning of program execution. Then call `refresh_display()` once every 3ms. This will copy the values of arrays `led[4]` (segment data) and `lcd[80]` (ASCII characters) to the displays.

The preferred way to display data on the LED display is to use library routine `snprintf`. For example, the following program fragment displays three variables on the first line of the display and one variable on the second line.

```
snprintf(&lcd[0],SPN,"P%03d I%6d V%6d",pwm,current.value,voltage.value) ;
snprintf(&lcd[20],SPN,"T%6d",temperature.value) ;
lcd_fix_zeros() ;
```

`snprintf` does have the unfortunate side effect of writing a zero after the last character of the string being printed. To deal with this, the display should be written in order, so later writes overwrite any zeros from earlier writes that wrapped to the next line. After all writes are complete the routine `lcd_fix_zeros()` replaces any zeros remaining with a space (hex 20).

The preferred way to display data on the seven-segment LED display is to use library routines `set_hex_display(start,nr,value)` and `set_bcd_display(start,nr,value)`. These routines take a 16-bit unsigned `value` and display it as `nr` digits starting from digit `start`. Arguments `start` and `nr` are unsigned 8-bit values. For example, the following call displays `current.value` as a decimal number on the three LED digits:

```
set_bcd_display(0,3,current.value) ;
```

[1] At the time of writing, the processor manual was available at http://atmel.com/dyn/resources/prod_documents/doc8272.pdf.

If we would rather display the value in hexadecimal we use:

```
set_hex_display(0,3,current.value) ;
```

To update the discrete LEDs simply write the desired value to `led[3]`.

The state of the four pushbutton switches can be accessed by reading variable `pb`. The value of the hex switch is returned as an eight-bit unsigned value by routine `hex_in()`.

B.2.2 Real-Time Clock Interrupt

Most of our Green Electronics Labs are *interrupt driven*. The lab code is executed in response to processor interrupts. A *real-time clock* causes an interrupt to occur at regular intervals to allow certain functions to be performed on a periodic basis.

By convention we use timer/counter 2 of the ATMega1284 to generate real-time clock interrupts. To configure this counter, the user calls `rtc2_init` with three arguments to set up registers `TCCR2A, TCCR2B,` and `OCR2A`. The user then defines a routine `rtc()` that is called on each interrupt.

For example, to set up an interrupt once every 1ms, the counter is configured with the call

```
rtc2_init(TCCRA_CTC, TCCR2B_D32, OCR2A_1MS_D32) ;
```

The first argument configures the counter in CTC (clear timer on compare match) mode. The second sets the counter source to divide the 8MHz processor clock by 32 - so the processor counts at 250kHz. The final argument sets the counter to compare and cause an interrupt when it reaches 250 - which gives a interrupt every 1ms.

To get an interrupt every 3ms the call is:

```
rtc2_init(TCCRA_CTC, TCCR2B_D1K, OCR2A_3MS_D1K) ;
```

Routine `rtc()` then defines what happens on each interrupt. This routine might request A/D conversions, refresh the display, and count down to perform other functions at longer periods.

For example, consider the following routine.

```
void rtc() {
  adreq |= AD_0 ; // start conversion on channel 0
  if(adidle) start_ad() ;
  sei() ;         // enable interrupts
  refresh_display() ; // refresh LED and update LCD
  if(++timer_50 > TIME_50) { // count down for 50ms routine
    t50=0 ;
    rtc50ms() ;  // do this every 50ms
  }
}
```

This routine requests an A/D conversion on channel 0 by ORing `AD_0` into `adreq` and then starting the converter if it is idle. Interrupts are then renabled to allow higher priority interrupts access to the processor while the remaining functions are performed. Next the displays are refreshed with a call to `refresh_display()`. The final loop counts down to perform other real-time functions every 50ms in routine `rtc50ms()`.

B.2.3 Analog Input

The ATMega1284 processor includes a 10-bit analog-to-digital (A/D) converter. The converter runs on a 125kHz clock (the 8MHz processor clock divided by 64) and takes 13 cycles to perform a conversion, so just over 9 conversions can be performed each 1ms.

To use the A/D converter, the programmer must first make a call to `adc_init()` to configure the converter. Conversions are requested by ORing requests into bit-vector `adreq`, and then calling `start_ad()` if `adidle` is true — as in the code fragment in Section B.2.2 above.

When a conversion is complete, the A/D interrupt handler calls a user-defined routing corresponding to the request to handle the converted value which is available in variable `adcv`. For the request `AD_0` a routine `ad_0()` is called. The following code fragment shows an example `ad_0()` routine.

```
void ad_0() {
  current.raw = adcv ;
  calibrate(&current) ;
}
```

This routine copies the result of the conversion into the `raw` field of structure `current` and then calls a calibration routine to compute a calibrated current value.

B.2.4 PWM

For some of our labs we need to produce pulse-width modulated waveforms — a digital signal with a fixed frequency and a pulse-width that is varied as required. For our labs we will configure one or both of timer/counter 0 and timer/counter 1 to generate PWM waveforms.

To use a timer/counter as a PWM generator, it is first configured with a call to an initialization routine. Then the pulse width is set by writing a control value to the appropriate output compare register.

For example, to use timer/counter 0 to generate a 16kHz PWM signal on output A we first configure the counter with the following call:

```
pwm0_init(TCCRA_PWM, TCCRB_D1) ;
```

The first argument sets control register TCCR0A to select PWM mode with output A enabled. The second argument sets the counter to operate directly

off of the 8MHz processor clock. Because the 8-bit counter in PWM mode goes through 510 cycles for each pulse, this gives a pulse frequency of 15.69kHz.

To set the pulse width, we set register `OCR0A`. A value of zero produces a minimum-width pulse - 125ns wide. A value of 255 (hex FF) produces an output that is always high.

B.2.5 Calibration

The library provides a calibration function to facilitate calibrating inputs from the A/D converter. The function works with the following structure.

```
struct cal { // value to be calibrated
  uint16_t raw ;  // raw sample
  uint16_t zero ; // raw value of zero
  int32_t scale ; // multiplied by 2^16
  int16_t value ; // calibrated value
} ;
```

Calibration is performed by measuring the raw A/D values at two points, zero and a *standard* value. For example, suppose we have measured the raw A/D readings for voltage at zero (`voltage_zero`) and 10V (`voltage_10v`) and that we want our calibrated voltage to be in units of 100mV. We initialize the structure with the following code.

```
  voltage.zero = voltage_zero ;
  // 10V std, 0.1V resolution
  compute_scale(&voltage,voltage_10v, 100UL) ;
```

We set the zero point by just assigning to `voltage.zero`. The value of `voltage.scale` is then computed by calling `compute_scale` with the raw value of our calibration measurement `voltage_10v` and the value we would like this to correspond to `100` (since 10V is 100 100mV increments). The routine computes the appropriate scale factor and stores it in the structure.

Once the structure is initialized each time we wish to convert a raw measurement to a calibrated value we just assign the raw value and call `calibrate`. The calibrated value is stored in the `value` field of the calibration structure. For example, when we read from the A/D we can execute the following code:

```
  voltage.raw = adcv ; // raw value read from A/D
  calibrate(voltage) ; // computes calibrated value in voltage.value
```

B.2.6 Filtering

To reduce measurement noise on slowly changing signals the library provides a filter structure and routine that implements an *autoregressive moving average* (ARMA) filter. The filter structure contains three fields:

```
struct filter { // variables to implement ARMA filter
  int32_t running_avg ;// running average
  int16_t filtered ;   // last filtered value
  uint8_t shift ;      // amount to shift by
}
```

A 32-bit running average `running_avg` holds the state of the filter. This value is shifted left by `shift` bits. The `filtered` field holds the last output of the filter. The `shift` field sets the time constant of the impulse response of the filter.

The filter function:

```
void filter(int16_t value, filterP f) {
  f->running_avg = f->running_avg - f->filtered + value ;
  f->filtered = (f->running_avg) >> f->shift ;
}
```

performs the following calculation

$$y_i = \frac{2^s - 1}{2^s} y_{i-1} + \frac{1}{2^s} x_i \tag{B.1}$$

where y_i is the filter output at timestep i, x_i is the filter input at timestep i, and s is the shift value.

Appendix C

Component Module

Figure C.1 shows a photograph of the Green Electronics component module. A schematic diagram of the module is shown in Figure C.2. The module contains an AC input circuit with fuse, rectifier, and filter capacitor, a second capacitor, an inductor, a current sense resistor, and a current transformer.

Power connections to the module are made via 0.25-inch quick-connect tabs (sometimes called *faston* or *blade* connectors). Signal connections are made via a 6-pin female header that accepts 22AWG solid hookup wire for connections to a solderless breadboard.

Each of the sub-modules are described in more detail in the following sections — in increasing order of complexity. For the individual components the reader is encouraged to consult the manufacturers data sheets for more details.

C.1 Current Sense Resistor

The current sense resistor, shown at the bottom left of Figure C.2 is simply a 10mΩ 5W resistor (a TT Electronics open-air resistor, part number OAR5R010FLF). Power connections are made to the resistor via two quick-connect terminals labeled *Ra* and *Rb*. Signal connections are available via two female header receptacles with the same labels. This four-point connection to a current-sense resistor, called a *Kelvin* connection, allows a differential voltage to be measured just across the resistor — avoiding any parasitic voltage drops on the power conductors leading to and from the resistor.

Figure C.1: Photograph of the Green Electronics Component Module.

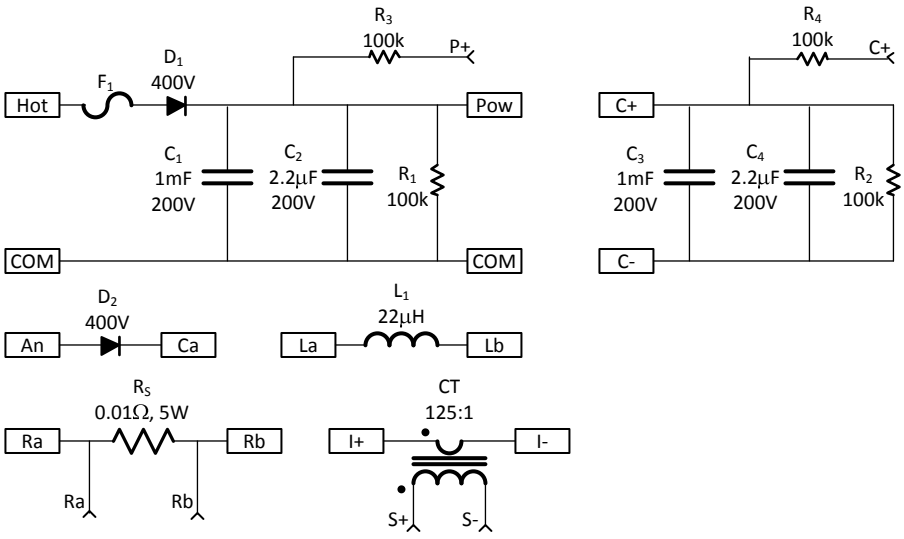


Figure C.2: Schematic diagram of the Green Electronics Component Module.

C.2 Inductor

A surface-mount power inductor is provided on the module, shown in the center of Figure C.2. This is a 22μH inductor with a current rating of 11.7A (a Vishay IHLP-6767GZER220M51). This inductor has a series resistance of $R_s = 22\text{m}\Omega$ and a saturation current of 11A[1].

In high-current operation, the inductor will dissipate power due to the series resistance $P_s = IR_s$ and due to core losses. A heat sink is provided on the inductor to help remove this power via natural or forced convection. However, the temperature of the inductor should be monitored and current reduced, frequency reduced, or additional airflow provided if the case temperature exceeds 100°C.

DO NOT TOUCH THE INDUCTOR to determine if it is getting hot. Use an infrared temperature meter.

At maximum current $I = 10A$, the inductor stores energy $E_L = 0.5I^2L = 1.1\text{mJ}$.

[1]This is the current at which the inductance L drops to 80% of its nominal value.

C.3 Diode

A 400V 20A diode is provided on the module, shown at the center left of Figure C.2 and is accessed via quick-connect tabs labeled *Anode* and *Cath*. The current batch of component boards uses an NXP BYT79X-600,127 for this component, which is actually a 600V 40A diode with a 55ns reverse recovery time, a maximum forward voltage of 1.9V, and a maximum reverse leakage current of 200μA.

C.4 Capacitor

The capacitor unit, at the upper right of Figure C.2 includes a 1mF 200V aluminum electrolytic capacitor C_3, a 2.2μF 200V film capacitor C_4, a 100kΩdrain resistor R_2, and a 100kΩsense resistor R_4. The two capacitors and the drain resistor are connected in parallel and to two quick-connect tabs, labeled $C+$ and $C-$. The sense resistor connects the positive terminal of the capacitors to one of the female header sockets, labeled $C+$.

Capacitor C_3 is a 1mF 200V aluminum electrolytic capacitor (Panasonic EET-ED2D102CA). This part has an effective series resistance of $R_s = 0.149\Omega$a maximum ripple current of $I_r = 2.84A$ and a lifetime of 3,000 hours at 105°C. To first approximation, the lifetime doubles for each 10°C reduction in temperature below this point.

When selecting a capacitor for energy storage in a green electronics application the series resistance and ripple current limits are often as important, if not more important, than the actual capacitance. The lifetime rating is also critical. Electrolytic capacitors often limit the reliability of green electronic systems. To design a system to operate 10 years (about 80,000 hours) for example, with a 3,000 hour at 105°C capacitor, one must ensure the temperature of the capacitor remains below 57°C.

The series resistance of C_3 gives it a *pole* at $f = \frac{1}{2\pi RC}$ 1kHz. The overall impedance of the capacitor is the sum of its resistive and capacitive impedance $Z_C(s) = R_s + \frac{1}{Cs}$. Below 1kHz, this impedance is dominated by the capacitance. Above 1kHz the resistive term dominates. Above a few 100kHz parasitic inductance dominates.

To provide capacitance above 1kHz, we place film capacitor C_4 in parallel with C_3. Capacitor C_4 is a 2.2μF (Panasonic ECQ-E2225KF). This capacitor has negligible series resistance, but its leads have a series inductance of about 10nH. This gives it a resonant frequency of $f = \frac{1}{2\pi\sqrt{LC}}$ 1MHz. Above this frequency the impedance of C_4 is dominated by its inductance.

Drain resistor R_2 is provided to drain the charge off of C_3 (and C_4) when not in use. When fully charged to $V = 200$V, C_3 contains $E_C = 0.5CV^2$ =20J of energy. This could deliver a nasty shock even hours after the circuit is disconnected. To prevent this, R_2 drains the energy from C_3 with a time constant of $\tau = RC = 100$s. Because of this long time constant one should WAIT AT LEAST 5 min (3τ) before touching the capacitor terminals after powering the

circuit down.

Sense resistor R_4 is intended to form the upper branch of a voltage divider to sense the voltage on the capacitor. For example, to scale a 200V voltage to a the 2.5V range of the A/D requires a divider with an 80:1 ratio. Using a pull-down resistor with the standard value of 1.2kΩ gives a ratio of 84.3:1 which will suffice. The 100kΩresistor limits the current that flows if the sense lead is disconnected from the bottom branch of the voltage divder or shorted to ground to about 2mA.

C.5 Current Transformer

The current transformer is a small transformer with a 1-turn primary and a 125-turn secondary that provides moderate frequency isolated current measurement. When a current is passed through the primary of the transformer — accessed via quick-connect tabs $I+$ and $I-$, a current 125-times smaller appears between header terminals $S+$ and $S-$ which can be sensed via a small-valued resistor. To avoid negative voltages a diode is often used in series with the sense resistor.

For example, a 20A current in the primary will cause a $\frac{10}{125} = 160$mA current to flow in the secondary. Using a 6.25Ω sense resistor will generate an 1V sense voltage.

This particular transformer is an ICE CT04-125. It is rated for up to a 20A primary current and has a secondary inductance of 2.2mH. The primary inductance is given by $L_P = \frac{L_s}{N_r^2} = 140$nH. This primary inducance affects the circuit being measured.

C.6 AC Input Circuit

The AC input circuit provides protection, rectification, and filtering for a 110V AC input. Capacitors C_1 and C_2 and resistors R_1 and R_3 are a copy of the capacitor submodule. Refer to Section C.4 for a discussion of their function.

Fuse F_1 (nominally 10A) provides circuit protection. Should their be a downstream short causing more than this amount of current to flow, this fuse will *blow* opening the input circuit and shutting off the current flow. However, even a *fast-acting* fuse may take a large fraction of a second to blow — more than enough time for considerable damage to occur to the downstream circuitry.

Diode D_1 is the same diode as used in the diode submodule (Section C.3). In this case, it is wired as a half-wave rectifier. When the *hot* AC input (the black wire) is in the positive half of its cycle and at a voltage above *Pow*, D_1 will conduct causing *Pow* to follow *hot* up to the top of its cycle (at about 170V). At that point the diode becomes reverse biased and shuts off until the next cycle. At the point where *hot* is at its most negative voltage (-170V), there can be as much as 340V across the reverse-biased diode — hence the need for a minimum voltage of 400V for this part.

The capacitors are used to filter the half-wave AC input. During the small part of the AC cycle when D_1 conducts, C_1 and C_2 are charged to 170V. During the remainder of the cycle, these capacitors supply current to the downstream circuit, partly discharging. To avoid overstressing the capacitor the downstream circuit should not draw more than $I_r = 2.84$A on average — higher peaks can be provided by C_2. To first approximation, C_1 filters the AC input, storing energy during the portion of the AC cycle when no current is being provided and C_2 filters the higher frequency (often 200kHz) pulsed output current — allowing C_1 to see the average of this current.

Appendix D

Half-Bridge Module

The half-bridge module is a push-pull driver stage that drives an output signal to either a power supply or ground. The driver uses power MOSFETs that are capable of driving very high currents. For our motor lab, we will use this module configured with FETs capable of driving up to 195A with a series resistance of $R_{ON} = 2\text{m}\Omega$. A MOSFET driver allows these MOSFETs to be controlled with ordinary logic signals.

A photograph of the module is shown in Figure D.1 and a schematic is shown in Figure D.2. The module drives the voltage from supply V_D or common *COM* (another name for ground) onto the output *Out* via MOSFETs M1 and M2. Two quick-connect tabs are provided for each of these high-power signals.

A four-pin header plugs into a solderless breadboard to provide a +12V supply *V12* signal ground *GND* high drive *Hin* and low-true low drive *Lin'*. An IRS21834 half-bridge driver converts these logic signals to higher-voltage, higher-current gate drive signals for the MOSFETs. Circuitry is included to provide a floating *bootstrap* supply for the high side driver (the V_B supply) and to filter the drain voltage to the half-bridge (the V_D filter).

The remainder of this appendix describes each section in more detail.

D.1 Power MOSFETs

MOSFETs M1 and M2 perform the main function of the half-bridge module switching the output between V_D and *COM*. The major properties of the half-bridge module are inherited from the properties of these MOSFETs. For our motor controller and photovoltaic controller labs we use a version of the half-bridge module populated with IRLB3036. As described in Table M.1 this is a 60V MOSFET with a typical on resistance R_{ON} of 2m$\Omega$and a maximum current

Figure D.1: Photograph of the Green Electronics Half-Bridge Module.

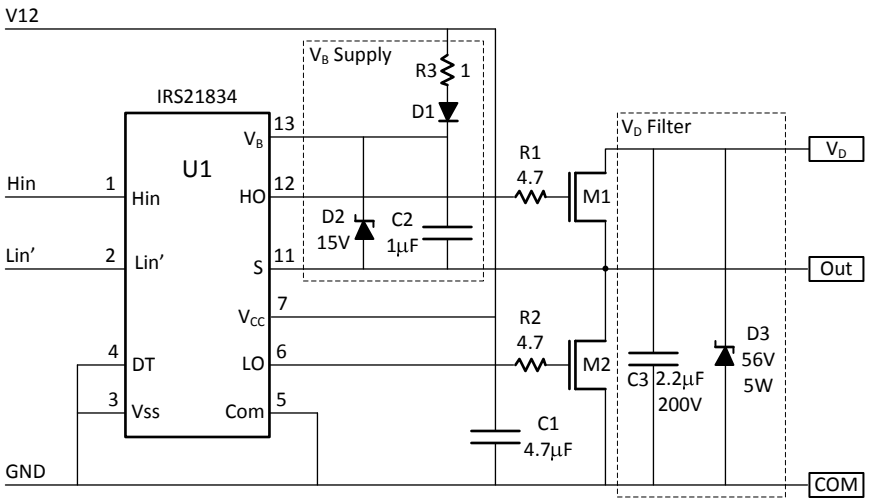


Figure D.2: Schematic diagram of the Green Electronics Half-Bridge Module.

of 195A.

We choose this transistor for our motor and PV labs to get a low on resistance with a maximum voltage that is comfortably above the 48V operating voltage of these labs.

For a different application we could populate the half-bridge module with a different MOSFET. For example if we chose the IRFB4332 FET we would have a 250V half-bridge with $R_{ON} = 29\text{m}\Omega$and 60A maximum current. Such a module would be suitable for switching drain voltages V_D up to about 200V.

While the IRFB3036 is rated for 195A maximum current[1], in practice the maximum amount of current is limited by power dissipation and cooling. At 195A and 2mΩthe conduction losses in the FET are $P_c = I^2R = 76W$ additional power is dissipated due to switching losses. The heat sinks on the Green Electronics lab kits have a thermal resistance of about[2] 3°C/W. If we want to limit the temperature rise of these FETs to no more than 60°C we are limited to a power dissipation of no more than 20W. Ignoring switching losses, this limits the practial maximum current to $I = \sqrt{\frac{P}{R}} = 100\text{A}$, more than enough for our purposes.a

[1]The 195A is actually a package limit. the transistor itself is rated at 270A.
[2]This number is approximate and may require forced-air cooling.

D.2 MOSFET Drivers

Power MOSFETs are happy (dissipate minimal power) in two states — on and off. When a MOSFET is on, it has little voltage across it. When a MOSFET is off, it has no current through it. In both cases there is minimal power dissipation.

When a MOSFET switches, however, it simultaneously has both high current and high voltage - leading to high power. For our IRFB3036, for example, during switching this device may for an instant have a current of $i = 100$A and a voltage of $v = 48$V giving it an instantaneous power dissipation of $P = iv = 4.8kW$. Fortunately it is energy that kills, not power, so as long as we don't allow the FET to remain in this high-power state for more than a few nanoseconds it won't burn itself up.

The role of the MOSFET driver is to drive the gates of the MOSFETs to rapidly turn them on and off — to minimize switching losses.

ground isolation

Appendix E

Buck Module

E.1 Circuit Measurements and Subtleties

E.1.1 Measurements

[Replot this figure with white background]

An oscilloscope screen shot of four key waveforms of our battery charger is shown in Figure E.2. We introduce the waveforms here and then describe each in detail below. The top waveform is the output of our switch (point V_x in Figure 4.3). This is a square wave that swings through up to 170V. The next waveform is the output of our current transformer (the voltage across R_{CT} in Figure 4.9). The third waveform is the voltage at the drain of FET M1 V_D. Finally, the bottom waveform is the output voltage V_O.

The waveform on V_x is shown in more detail in Figure E.3 and closeups of the overshoots of the rising and falling edges are shown Figures ?? and ??. The rise time is ?? 20ns. It is important to keep the rise time short to keep the switching losses in M1 small. To first approximation, the energy lost on the rising edge is

$$E_r = t_r I_L V_i. \tag{E.1}$$

With $V_i = 170$V and I_L =10A, this is $E_r = (2 \times 10^{-8})(170)(10) = 34\mu$J. With a frequency of $f = 200$kHz we have $P_r = fE_r = 7$W.

The down side of a fast rise time is that it couples energy into various parts of the circuit through parasitic capacitance. This in turn causes ringing as evidence by the ringing on multiple waveforms of Figure E.2.

It is very important to manage the current in inductors on transients. The current transformer has an inductance of 500nH.

with 125:1 xformer - 10A

Current Sense Switch Output Battery Voltage Drain Voltage

Also

3843 oscillator 3843 output

Repeat top for discontinous mode at low current

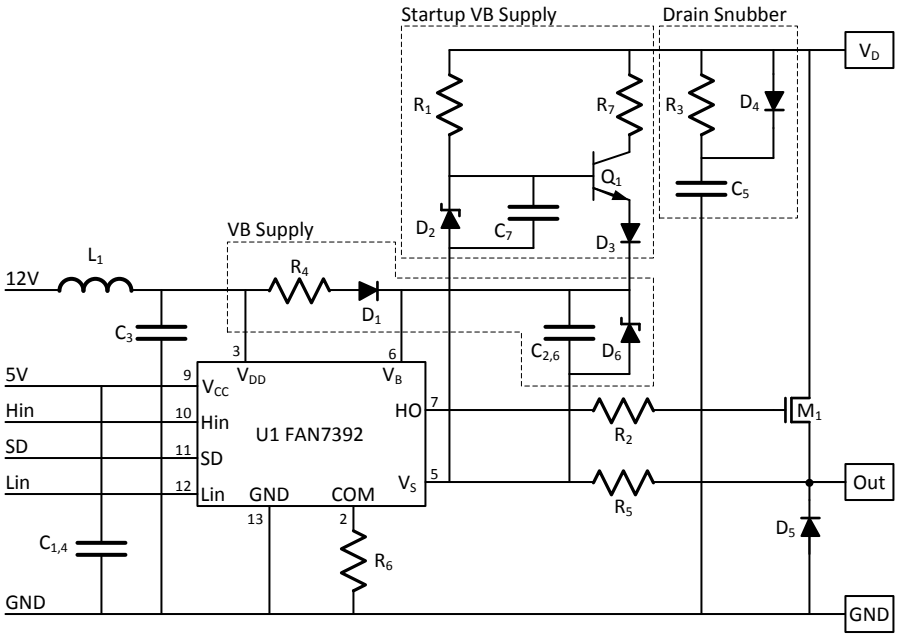


Figure E.1: Schematic of the buck module.

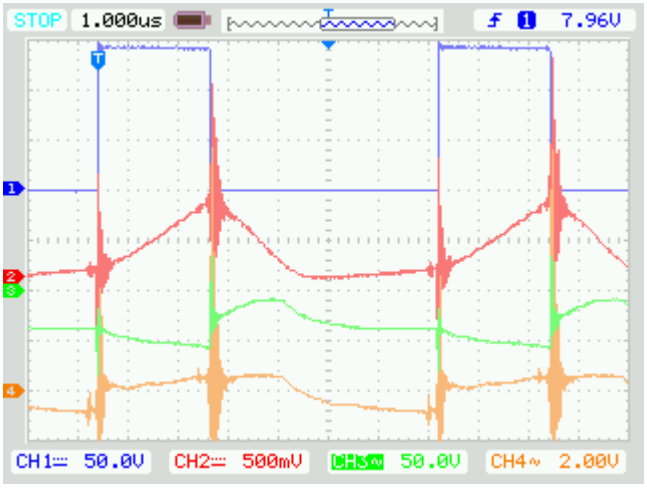


Figure E.2: Waveforms from battery charger. From top to bottom: V_x, I_{CT},V_D ,V_O.

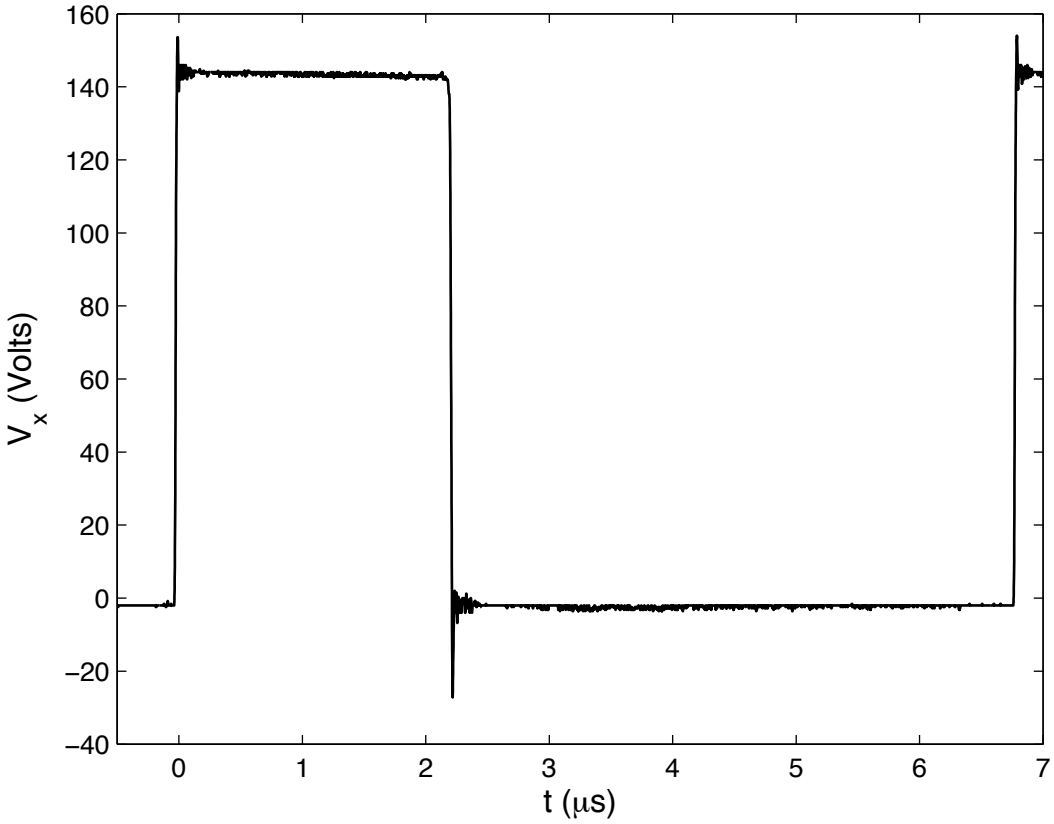


Figure E.3: Oscilloscope trace of V_x waveform.

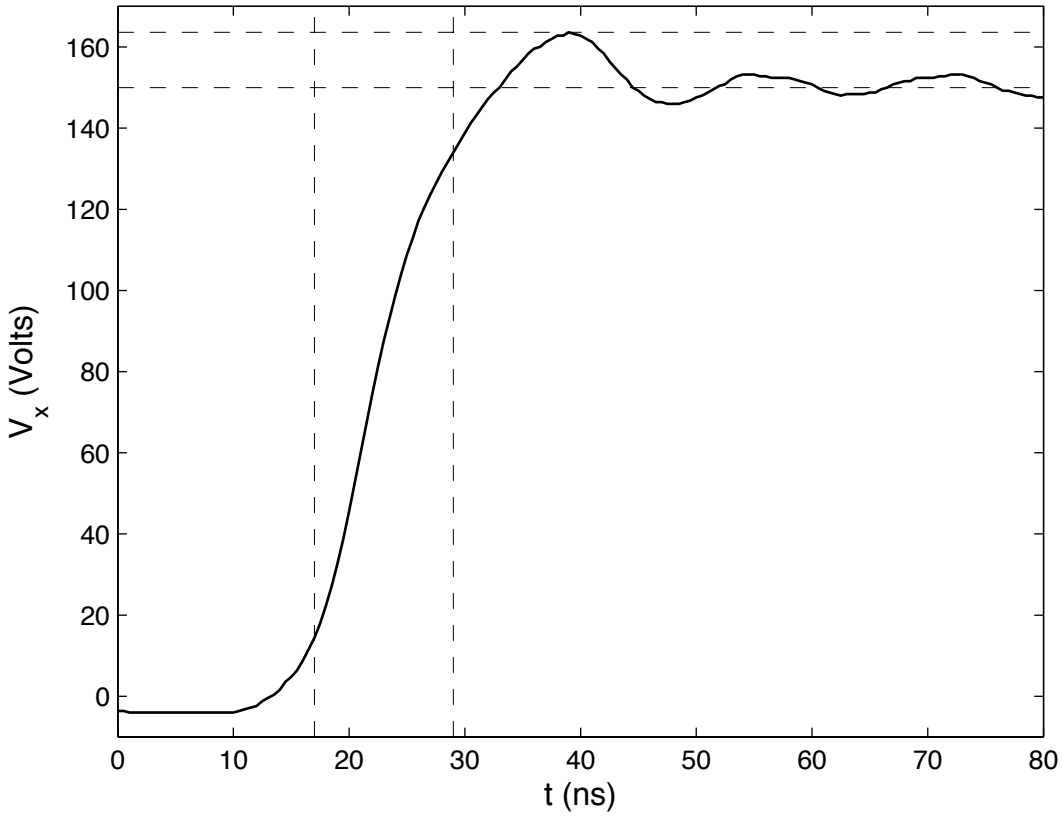


Figure E.4: Close-up of rising edge of V_x. Rise time is 15ns and overshoot is 14V.

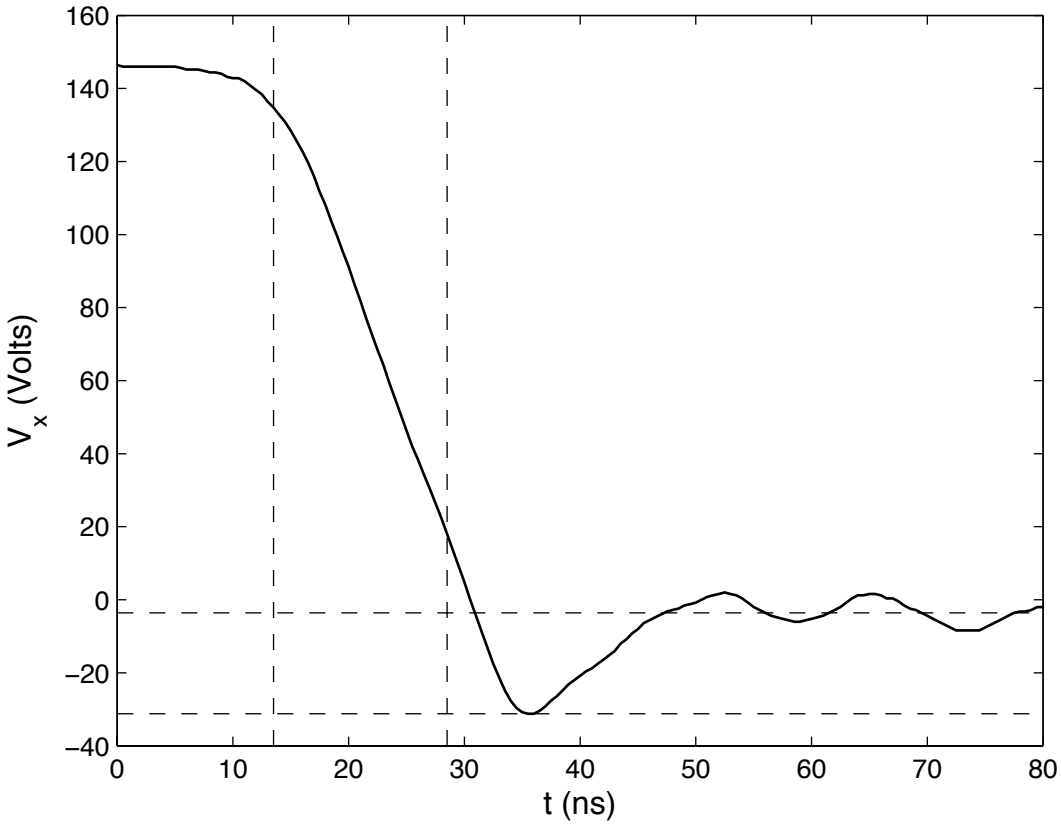


Figure E.5: Close-up of falling edge of V_x. Fall time is 19ns and undershoot is 28V.

E.1.2 Input Snubber

E.1.3 Overlap Current

Appendix F

Motor and Battery Cart

contains motors with sensors
load
48V of batteries with switch and fuse

Appendix G

Project Ideas

Power-Factor Correcting Input Circuit:

Inverter:

Battery Management System:

Battery State Indicator:

High-Efficiency Power Supply:

Intelligent Lighting:

Intelligent Appliance Control:

Appendix H

Circuit Theory

This appendix gives a quick overview of the basics of linear DC circuit theory. That is, circuits composed of independent current and voltage sources and resistors.

H.1 Voltage, Current, and Power

Electric circuits involve the movement of charged particles (electrons and holes) through conductors and components (such as resistors, capacitors, and inductors). We refer to the movement of charged particles as *current* which is measured in *Amperes* or *Amps* which we abbreviate as A. One Ampere is a flow of 1 Coulomb of charge every second.

If we move a charge carrier across an electric field, that carrier aquires potential energy - just as if we lift a mass through a gravitational field. We refer to the potential of a particle as its *voltage* which is measured in *Volts*. A conductive node of a circuit has carriers that are all of the same potential which we refer to as the voltage of that node. We pick one node in a circuit arbitrarily to be a reference of zero Volts — this is often called *ground* or *GND*.

Many people find an analogy to hydraulics helpful to understand current and voltage. An electrical circuit is analgous to a hydraulic circuit — where water molecules replaces the charge carriers. The voltage of a node corresponds to the pressure in a water pipe. The current in a wire corresponds to the flow of water through a pipe — liters per second.

The *power* produced by a source is the product of the voltage across the source times the current out of the source:

$$P = VI. \tag{H.1}$$

If the net current flows into the positive terminal of a device, we refer to it as a *load* rather than a source. We tend to refer to power out of a source or into a load.

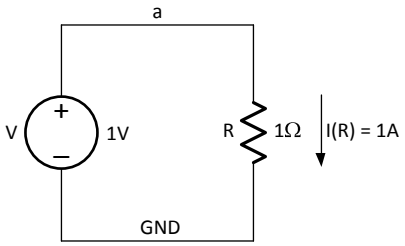


Figure H.1: A simple electrical circuit. A voltage source V applies 1V across a 1Ω resistor R. A current of 1A flows through the component.

The *energy* produced by a source or consumed by a load over a period of time is just the integral of the power over that period.

$$E = \int_{t=t_1}^{t_2} P dt. \tag{H.2}$$

If power is constant over the period, energy is just the product of power and time

$$E = P(t_2 - t_1). \tag{H.3}$$

When analyzing circuits its is important to keep in mind that *voltage* always refers to a relative measurement between two nodes. There is no such thing as the voltage of a node, a, $V(a)$. There is only the difference in voltage between two nodes a and b $V(a) - V(b)$. Sometimes we will relax this convention and use $V(a)$ to refer to $V(a) - V(GND)$. However, even when we use this shorthand, it is important to remember that voltage is always a relative difference between two nodes.

To bring these concepts together, consider the simple circuit of Figure H.1 that contains two nodes *GND* and a and two components, V and R. The circular symbol represents a *voltage source* V with a value of 1V. The positive terminal, node a is always 1V higher in potential than its negative terminal *GND*. A voltage source is analgous to a water pump that causes the pressure at its output to be a fixed amount higher than the pressure at its input.

The other component in our circuit is a resistor R with a value of one *Ohm* or Ω. As we shall see shortly applying 1V across the terminals of a 1Ω resistor causes a current to flow in the circuit with a value of $I = V/R = 1$A.

The instantaneous *power* dissipated by the resistor and produced by the

source is given by:

$$P = VI. \tag{H.4}$$

This is analagous to water flowing down a hill returning its potential energy as kinetic — and ultimately thermal — energy. A component *dissipates* power if current flows into its more positive terminal — like the resistor in Figure H.1 and it *generates* power if current flows out of its more positive terminal — like the voltage source in Figure H.1.

H.2 Resistors and Ohm's Law

A resistor, like the one in our simple circuit, is a passive element that, as its name implies, resists current flow. Giving a current that is proportional to the voltage applied across its terminals. The flow of current through the resistor is governed by *Ohms Law*:

$$V = IR, \tag{H.5}$$

or equivalently:

$$I = V/R. \tag{H.6}$$

The constant of proportionality that relates voltage and current here is called *resistance* and has units of *Ohms*, abbreviated Ω.

Figure H.2 shows photographs of a few example resistors. Most resistors today are *surface-mount* (SMT) resistors like those shown in Figure H.2(a). These small rectalinear solids are soldered directly to the surface of a printed circuit board. Each resistor is labeled with its resistance in scientific notation with the "E" omitted. For example, the resistors labeled 222 have a resistance of 22E2 or 2,200Ωor 2.2kΩ.

SMT resistors are described by their resistance and their size, which is specified by their length and width in hundredths of an inch. The resistors in Figure H.2(a) are 0805 resistors which are 0.08 × 0.05 inch (2.0 × 1.3 mm) and have a power rating of 0.1W. Resistors as small as 01005 (0.01 × 0.005 inch) with a power rating of 1/32 W are widely used.

Older systems use *through-hole* resistors like the 1/4 Watt resistor shown in Figure H.2(b). These resistors are inserted through holes in a printed-circuit board and soldered in place. Through hole components are often used for small-quantity prototypes because they are easier to assemble by hand and rework than SMT components. SMT components, however, are preferred for high-volume assemblies because they are less expensive to assemble with automated machinery, require less board area, and don't require holes to be drilled and plated.

Small through-hole resistors are labeled with their resistance value using the color code shown in Table H.1. For example, the resistor in Figure H.2(b) has

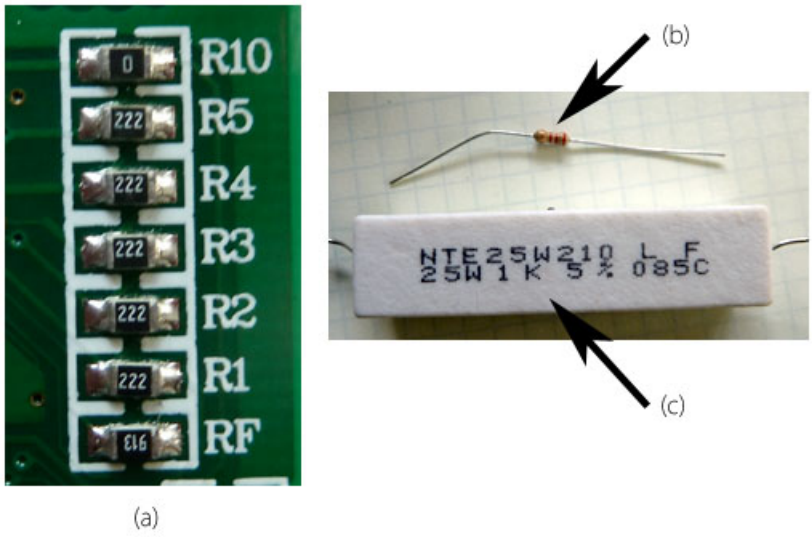


Figure H.2: Some example resistors: (a) surface mount resistors, (b) 1/4 W through-hole resistors, (c) 25W through hole resistor.

three red lines coresponding to 222 or 2.2kΩ. The fourth stripe on the resistor of Figure H.2(b) denotes the tolerance of the component with gold corresponding to 5%.

In addition to its resistance, a resistor is characterized by its power rating, its tolerance, and its temperature coefficient. For high-power applications, the resistor must be physically large to dissipate the power without excessive temperature rise like the 25W resistor of Figure H.2(c).

H.3 Kirchoff's Laws

Two conservation laws, *Kirchoff's Current Law* (KCL) and *Kirchoff's Voltage Law* (KVL) govern the analysis of circuits.

Kirchoff's Current Law (KCL) states that the sum of currents into a node is zero. That is that current is conserved at each node of a circuit — it is neither created nor destroyed.

$$\sum_{b \text{into} N} i_b = 0; \tag{H.7}$$

Figure H.3 illustrates KCL. The figure shows a node N with three adjacent branches b_1, b_2, and b_3. The three branch currents into N must sum to zero

Color	**Value**
Black	0
Brown	1
Red	2
Orange	3
Yellow	4
Green	5
Blue	6
Violet	7
Gray	8
White	9
Gold	-1
Silver	-2

Table H.1: Numeric color code used for labeling electronic components.

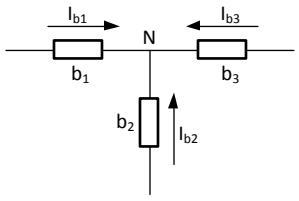


Figure H.3: Kirchoff's current law (KCL). The three currents into node N must sum to zero, $I_{b1} + I_{b2} + I_{b3} = 0$.

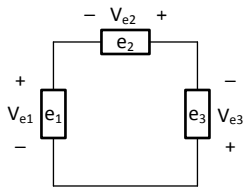


Figure H.4: Kirchoff's voltage law (KVL). The voltages across the three elements of this loop — oriented in a clockwise direction — must sum to zero: $V_{e1} + V_{e2} + V_{e3} = 0$.

because current cannot be created or destroyed at node N. This means that if we know any two of the currents we can calculate the third. For example, the current flowing into branches b_1 and b_2 must equal the current flowing out of b_3, $-I_{b3} = I_{b1} + I_{b2}$.

Kirchoff's Voltage Law states that the sum of voltages around a loop is zero. That is, if you sum up the voltages of the elements around any loop in the circuit you will wind up back where you started.

$$\sum_{e \text{around} L} V_e = 0; \tag{H.8}$$

Figure H.4 illustrates KVL. The figure shows a loop with three elements e_1, e_2, and e_3. If we walk around this loop in a clockwise direction and consider the first terminal of each element we encounter the negative terminal, as labeled in the figure, the sum of the voltages across the three elements must sum to zero: $V_{e1} + V_{e2} + V_{e3} = 0$. If this were not the case, the node we started at would have two different voltages with respect to the other two nodes.

H.4 Examples

The use of KCL and KVL are best illustrated by a series of examples that themselves show some useful circuit identities.

H.4.1 Series Resistors and Voltage Dividers

Consider the series connection of two resistors shown in Fig H.5. Applying KCL (H.3) at node b we know that $I_{R1} = I_{R2} = I$. Applying KVL (H.3) around the loop and Ohm's law (H.2) to calculate the voltage across each resistor, we know

$$V = IR_1 + IR_2 = I(R_1 + R_2). \tag{H.9}$$

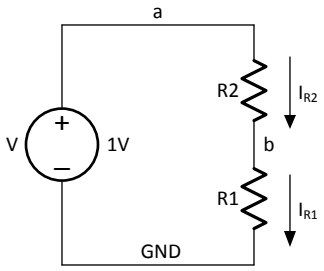


Figure H.5: Series connection of two resistors. Applying KCL and KVL we see $R_s = R_1 + R_2$ and $V(b) = V(R_1) = \frac{VR_1}{R_1+R_2}$.

From this expression we see that the resistance of two resistors (or any number of resistors) in series is just the sum of their resistances.

$$R_s = \sum_i R_i \tag{H.10}$$

From (H.4.1) we calculate the current to be $I = \frac{V}{R_1+R_2}$. And thus the voltage at node b is:

$$V(b) = V(R1) = V\left(\frac{R1}{R1+R2}\right). \tag{H.11}$$

Because the voltage of source V is divided across the series resistors in proportion to their resistance, a series connection of resistors is sometimes called a *voltage divider* and the voltage at each point in the divider can be found by applying (H.4.1).

H.4.2 Parallel Resistors and Current Dividers

Figure H.6 shows a 1A current source driving a parallel connectin of two resistors. From Ohm's law (H.2) we have $I_{R1} = V(a)/R1$ and $I_{R2} = V(a)/R2$. Applying KCL at node a gives $I = I_{R1} + I_{R2} = V(a)\,(1/R1 + 1/R2)$. Solving for $V(a)$ gives $V(a) = I\frac{1}{1/R1+1/R2}$. Thus, the equivalent resistance of two (or more) resistors in parallel is:

$$R_P = \frac{V(a)}{I} = \frac{1}{\sum_i 1/R_i}. \tag{H.12}$$

which for the special case of two resistors we can rewrite as

$$R_P == \frac{R_1 R_2}{R_1 + R_2}. \tag{H.13}$$

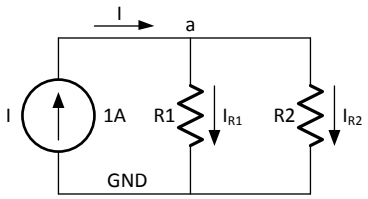


Figure H.6: Parallel connection of two resistors. Applying KVL and KVL we see $R_P = \frac{1}{1/R1+1/R2} = \frac{R1R2}{R1+R2}$ and $I(R1) = \frac{I/R_1}{1/R1+1/R2} = \frac{IR_2}{R_1+R_2}$.

The resistors in parallel act as a *current divider* with the amount of current I flowing in resistor R_i being proportional to $1/R_i$.

$$I_{R1} = I\left(\frac{1/R_1}{1/R_1 + 1/R_2}\right) = I\left(\frac{R_2}{R_1 + R_2}\right). \tag{H.14}$$

The reciprocal of resistance is called *conductance*, $G_i = 1/R_i$. Rewriting (H.4.2) in terms of conductance, we have

$$I_{R1} = I\left(\frac{G_1}{G_1 + G_2}\right). \tag{H.15}$$

H.4.3 Combined Series Parallel Circuit

Consider the series-parallel circuit of Figure H.7(a). Suppose we need to solve for the voltage at node b. We start by replacing the parallel combination of R_2 and R_3 with an equivalent resistor R_P using (H.4.2) as shown in Figure H.7(b). Then applying (H.4.1) we have

$$\begin{aligned} V(b) &= V\left(\frac{R_S}{R_1 + R_S}\right), && \text{(H.16)} \\ &= V\left(\frac{\frac{R_2R_3}{R_2+R_3}}{R_1 + \frac{R_2R_3}{R_2+R_3}}\right), && \text{(H.17)} \\ &= V\left(\frac{R_2R_3}{R_1R_2 + R_1R_3 + R_2R_3}\right). && \text{(H.18)} \end{aligned}$$

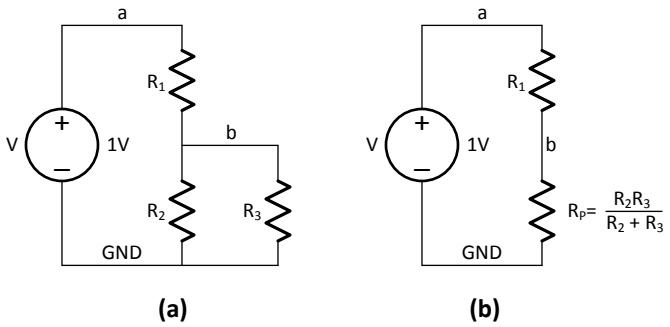


Figure H.7: A series-parallel resistor circuit.

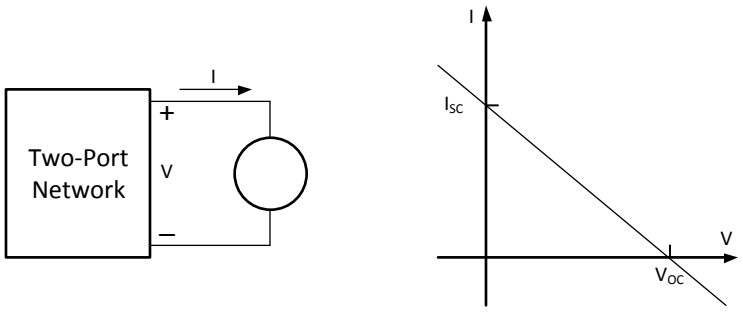


Figure H.8: V-I characteristics. A two-port network can be characterized by its V-I characteristic. If the network is *linear* this characteristic is a straight line with intercepts at the network's open-circuit voltage V_{OC} and its short-circuit current I_{SC}.

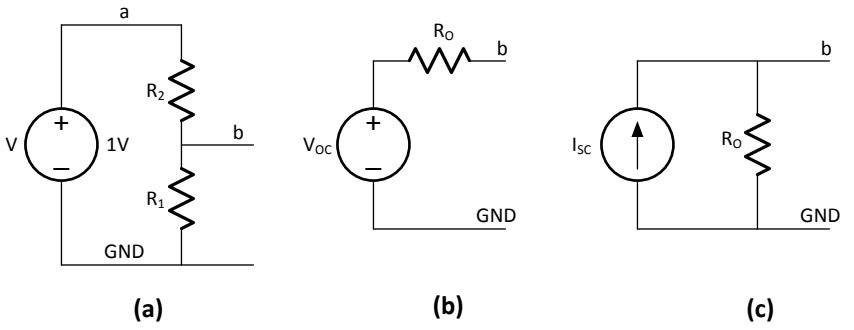


Figure H.9: Thevenin and Norton Equivalents. (a) A voltage divider. (b) Thevenin-equivalent circuit. (c) Norton-equivalent circuit

H.4.4 Thevenin and Norton Equivalents

As shown in Figure H.8(a), a circuit with two ports (a two-port network) can be characterized by applying a voltage source with voltage V between the two terminals and measuring the current, I. The function I(V) is called the V-I characterisitic of the circuit.

Because resistors are *linear* elements, any circuit composed entirely of independent current and voltage sources and resistors has a linear V-I characteristic as shown in Figure H.8(b). We call the intercepts of this curve the *open-circuit voltage* V_{OC} and short-circuit current I_{SC} of the network. V_{OC} is the voltage on the network terminals when the current is zero and I_{SC} is the current when the voltage is zero — i.e., when there is a *short circuit* across the network terminals. The ratio of these two quantities is the output resistance of the circuit $R = V_{OC}/I_{SC}$.

From the network terminals, two networks with the same V_{OC} and I_{SC} are indistinguishable. It is sometimes useful to substitute a canonical network with the same V-I characteristic. Consider the voltage divider of Figure H.9(a). From (H.4.1) we know that $V_{OC} = \frac{R_1}{R_1+R_2}$. When the output is shorted (i.e., node b connected to GND), the current is due to the 1V source across R_2 so $I_{SC} = \frac{1}{R_2}$. The output resistance is $R_O = \frac{V_{OC}}{I_{SC}} = \frac{R_1 R_2}{R_1+R_2}$ — the parallel combination of the two resistors.

To simplify analysis, we can replace the voltage divider with a *Thevenin-equivalent* circuit with the same V_{OC} and R_O as shown in Figure H.9(b). A Thevenin-equivalent circuit is just a voltage-source with value V_{OC} and a series resistor vith value R_O. When the output is shorted, the current flowing through the resistor is $I_{SC} = \frac{V_{OC}}{R_O}$.

Alternatively we can replace the voltage divider with a *Norton-equivalent*

circuit as shown in Figure H.9(c). Here a current source with value I_{SC} is connected in parallel with a resistor with value R_O. When no output current flows, all of the current from the source flows through the resistor generating $V_{OC} = I_{SC}R_O$.

H.5 Linearity and Superposition

Appendix I

Capacitors and RC Circuits

Green Electronic systems rely heavily on *capacitors* and *inductors*, circuit elements that store energy so that we can convert it from one form to another. In this chapter we will explore the properties of capacitors and circuits composed of capacitors and resistors. In Appendix J we will visit inductors.

I.1 Capacitors

A capacitor is a two-terminal device that stores energy in an electric field. A schematic symbol for a capacitor is shown in ??. As current flows into a capacitor, the capacitor *charges up*, building up voltage across its two terminals. The capacitor can later be *discharged* releasing the stored energy.

At any point in time, the *charge* Q in the capacitor is proportional to the voltage across the capacitor's terminals.

$$q = Cv \tag{I.1}$$

The constant here C is the *capacitance*. Differentiating Erefcap1 with respect to time gives us

$$\frac{dq}{dt} = C\frac{dv}{dt} \tag{I.2}$$

$$i = C\frac{dv}{dt} \tag{I.3}$$

$$\tag{I.4}$$

Thus, we see that the current through a capacitor is proportional to the derivative of the voltage across the capacitor.

Returning to our analogy to hydraulics, a capacitor is like a tank of water. As water flows into the tank the level in the tank, and hence the pressure at the bottom of the tank, increases. Similarly, as current flows into a capacitor the voltage in the capacitor increases.

Figure I.1: RC Circuits: (a) a low-pass filter, (b) a high-pass filter

If we start with a discharged capacitor and charge it to V volts with a steady 1A current, the capacitor voltage will increase linearly over the CV seconds required to complete the charge, so $v(t) = t/C$. At any point in time, the charging power is $P = iv = v = t/C$. Thus, integrating over the CV seconds the total charging energy, which is stored in the capacitor at the end of charging, is:

$$E_C = \int_0^{CV} \frac{t}{C} dt = \frac{1}{2C} t^2 |_0^{CV} = \frac{CV^2}{2}. \tag{I.5}$$

Capacitor circuits are particularly easy to analyze using Laplace transforms. Taking the transform of (I.1) gives:

$$i(s) = Csv(s). \tag{I.6}$$

Thus, we can consider a capacitor as an impedance with value:

$$Z_C = \frac{V(s)}{I(s)} = \frac{1}{Cs}. \tag{I.7}$$

I.2 RC Circuits

We use KCL and KVL to analyze RC circuits in the same manner that we analyzed purely resistive circuits. Consider the high-pass filter circuit of Figure **??**. This is just a voltage divider with the bottom resistor replaced by a capacitor. From KCL we know that the current in the resistor is the same as the current in the capacitor, thus we have

$$\frac{v_i - v_o}{R} = C\frac{dv_o}{dt}. \tag{I.8}$$

We can solve (I.2) for specific waveforms $v_i(t)$. For example, for a unit step, $v_i(t) = u(t)$, we have

$$v_o(t) = 1 - exp\left(\frac{t}{RC}\right). \tag{I.9}$$

This is a *low-pass* response with a time constant of $\tau = RC$.

As a second example, consider the

I.3 Impulse Response and Step Response

Appendix J

Inductors, LR, and LC Circuits

Appendix K

Operational Amplifiers

An *operational amplifier* or *opamp* is a three-terminal device that provides gain. That is, the power out of the amplifier is greater than the power into the amplifier. We use opamps to construct a variety of signal conditioning circuits to amplify, level shift, and filter circuits. This appendix introduces the opamp, describes how to analyze opamp circuits, gives a number of example circuits, and discusses the limitations of real opamps.

K.1 Model

As shown in Figure K.1, an opamp can be modeled as a dependent voltage source where the output voltage is proportional to the difference between the positive and negative input voltages.

$$V_O = A\,(V_P - V_N) \tag{K.1}$$

The gain A is assumed to be very large.

An opamp is most often used with *negative feedback* from the output to the negative input. Because the gain A is very large, as long as the output of the opamp remains in its operating region (i.e., it doesn't peg at a power supply), the negative feedback will drive the negative input to the same voltage as the positive input. Thus, when we analyze opamp circuits with negative feeback we can, to first approximation, assume $V_N = V_P$.

With this assumption, analyzing an opamp circuit is a simple process. We first compute the voltage on the positive input, V_P. Then, we calculate what output voltage V_O is required to make $V_N = V_P$. We illustrate this analysis method in the next section on several common opamp circuits.

K.2 OpAmp Circuits

Figure K.2(a) shows an opamp connected as a *voltage follower*. As the name implies, the output voltage follows in the input voltage $V_O = V_I$. The opamp

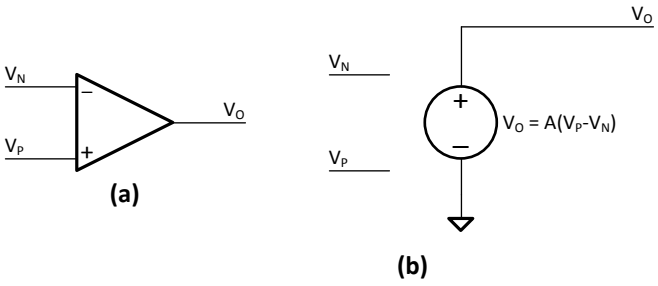


Figure K.1: An opamp can be modeled as a *dependent* voltage source. The output voltage is equal to a gain A times the difference in input voltages, $V_O = A\,(V_P - V_N)$: (a) schematic symbol, (b) model.

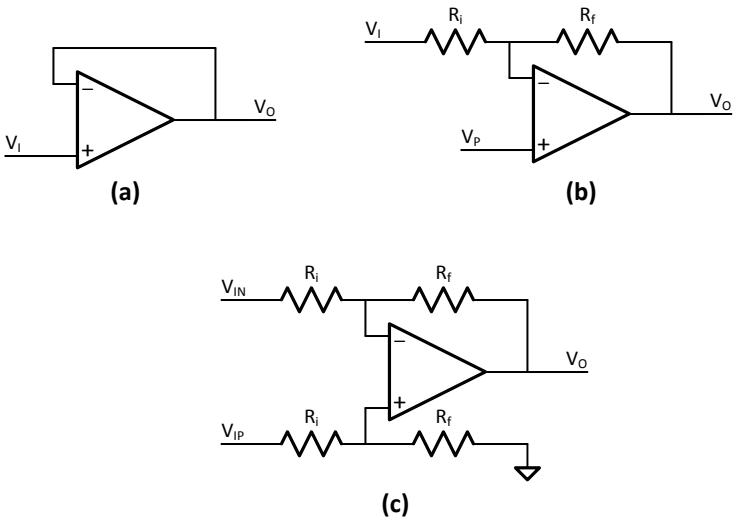


Figure K.2: Common opamp circuits: (a) voltage follower, (b) inverting and non-inverting amplifier, (c) differential amplifer.

provides isolation and gives a low impedance drive. It is used for example to provide a low-impedance copy of a voltage generated by a high-resistance voltage divider (as in Figure 2.2).

The analysis of this circuit using our two-step procedure is trivial. The voltage on the positive input is $V_P = V_I$. Assuming $V_N = V_P$, the output voltage required is $V_O = V_P = V_I$.

Figure K.2(b) shows an opamp connected as an *inverting* or *non-inverting* amplifier. From input V_I this is an inverting amplifier while from input V_P it is a non-inverting amplifer.

To analyze this circuit lets first consider the case where $V_P = 0$, i.e., the positive input is connected to ground. In this case our assumption that $V_N = V_P$ makes V_N a *virtual ground*. Thus, the current in R_i flowing toward the input is $I(R_i) = -\frac{V_I}{R_i}$. Because no current flows into the opamp input, by KCL, all of this current continues on into R_f giving $V(R_f) = I(R_i)R_f = -V_I\frac{R_f}{R_i}$. With $V_N = 0$ this is also the output voltage:

$$V_O = -V_I\frac{R_f}{R_i}. \tag{K.2}$$

This circuit gives amplifies the input with a gain of $-\frac{R_f}{R_i}$.

Now consider the case where V_P is non-zero and V_I is zero. Assumign $V_N = V_P$, we have $I(R_i) = \frac{V_P}{R_i}$ giving a voltage across R_f of $V(R_f) = V_P\frac{R_f}{R_i}$. This is summed with the voltage $V_N = V_P$ to give an output voltage of

$$V_O = V_N + V(R_f) = V_P + V_P\frac{R_f}{R_i} = V_P\left(1 + \frac{R_f}{R_i}\right). \tag{K.3}$$

By superposition we can write the general expression for V_O as:

$$V_O = V_P\left(1 + \frac{R_f}{R_i}\right) - V_I\frac{R_f}{R_i}. \tag{K.4}$$

Here the gain from the positive input is one more than the negation of the gain from the negative input.

In cases where we want to amplify a differential voltage, we would like the gains from the two inputs to be equal and opposite. The circuit shown in Figure K.2(c) accomplishes this by dividing V_{IP} to give $V_P = V_{IP}\frac{R_f}{R_i+R_f}$. Substituting this into (K.2) gives:

$$V_O = (V_{PI} - V_{NI})\left(\frac{R_f}{R_i}\right). \tag{K.5}$$

Note that the *common-mode rejection* of this circuit depends on the matching between the two pairs of resistors.

Figure K.3: Opamp circuits using capacitors: (a) integrator, (b) differentiator, (c) low-pass filter, (d) two-pole low-pass filter.

K.3 OpAmp and Capacitor Circuits

We can analyze opamp circuits using capacitors by substituting the complex impedance of the input and feedback networks into (K.2). Consider the *integrator* of Figure K.3(b). We have $Z_i = R_i$ and $Z_f = \frac{1}{Cs}$ which gives

$$V_O(s) = -V_I(s)\frac{1}{RCs}. \tag{K.6}$$

so

$$V_O = -\frac{1}{RC}\int V_I dt. \tag{K.7}$$

The same result can be arrived at by applying our two-step analysis method using the constituent equation for the capacitor.

Putting the capacitor into the input circuit as in Figure K.3(b) gives a differentiator.

$$V_O(s) = -V_I(s)RCs. \tag{K.8}$$

$$V_O = -RC\frac{dV_I}{dt}. \tag{K.9}$$

If we combine resistors and capacitors in each leg we get a more complex functionality. For example, Figure K.3(c) shows a low-pass filter. To qualitatively analyze this circuit, recall that at low frequencies the capacitor is an open and at high frequencies the capacitor is a short. Thus, at DC this is just an inverting amplifier with gain $-\frac{R_f}{R_i}$, and at high frequencies the output is zero — hence the low-pass functionality. We leave detailed analysis of both this circuit the two-pole version in Figure K.3(d), and corresponding high-pass and band-pass circuits as exercises.

K.4 Real OpAmps

Up to this point we have been assuming an ideal opamp. While actual opamps are quite good, they are not perfect and some are better than others. Real opamps have limitations of operating range, finite gain, finite bandwidth, limited common mode rejection, offset voltage, finite input impedance, and finite output current.

In his section we will look at the real opamp we will use in our labs, an MCP602 5V dual opamp to illustrate some of these limitations. The MCP602

Parameter	Value	Units
Input Range	-0.3 to 3.8	Volts
Output Range	0.1 to 4.9	Volts
Gain	$> 10^5$	V_O/V_I
Gain-Bandwidth Product	2.8	MHz
Common-Mode Rejection Ratio (CMRR)	$> 5,000$	
Input Offset Voltage	< 2	mV
Input Current	< 60	pA
Output Current	< 1	mA

Table K.1: Parameters of an MCP602 opamp

is an integrated circuit that contains two opamps that operate from a single 5V supply[1]. The key parameters of each amplifier in an MCP602 are shown in Table K.1. These parameters assume a power supply of V_{DD} of exactly 5V and have been simplified a bit from the manufacturers data sheet. You should consult the data sheet for details.a

Perhaps the most important parameters are the input and output range. For the amplifiers in the MCP602 to function as opamps, their input and output voltages must remain within these ranges. For example, if a differential amplifier Figure K.2(c) is used to sense the voltage across a current-sense resistor and the terminals of the resistor drop well below ground due to impulse noise, the output of the op-amp will be in error — and damage may even be done to the op amp.

Similarly, if keeping $V_N = V_P$ would require V_O to go above the power supply or below ground, it won't. The output voltage will peg at the supply and the amplifer will not be able to force the two inputs together.

The gain A and gain-bandwidth product specify the frequency response of the part. The MCP602 has a DC gain of 10^5 with a single pole at 28Hz The amplifier acts like an ideal amplifer in series with an RC low-pass filter with $f_c = \frac{1}{2\pi RC} = 28$. This causes the gain to roll-off to a value of 1 at 2.8MHz. Because feedback requires reasonably high gain, at least 100, the MCP602 is really only suitable for applications needing a bandwidth of no more than 28kHz.

The common-mode rejection ratio specifies the gain from the common-mode or average of the two inputs ($V_{CM} = \frac{V_P+V_N}{2}$) to the output. The common-mode gain is the reciprocal of the rejection ratio. That is the gain from the common-mode to the output is $A_{CM} = \frac{1}{5000}$.

The input offset voltage V_{OS} is a DC noise voltage in series with one of the two inputs. Negative feedback drives V_N to $V_P \pm V_{OC}$. In a circuit with gain, the error seen at the output will be the input offset voltage times the gain. For example, in an inverting amplifier (Figure K.2(b)) with a gain of -10, a 2mV input offset voltage gives an error of up to ±20mV at V_O.

While the input current of the opamp isn't zero, 60pA is pretty close. Only the highest-impedance sensors need be concerned with this finite input current.

[1]It can operate from 2.7 to 6.0V, but we will use the part with a 5V supply.

On the other hand, the output drive is quite modest. If more than 1mA of output current is needed, additional buffering is required.

K.5 Exercises

common mode rejection
- low pass analysis
- two-pole low-pass analysis
- high-pass
- band-pass

Appendix L

Real Circuit Elements

L.1 Resistors

resistor - pretty close to ideal, power limit, tolerance

L.2 Inductors

inductor - series resistance, interwinding capacitance

L.3 Capacitors

capacitor - series inductance, esr types for different frequency bands and voltages
electrolytic polymer film (polycarbonate) ceramic
    tolerances and standard values

L.4 Transformers

magnetizing and leakage inductance
    core saturation
    series resistance

L.5 Standard Values

L.6 Cost Model

a simple cost model for components

Appendix M

Switches

M.1 Metal-Oxide-Semiconductor Field Effect Transistor (MOSFET)

MOSFETs

zero-th order model - on or off - when on has resistance Ron

1st order model - turned on which charge - amount of charge determines conductance Qth needed to start turning on Then conductance linearly goes from 0 to 1/Ron with next Qon Beyond that point

Gate is non-linear capacitor

Key specifications are Vds max, Ids max, Ron, and tsw

Figure of merit is V^2/R which has dimensions of power.

V/R per dollar is about 9E5 for 100V to 400V FETs and about 4E5 outside this range.

Table showing some typical power mosfets

M.2 Insulated-Gate Bipolar Transistors

IGBTs

Figure of merit is P=VI All drop about 2V, so conduction power is 2I

Part No.	$V_{DS\max}$ (V)	$I_{DS\max}$ (A)	R_{ON} (mΩ)	Q_g (nC)
IRLB3036	60	195	2	91
IRFB4332	250	60	29	99

Table M.1: Characteristics of selected power MOSFETs

M.3 Diodes

Switches to allow current to flow in only one direction. Called rectification. Simple control (none), but costs us a diode drop (from 0.5 to 2V) Function of diode often done today by MOSFET (synchronous rectification).

M.4 Silicon Controller Rectifiers (SCRs)

SCRs

M.5 Driver circuits

M.6 Isolation

Appendix N

Feedback Control

Many *green electronic* systems involve controlling a device to drive a key parameter to a set point or to optimize some metric. For example, in our photovoltaic controller (Chapter **??**) we control pulse width t_a to drive the operating voltage of our solar cell array V_i to a set point V_t. This set point, in turn, is controlled to optimize power. As a second example, in our motor controller (Chapter ??) we control pulse width to drive motor torque to a set point.

In this chapter we show how the principles of feedback control can be applied to build controllers of this type.

This chapter assumes a knowledge of Laplace transforms because their use greatly simplifies the mathematics needed to describe these systems. If you are not familiar with Laplace transforms, you can use the results derived in a *cookbook* manner without the need to understand their derivation.

N.1 Basic Principles

The principles of feedback control can be applied any time we need to control a device via an input to drive an output to a desired setpoint. A basic feedback

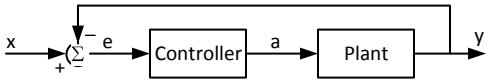


Figure N.1: A basic feedback control system. The The current output y is subtracted from the input set point x to produce an error signal e. A controller uses the error signal to compute an input w to drive the *plant* to the desired output.

control system is shown in Figure N.1. The device we are trying to control, called the plant, takes an input signal w and produces an output signal y. We use a controller to generate the input signal based on an error signal, the difference between the desired set point and the current output, $e = x - y$.

There are many examples of feedback control in everyday life. The process of steering a car, for example, involves feedback control. In this case, the car is the plant, the position (or angle) of the steering wheel is the input, w, and the position of the car on the road is the output, y. The driver, as the controller is mentally computing an error signal, e, by observing the difference between the car's position, y, and its desired position x, usually the center of the lane. Using this error signal, the driver determines a steering wheel angle w that will return the car to the desired position, x.

The car, as a plant, integrates the steering wheel position w to give the agngle, or direction of the car, and then integrates the angle of the car to give its position on the road. Hence we have $y(t) = \int \int aw(t)dt^2$, where a is the gain of the steering system (in units of $\frac{\mathrm{m}}{\mathrm{s}^2\mathrm{rad}}$)

To generate an input w that drives the plant to the desired state, we will use controllers that combine the error, its integral, and its derivative: integral

$$w(t) = pe(t) + q \int e(t)dt + r\frac{de(t)}{dt}. \tag{N.1}$$

We refer to this as a proportional-integral-differential or PID controller. The constant's p, q, and r are the proportional, integral, and differential gain respectively. The response of our overall system is determined by these three constants along with the characteristics of the plant. By setting p, q, and r appropriately we can get the plant output $y(t)$ to track the input $x(t)$, rapidly responding to transients in x but with minimum overshoot and no oscillation.

It is particularly important that we choose gain constants that give a *stable* system response, one where a bounded input gives a bounded output. Choosing the constants incorrectly can give a system where a simple transient on the input results in unbounded output oscillations, or where an input of a particular frequency excites a resonance — giving an output of increasing magnitude.

Returning to our car example, if we use just a proportional controller, setting the steering wheel position w equal to our distance from the center of the lane e, our car will oscillate back and forth across the centerline of the lane — and we are likely to get stopped and asked to take a sobriety test. To control the car in a stable manner, we need to include a derivative term in our control law, reducing our steering input w proportion to the rate at which we are approaching the centerline. In the sections below, we explain how to choose the proper values for p, q, and r to give the desired response.

We often use subsets of the full PID controller. If all we need is proportional control, we use a P controller where $q = r = 0$. If $r = 0$ we have a PI controller, and if $q = 0$ we have a PD controller.

The rest of this section formulates a feedback control system using Laplace transforms. If you are not familiar with Laplace transforms, you can skip to the beginning of Section N.2.

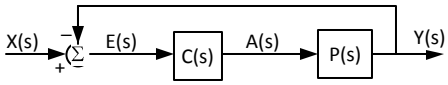


Figure N.2: A feedback control system expressed using Laplace transforms.

The feedback control system is easy to analyze using Laplace transforms. Consider the system as redrawn in Figure N.2 with signals x, y, e, and w represented by their Laplace transforms, $X(s)$, $Y(s)$, $E(s)$, and $W(s)$ respectively and the controller and plant represented by their transfer functions $C(s)$ and $P(s)$. From this block diagram we can write

$$Y(s) = E(s)C(s)P(s) = (X(s) - Y(s))C(s)P(s). \tag{N.2}$$

Which reduces to

$$Y(s) = X(s)\frac{C(s)P(s)}{1 + C(s)P(s)} \tag{N.3}$$

Expressing this as a transfer function

$$G(s) = \frac{Y(s)}{X(s)} = \frac{C(s)P(s)}{1 + C(s)P(s)}. \tag{N.4}$$

We often refer to the combined forward transfer function as $H(s) = C(s)P(s)$ in which case these equations become

$$Y(s) = X(s)\frac{H(s)}{1 + H(s)} \tag{N.5}$$

$$G(s) = \frac{Y(s)}{X(s)} = \frac{H(s)}{1 + H(s)}. \tag{N.6}$$

Returning to our car example, the transfer function of the car is $P(s) = \frac{a}{s^2}$. When we apply a PD controller with gains p and r, we get a combined forward response of

$$H(s) = C(s)P(s) = (rs + p)\left(\frac{a}{s^2}\right) = \frac{ars + ap}{s^2}. \tag{N.7}$$

Closing the feedback loop around this system gives

$$G(s) = \frac{H(s)}{1 + H(s)} = \frac{ap + ars}{s^2 + ars + ap}. \tag{N.8}$$

From Equation (N.8) we can compute the natural frequency and damping factor of the entire system. Or, conversely, we can compute the values of p and r that give us the desired natural frequency and damping factors.

N.2 First Order System

A first-order system has a response that is determined by a first-order differential equation. A typical first-order system is described by a single parameter, its time constant, τ, and it responds to a unit step on its input by following the input with an decaying exponental curve

$$y(t) = 1 - \exp\left(\frac{-t}{\tau}\right). \tag{N.9}$$

As we shall see in the derivation below, we can improve the time constant of a first-order system by using a feedback control system with a PI controller. The resulting system is a second-order system. The gains of this controller p and q along with the original time constant τ determine the frequency ω and the damping factor ζ of the resulting system as follows:

$$\omega = \sqrt{aq} \tag{N.10}$$

where $a = \frac{1}{\tau}$ and

$$\zeta = \frac{p+1}{2\tau\omega} = \left(\frac{p+1}{2}\right)\sqrt{\frac{a}{q}}. \tag{N.11}$$

Of course in designing controllers with these equations, it is important to keep in mind that our controller has limited output range. For our photovoltaic controller of Chapter **??**, for example, the pulse width t_a is limited to a minimum of 0 (we can't have a negative pulse width) and a maximum of t_c (at most our pulse can be high all the time). If we increase our gains in an attempt to make our response time arbitrarily fast, we are likely to exceed the operating range of our controller. The controller will *clip* the input w to its range resulting in non-linear behavior. Clipping the input is not necessarily a bad thing, but it causes the response to deviate from what is predicted by linear system analysis.

If you aren't familiar with Laplace transforms, you can stop reading this section here. Equations (N.10) and (N.11) are all you need to design PI controllers for first-order systems using a cookbook method. If you are curious how this is derived, read on.

Consider, for example, our motor control system. In this case, the motor is the *plant*, the input to the motor a is the pulse width (which determines the effective voltage seen by the motor), and the output y we are concerned with is the motor's torque, which is proportional to the current through the motor's winding. For the special case where the motor is at rest, it is basically an inductor and if we ignore its series resistance we have

$$y = k\int a\mathrm{d}t, \tag{N.12}$$

where k is a constant that is proportional to $\frac{V}{L}$. Taking the transform gives

$$P(s) = \frac{k}{s} \tag{N.13}$$

and

$$Y(s) = \frac{kA(s)}{s} \tag{N.14}$$

If we use a proportional control law, where the input is proportional to the error $a(t) = pe(t)$, substituting into (N.6) gives

$$G(s) = \frac{kp}{s + kp} = \frac{a}{s + a}, \tag{N.15}$$

where $a = kp$ is the combined gain.

If we apply a unit step, $X(s) = \frac{1}{s}$ to this system, the response is

$$Y(s) = \frac{a}{s^2 + as}, \tag{N.16}$$

which in the time domain gives

$$y(t) = 1 - \exp(-at). \tag{N.17}$$

In this case, the feedback has converted the integrator into a first-order system where the output follows the input with a time constant of $\tau = \frac{1}{a}$. The response of this system with $a = 10$ to a unit step function is shown in Figure N.3.

For a general first-order system where

$$P(s) = \frac{a}{s + a}. \tag{N.18}$$

Applying a proportional controller with gain k gives

$$G(s) = \frac{ka}{s + a(k + 1)}. \tag{N.19}$$

The response to a unit step is

$$Y(s) = \frac{ka}{s^2 + a(k + 1)s}. \tag{N.20}$$

Which in the time domain is

$$G(s) = \frac{k}{k + 1} \left(1 - \exp(-a(k + 1)t)\right). \tag{N.21}$$

Here the feedback control has reduce the time constant by a factor of $k + 1$ from $\tau = a^{-1}$ to $\tau = (a(k+1))^{-1}$. However, at the same time, the feedback has reduced the overall gain from input to output from 1 to $\frac{k}{k+1}$.

If we want the output to exactly follow the input, a simple proportional controller won't do. One commoon approach is to add an integral term to our control equation. In this case

$$C(s) = p + \frac{q}{s}. \tag{N.22}$$

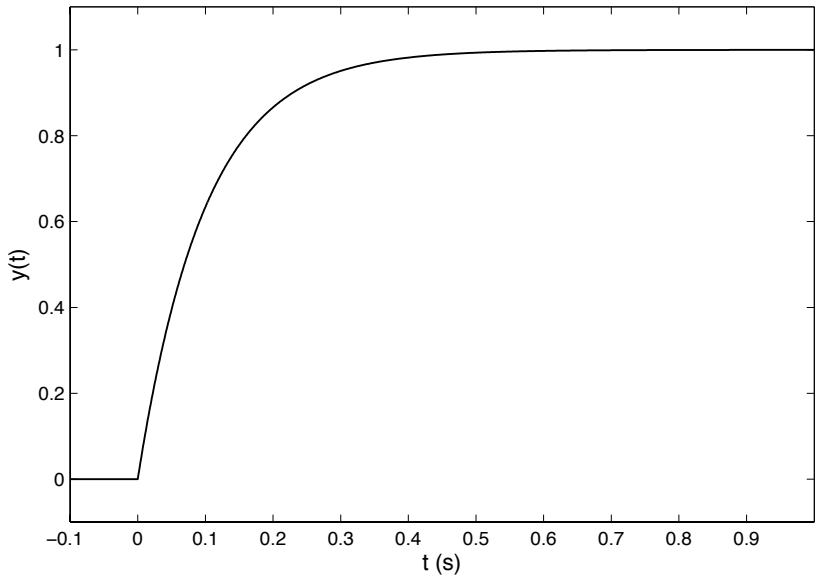


Figure N.3: Response of the system of Equation (N.15) with $k = 10$ to a unit step.

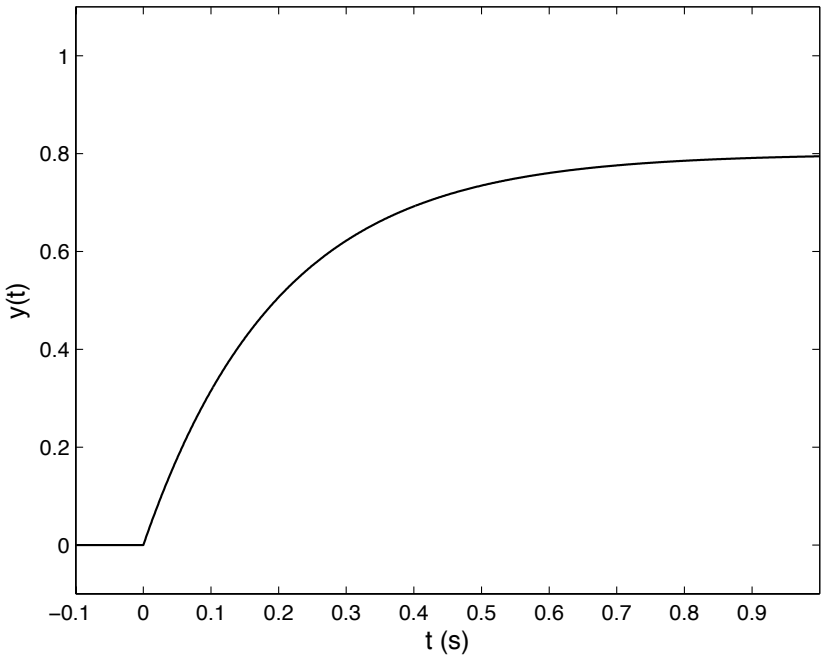


Figure N.4: Response of the system of Equation (N.19) with $a = 1$ and $k = 4$ to a unit step.

Which gives

$$H(s) = C(s)P(s) = \frac{aps + aq}{s^2 + as}. \tag{N.23}$$

Substituting (N.23) into (N.4) gives

$$G(s) = \frac{aps + aq}{s^2 + a(p+1)s + aq}. \tag{N.24}$$

The feedback with a PI controller has converted our first-order system into a second-order system with a frequency of

$$\omega = \sqrt{aq}$$

and a damping factor of

$$\zeta = \frac{a(p+1)}{2\sqrt{aq}} = \frac{(p+1)}{2}\sqrt{\frac{a}{q}}$$

.

Increasing the integral gain q increases the frequency of the response. Increasing the proportional gain p relative to $\omega \propto \sqrt{q}$ increases the damping factor. We will see how this is easy to visualize in the context of the open-loop frequency response of the system in Section N.5.

For a desired ω and ζ we can calculate the required proportional and integral gains as:

$$q = \frac{\omega^2}{a}. \tag{N.25}$$

and

$$p = \frac{2\omega\zeta}{a} - 1. \tag{N.26}$$

The response of a first-order system with a PI controller to a unit step for several values of ζ is shown in Figure N.5. With *critical damping*, $\zeta = 1$, the system responds quickly with no overshoot. Overdamping the system, $\zeta > 1$, gives a slow response. Underdamping the system, $\zeta < 1$, gives a more rapid response at the expense of overshoot, and as ζ gets smaller, ringing.

N.3 Second-Order System

A second-order system, like the car in our example, or the photovoltaic controller from Chapter **??** has a response that is determined by a second-order differential equation. A second-order system is described by two parameters: frequency ω and damping factor ζ. Its response to a unit step is an exponentially growing or shrinking sinusoid

$$y(t) = 1 - \sin(\omega_d t)\exp(-\zeta\omega t), \tag{N.27}$$

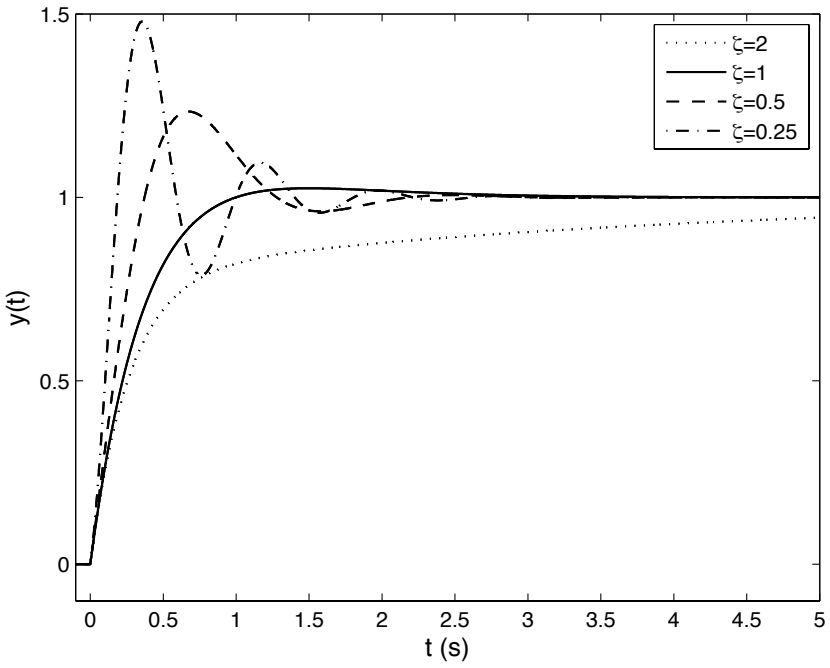


Figure N.5: Response of a first-order system with a PI controller (N.24) to a unit step with $a = 1$, $b = 3$, and $\zeta = 0.25, 0.5, 1$, and 2.

where $\omega_d = \omega\sqrt{1-\zeta^2}$.

As we shall see in the derivation below, we can stabilize a second-order system using a PD controller. The gains of this controller determine ω' and ζ' of the feedback system as:

$$\omega' = \sqrt{\omega^2 + p} \tag{N.28}$$

$$\zeta' = \frac{2\zeta\omega + r}{2\sqrt{\omega'}}. \tag{N.29}$$

These equations make it clear that increasing the derivative gain r increases the damping factor.

Again, if you are not familiar with Laplace transforms, you can stop reading this section here and simply use Equations (N.28) and (N.29) as a cookbook method to design controllers for second-order systems. If you are curious about their derivation, read on.

Let us consider first a second order system like our car that just integrates our input signal twice

$$P(s) = \frac{a}{s^2}. \tag{N.30}$$

From Equation (N.8) we know that applying a PD controller to this system gives

$$G(s) = \frac{H(s)}{1 + H(s)} = \frac{ap + ars}{s^2 + ars + ap}. \tag{N.31}$$

So we have $\omega = \sqrt{ap}$ and $\zeta = \frac{ar}{2\sqrt{ap}}$. Note the similarity of these terms to those derived for first-order systems but with the roles of the gains shifted. The extra $\frac{1}{s}$ in the plant makes the proportional gain term act like an integral gain term on a first order-system and the derivative gain term act like the proportional gain term on a first-order system.

For a more general second order system with a transfer function of

$$P(s) = \frac{b}{s^2 + as + b} \tag{N.32}$$

applying a PD controller gives

$$G(s) = \frac{brs + bp}{s^2 + (a + r)s + (b + p)}. \tag{N.33}$$

So we have $\omega = \sqrt{b + p}$ and $\zeta = \frac{a+r}{2\sqrt{b+p}}$. Note also that the DC gain of the system, is $\frac{bp}{b+p}$.

N.4 Controlling a Buck Converter

As a *green electronics* example of a second-order control system, consider a *buck* converter, like the one used in Chapter **??**, but where we are regulating the output voltage rather than the input voltage.

In this section we will examine two buck controllers. A voltage mode converter applies our techniques for generating second-order control systems to control output voltage v_o by manipulating the pulse width t_a. A more elegant solution is to use a layered control system where an inner loop controls the peak inductor current i_p and the output loop controls the resulting first-order system — manipulating i_p to control v_o.

N.4.1 Voltage-Mode Control

[Figure showing controller]

In a voltage mode controller, we control the output voltage v_o of the ocnverter by adjusting the duty factor d_a of switch a. A schematic of our regulator is shown in Figure **??**. A pulse-width modulator accepts our command for d_a and controls the switch a so it is on for a period of $t_a = d_a t_c$ each cycle, t_c.

The inductor current increases by $\Delta i_a = \frac{(V_i - V_o)t_a}{L}$ when a is on and then decreases by $\Delta i_b = \frac{V_o(t_c - t_a)}{L}$ when a is off. This gives a net change in inductor current for the cycle of

$$\Delta i = \frac{V_i t_a - V_o t_c}{L} \tag{N.34}$$

Approximating this difference equation as a differential equation gives

$$i = \int \frac{(V_i t_a - V_o t_c)}{t_c L} dt = \int \frac{V_i d_a - V_o}{L} dt. \tag{N.35}$$

The laplace transform of the current is

$$i(s) = d_a(s)\frac{V_i}{Ls} - v_o(s)\frac{1}{Ls}. \tag{N.36}$$

This current gets integrated on the output capacitor giving an overall transfer function of

$$v_o(s) = d_a(s)\frac{V_i}{LCs^2} - v_o(s)\frac{1}{LCs^2}. \tag{N.37}$$

which reduces to

$$v_o(s) = d_a(s)\frac{\frac{V_i}{LC}}{s^2 + \frac{1}{LC}} \tag{N.38}$$

We see that the open loop plant has a DC gain of $g = V_i$ a frequency of $\omega = \frac{1}{\sqrt{LC}}$ and a damping factor of $\zeta = 0$, i.e. the plant is unstable.

Parameter	Value	Units
L	18	μH
C	2	mF
V_i	170	V
t_c	10	μs

Table N.1: Paramter values for buck converter controller

With a PD controller generating d_a from an error signal comparing v_o to its desired value, the open-loop forward transfer function is

$$H(s) = \left(\frac{krs + kp}{s^2 + \frac{1}{LC}} \right). \tag{N.39}$$

where $k = \frac{V_i}{LC}$, and the closed-looop system response is given by

$$G(s) = \left(\frac{krs + kp}{s^2 + krs + kp + \frac{1}{LC}} \right). \tag{N.40}$$

Thus we have $\omega = \sqrt{\frac{pV_i+1}{LC}}$, $\zeta = \frac{kr}{2\omega}$, and a DC gain of $g' = \frac{kpLC}{kpLC+1}$.

Consider the parameter values in Table N.1, taken from an actual buck converter. Suppose we want our DC gain to be $g' = 0.995$ and a damping factor of $\zeta = 0.8$. We can solve for our controller parameters from the equations above. To get our DC gain, we need a proportional gain $p = 1.17$. This gives us a frequency of $\omega = 7.45 \times 10^4$r/s. Then to get the damping factor, we need the derivative gain to be $r = \frac{2\zeta\omega}{k} = 2.53 \times 10^{-5}$.

Figure N.6 shows the response of the buck converter using our controller with $p = 1.17$ and $r = 2.53 \times 10^{-5}$ to a step in target voltage (from 20V to 21V at 200μs.). The top trace shows the transition in output voltage v_o follows the step in target voltage with a 70μs delay. The middle trace shows the inductor current. When the output voltage is steady, the inductor is a sawtooth with amplitude of 10A. During the first half of the transition, i_L ramps up to over 50A and during the second half of the transition it ramps back down. It is the derivative feedback that starts the inductor current ramping down before the output voltage reaches the target level. Finally, the bottom trace shows the pulse width t_a. The controller here limits the pulse width to fall between 0 and 2.5μs. Both extremes are reached during the transient.

The high current transient seen by our voltage-mode controller could exceed the rated limits of some of the components in our buck converter. One way of limiting this transient is to use current-mode control as described below.

N.4.2 Current-Mode Control

[Figure showing current-mode control loop]

A buck-converter using current-mode control is illustrated in Figure ??. The controller consists of two nested loops. The inner loop terminates each PWM

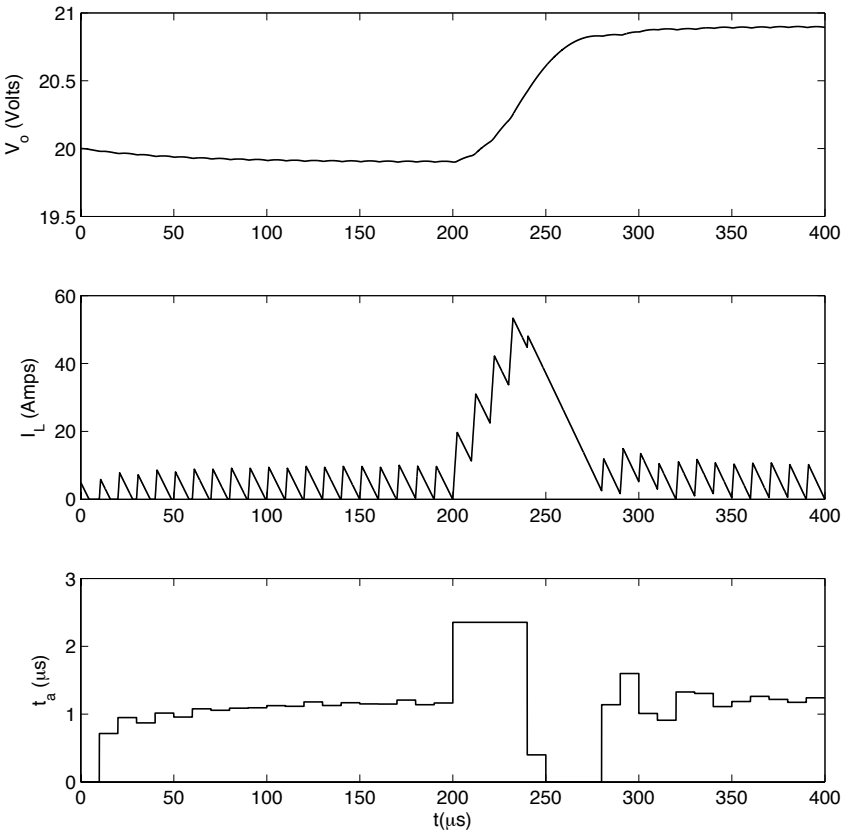


Figure N.6: Response of our example PD feedback buck converter controller a step in the target voltage.

pulse when the inductor current reaches a target $i_{\max}$. The outer loop computes $i_{\max}$ from the output voltage error $e = v_o - v_t$.

To simplify the analysis, assume the control variable sets average inductor current i. Then our system is a simple first-order integrator:

$$v_o = \int \frac{i}{C} dt, \tag{N.41}$$

$$v_o(s) = \frac{i(s)}{Cs} \tag{N.42}$$

If we apply a PI controller to this first order system we have

$$H(s) = \frac{ps + q}{Cs^2}. \tag{N.43}$$

$$G(s) = \frac{\frac{p}{C}s + \frac{q}{C}}{s^2 + \frac{p}{C}s + \frac{q}{C}}. \tag{N.44}$$

$$\omega = \sqrt{\frac{q}{C}} \tag{N.45}$$

$$\zeta = \frac{p}{2C\omega}. \tag{N.46}$$

If $C = 2$mF and we choose $\omega = 6.28 \times 10^4$ ($f = 10$kHz) and $\zeta = 1$ we get $q = 7.89 \times 10^6$ and $p = 251$.

Figure N.7 shows the response of this current-mode buck controller to a step in target voltage. To protect converter components, the controller is designed to limit the peak current to 30A. The top panel shows that the voltage ramps smoothly from 20 to 21V over $100\mu s$, as fast as is possible given the peak current limitation. The second panel shows the target peak current i_p computed by the PI controller which steps from 10A to 30A during the transition and then back to 10A. The actual inductor current waveform is shown in the third panel. The inner loop controls switch a to make this current ramp up to i_p each cycle in a sawtooth waveform. The pulsewidths produced by the inner loop are shown in the final panel.

Compared to the voltage-mode controller, the current mode controller prevents overstressing components with high current during transients, gives a crisper response, and is simpler to control and stabilize.

N.5 Frequency Response of Controllers

Considerable insight can be gained by looking at the frequency response of the forward transfer function, $H(s)$, of a feedback control system (i.e., the transfer function of the system with the feedback disconnected).

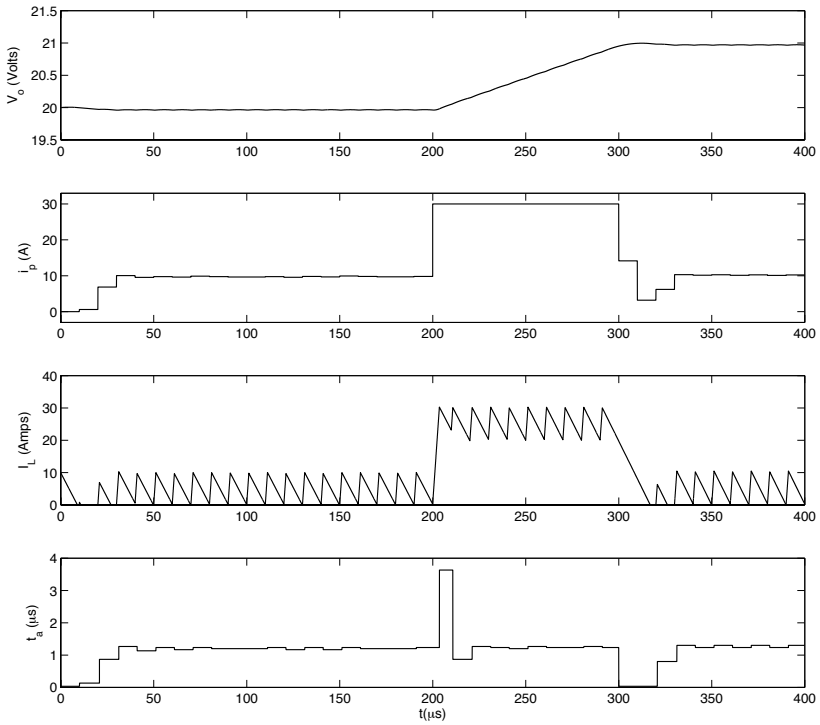


Figure N.7: Response of our example current-mode buck controller a step in the target voltage.

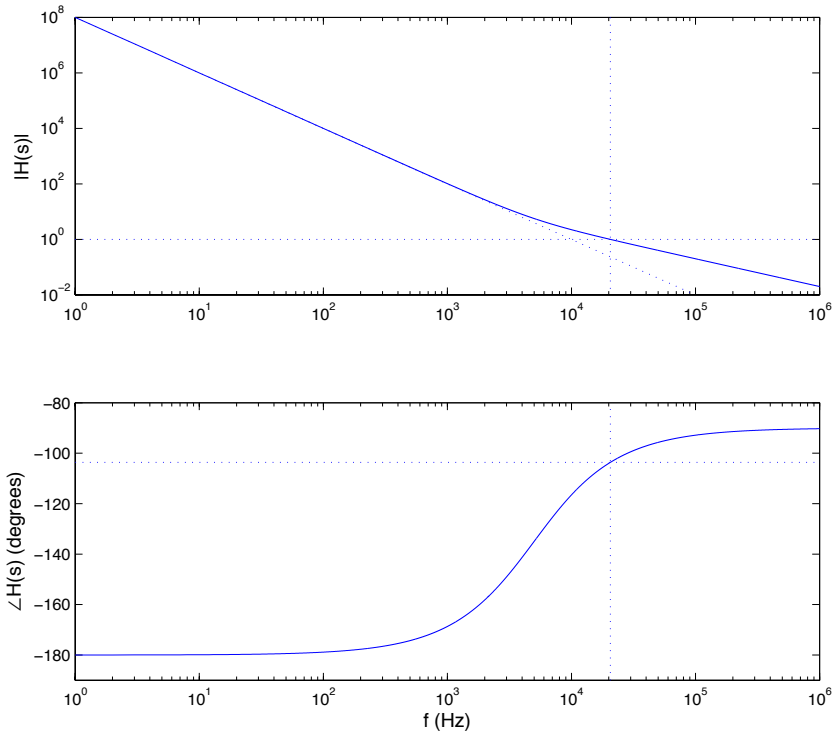


Figure N.8: Open-loop frequency response of the current-mode controller of Equation (N.43). The magnitude plot shows an undamped frequency of $f = 10$kHz and a unity-gain frequency of $f = 20.6$kHz. The phase plot shows a phase margin of 76°.

For example, Figure N.8 shows the open-loop frequency response of of the buck converter with current-mode control developed in Section N.4.2. Because of the s^2 in the denominator (two poles at the origin), the response starts with a phase of -180° and the magnitude falls off at 20dB per decade. The PI controller inserts a zero into the response at a frequency of $f = \frac{q}{2\pi p} = 5\text{kHz}$. This shifts the phase of the response to -90° (reaching -135° at 5kHz) at reduces the drop in magnitude to 10dB per decade.

The magnitude reaches unity-gain (shown by the dotted line) at $f = 20.6\text{kHz}$. The difference between the open-loop phase at this unity-gain frequency and -180° is called the *phase margin* of the system and determines the system stability. For our current-mode controller, the phase at unity gain is -104°. This gives a phase margin of $180 - 104 = 76°$ which corresponds to a critically-damped system ($\zeta = 1$).

Without the stabilizing pole, an extrapolation of the 20db per decade line reaches unity gain at 10kHz, the value we chose for ω, the undamped frequency.

As a second example, Figure N.9 shows the frequency response of the forward transfer function of our voltage-mode controller of Section N.4.1. Here our magnitude response starts flat with a DC gain of $pV_i = 199$ and a phase of 0. The tank circuit formed by the inductor and output capacitor introduce two poles at the resonant frequency of $f = \frac{1}{2\pi\sqrt{LC}} = 839\text{Hz}$. This pair of poles causes the phase to shift to -180° and causes the magnitude to peak and then fall off at 20dB per decade. The magnitude reaches unity-gain at 20.1kHz.

The derivative feedback inserts a zero into the response at $f = \frac{p}{2\pi r} = 7.4\text{kHz}$. This zero shifts the phase back to -90° to stabilize the system. The phase margin at the unity-gain frequency of 20.1kHz is $-110°$ corresponding to our choice of $\zeta = 0.8$.

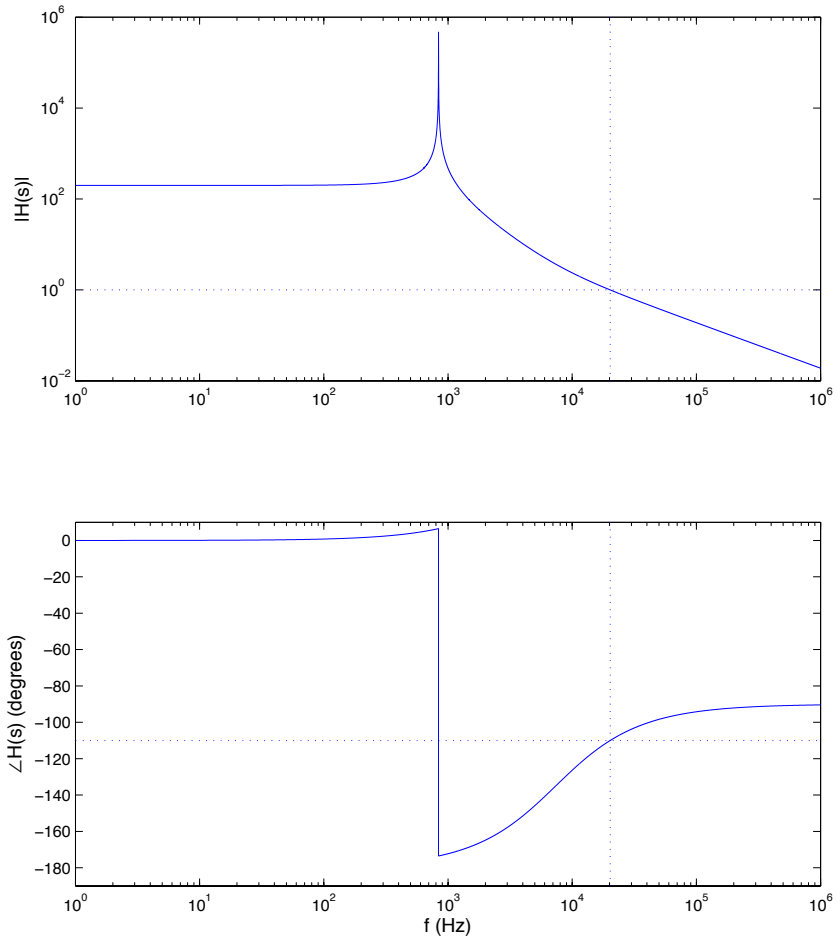


Figure N.9: Open-loop frequency response of the voltage-mode controller of Equation (N.39).

Appendix O

SPICE Simulation