

Electronics and Microprocessing for Research

Electronics and Microprocessing for Research:

You Can Make It

By

David Dubins

Cambridge
Scholars
Publishing



Electronics and Microprocessing for Research: You Can Make It

By David Dubins

This book first published 2018

Cambridge Scholars Publishing

Lady Stephenson Library, Newcastle upon Tyne, NE6 2PA, UK

British Library Cataloguing in Publication Data

A catalogue record for this book is available from the British Library

Copyright © 2018 by David Dubins

All rights for this book reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of the copyright owner.

ISBN (10): 1-5275-1333-5

ISBN (13): 978-1-5275-1333-4

To my sons Aidan and Remy, for inspiring and testing my projects;

To Andrew Cooper, for sparking my interest in electronics;

and to Lily, for supporting many late nights in the lab.



Epigraph by Remy Dubins

TABLE OF CONTENTS

Table of Contents	vii
List of Figures	xiii
List of Tables.....	xxii
Acknowledgements	xxv
Preface.....	xxvi
Section 0.....	1
Introduction and Course Objectives	1
Introduction	1
Course Objectives.....	2
Section 1.....	4
Introduction to Electricity	4
What is Electricity?	4
Charge	5
Voltage	6
Power	9
The Generalized Power Law.....	9
Resistance.....	10
Ohm's Law	10
Resistors	11
Measuring Voltage, Resistance, and Current.....	12
Using a Multimeter to Analyze Your Complicated Circuit	13
Measuring Overall Circuit Power Consumption and Overall Circuit Resistance	13
Electrical Ground	15
DC Ground	15
AC Ground	16
Different Ground Symbols	17
Types of Returns.....	17
Voltage Sources: Series vs. Parallel.....	19
Batteries in Series	19
Batteries in Parallel.....	20
Circuit Configurations	20
Kirchhoff's Voltage Law (KVL).....	21
The Voltage Divider Equation	24
Kirchhoff's Current Law (KCL).....	25
The Current Divider Equation	28
Calculating Current-Limiting Resistor Values for LEDs	30
Anode vs. Cathode: Devices with Polarity	31
Introduction to Switches.....	32
Breadboarding	33
Circuit Diagram Etiquette Example: Light Theremin	35
Activity 1-1: 9V Battery + LED + 10K Resistor.....	36

Activity 1-2: 9V Battery + 10K Resistor + 100K Resistor	38
Demo: Light Theremin	38
Learning Objectives for Section 1	39
Section 2.....	40
Capacitance, Power and Logic	40
Capacitors.....	40
Capacitor Circuit Diagram Symbols	41
Capacitor Ratings	42
Capacitors in Series and Parallel	42
Capacitors: Typical Uses	43
Capacitor Equations.....	44
Charging a Capacitor through a Resistor	44
Discharging a Capacitor through a Resistor	45
Voltage Divider Design: 10% Rule	46
Other Options for Delivering Lower Voltage	49
Datasheet Example: LM317 (Variable Linear Voltage Regulator)	49
How Hot Will My Chip Get? Heat Dissipation Calculations	52
Thévenin's Theorem.....	53
Thévenin's Theorem by Measurement (Using a Multimeter)	54
Mesh Current Method.....	57
Thévenin's Theorem Method (Theoretical)	60
Integrated Chips	63
PDIP/DIP.....	63
Surface Mount Technology	64
Logic Circuits.....	64
AND Gate: (e.g. 74HC08).....	65
OR Gate: (e.g. 74HC32).....	67
NOT Gate: (e.g. 74HC04)	68
Combining Logic Circuits	69
Activity 2-1: Capacitor Charging and Discharging	73
Activity 2-2: LM317 Voltage Regulator	74
Activity 2-3: Logic Gates	76
Learning Objectives for Section 2	77
Section 3.....	78
Introduction to Programming in the Arduino C++ Environment	78
Introduction to the Arduino Uno Microcontroller	78
Connecting a Serial LCD Module to the Arduino Uno.....	79
Your First Sketch.....	81
Basic Programming Concepts	82
Commenting Your Code.....	82
Storing and Accessing Data in Variables.....	85
Declaring and Using Variables.....	86
Integers.....	86
Long Integers.....	88
Arduino Global Space, Setup Function, and Loop Function	89
Float Variables.....	90

if...then...else Statements (and Logical Expressions).....	91
Boolean Variables.....	93
Boolean Operators	94
Byte Variables	95
String and Char Variables.....	95
Casting Variable Types.....	97
Arrays of Variables	98
Char Array.....	100
Data Types: More Complicated Conversions	100
Defining Programming Loops in Arduino.....	100
For Loops	100
C++ Shorthand Increment Expressions	103
Do... While Loops.....	103
While Loops	104
For, Do...While, or While?	105
The Break Command.....	106
Switch Case	107
General Programming Tips	108
Activity 3-1: Programming Challenge.....	109
Learning Objectives for Section 3	110
Section 4.....	112
Arduino Pins, and Writing Functions	112
Byte Variables and Digital Pins.....	112
What is a Digital Pin?.....	114
Digital OUTPUT Mode Example	115
Pulse Width Modulation (PWM) Example	116
Digital Input Mode Example	118
Analog Pins	120
Using Analog Pins as Digital Output Pins.....	121
Analog Read Example	121
External Analog Reference: AREF Pin	123
Arduino Pin Conflicts.....	124
Arduino Digital and Analog Pins: Summary Tables	124
The Serial Monitor	125
The Serial Plotter.....	126
Subroutines and Functions	127
Properties of Functions.....	127
Void Functions	128
Call-by-Value vs. Call-by-Reference.....	129
Float Functions	130
Integer (and other) Functions.....	132
Function DOs and DON'Ts	132
#define and #ifdef Statements	133
Activity 4-1: NTC Thermistor Circuit.....	135
Calibrating a Thermistor.....	136
Two-Term Exponential Thermistor Equation.....	137

Learning Objectives for Section 4	140
Section 5.....	142
Switching Higher Power Devices: Relays, Transistors, MOSFETs, TRIACs	142
Voltage and Current Limitations of the Arduino Uno	142
Relays.....	143
High Side Switching vs. Low Side Switching	146
Powering a Relay with a Separate Supply	147
Vin Pin: Arduino Uno.....	148
Diodes (P-N Junction, or Rectifier Diodes).....	148
Transistors.....	151
Bipolar Junction Transistors (BJTs)	151
NPN Transistors: Selecting a Base Resistor Value.....	153
NPN Transistors in the Active Region.....	155
Darlington Pairs.....	158
Current Gated vs. Voltage Gated.....	159
MOSFETs	159
N-Channel MOSFET Construction	160
2N7000 (N-Channel MOSFET).....	161
TRIACs	162
BT139-600E (TRIAC).....	163
Protecting your Circuit from DC Motors.....	164
Protection Diode.....	164
Reducing DC Motor Noise with Capacitors	165
Activity 5-1: Hot Plate Thermostat.....	166
Activity 5-2: Transistor as a Switch for a DC Motor.....	168
Parsing Serial Data	170
Activity 5-3: MOSFET as a Switch for a DC Motor	171
Learning Objectives for Section 5	172
Section 6.....	173
Process Control	173
When “Close Enough” Isn’t Close Enough.....	173
How a DC Motor Works	173
H-Bridge.....	174
L298N H-Bridge Motor Driver Module	176
Stepper Motors	178
Servo Motors	179
System Control Strategies	180
Open-Loop Control	180
Feed Forward Control	182
Feedback Control	184
Relay Switch Strategy	185
Proportional (P) Controller	186
Proportional-Integral (PI) Controller.....	190
Proportional-Integral-Derivative (PID) Controller	192
Combining Feedback Strategies	194
Activity 6-1: L298N Motor Driver Controlling a DC Motor.....	197

Activity 6-2: Stepper Motor	198
Activity 6-3: SG90 Servo Control	200
Activity 6-4: PID Control of a 12V CPU Fan.....	202
Learning Objectives for Section 6	203
Section 7.....	205
Operational Amplifiers.....	205
Introduction	205
Open Loop Configuration (Comparator)	206
Closed Loop Configuration	206
Buffer	207
Inverting Amplifier.....	211
Biasing the Output of an Inverting Amplifier	213
Non-Inverting Amplifier	215
Biasing the Output of a Non-Inverting Amplifier	216
Differential Amplifier.....	217
Summing Amplifier (Inverting).....	218
Summing Amplifier (Non-Inverting)	219
Summing Amplifier (Non-Inverting) Equations Solved:.....	220
Negative Voltage?	225
Solution 1: Using a Virtual Ground	225
Solution 2: Negative Voltage Generator.....	226
Op-Amps Can Do Calculus	227
Signal Attenuation: Reducing the Voltage	227
Activity 7-1: Load Cell Scale	228
Activity 7-2: pH Meter	232
Learning Objectives for Section 7	234
Section 8.....	235
Data Filtering, Smoothing, and Logging	235
Data Filtering.....	235
Low-Pass Filters (LPFs)	235
High-Pass Filters (HPFs)	240
Higher Order Filters	242
Band-Pass Filters	242
Second Order Low-Pass and High-Pass Filters	244
Operational Amplifiers: Practical Considerations	245
Impedance Considerations: Op-Amp Inputs.....	245
Impedance Considerations: Op-Amp Output.....	246
Measuring Output Impedance.....	248
Measuring Input Impedance	248
Practical Strategies to Reduce Signal Noise	249
Measuring Noise	251
Data Smoothing.....	252
Mean Filter	252
Median Filter	253
Mode Filter	254
Mean Filter with Threshold Rejection	254

Data Logging.....	257
Arduino TimeLib.h Library.....	258
Replacing delay().....	259
Notes about millis()	260
Logging through the Serial Port	260
Logging to an External microSD Card	261
Logic Shifters	261
Activity 8-1: Noise Reduction.....	263
Activity 8-2: Data Smoothing.....	263
Activity 8-3: Data Logging to an SD Card	264
Learning Objectives for Section 8	266
APPENDIX.....	267
Troubleshooting Guide.....	267
Troubleshooting Flowchart.....	270
Troubleshooting Zones	271
Serial.print(), Serial.println(), and Serial.write(): Escape Sequences and Advanced Formatting	271
Advanced String and Character Manipulation Commands.....	272
Additional String Conversion Commands	274
Arrays of Strings and Arrays of Char Arrays	274
ASCII Tables	276
Using Special Characters (Extended ASCII)	280
String and Char Arrays: Advanced Functions	281
Structures.....	283
Increment Operators as Array Index Values.....	285
Bitwise Operations	286
Bitwise AND (&).....	287
Bitwise OR ().....	288
Bitwise NOT (~).....	289
Bitwise XOR (^).....	289
Shifting Bits with “<<” and “>>”	290
Bitwise Operators: Short Forms	291
Port Manipulation.....	293
Interrupts	295
Tips to Optimize Sketch Memory.....	297
Ohm’s Law Equation Table.....	302
Common Fixed Resistor and Capacitor Values	303
.ino Files.....	304
Thermostat.ino (Section 5)	304
PID.ino (Section 6)	305
QuickStats.h (Section 8)	307
Derivation for $V_{in(+)}$ (Section 7).....	310
Arduino Uno Pin-out Diagram	312
List of Abbreviations.....	313
Bibliography.....	315
Index	320

List of Figures

- Figure 1-1. Mobile valence electron in outer shell of copper atom.
- Figure 1-2. Different ways of illustrating wire connections on a circuit diagram.
- Figure 1-3. A battery provides a constant voltage source.
- Figure 1-4. Conventional current vs. actual flow of electrons.
- Figure 1-5. Cathode and anode reactions of an alkaline battery.
- Figure 1-6. Circuit symbols and voltage vs. time diagrams for Direct Current (DC) and Alternating Current (AC) voltage sources.
- Figure 1-7. Using DC batteries in series.
- Figure 1-8. Resistance is the proportional current of electrons induced by a change in voltage. For an ohmic device like a fixed-value resistor, this relationship is linear.
- Figure 1-9. Circuit diagram symbols for fixed-value resistors.
- Figure 1-10. Measuring voltage (left), resistance (middle), and current (right) using a digital multimeter.
- Figure 1-11. Measuring the overall voltage, current, and power consumption of your circuit.
- Figure 1-12. Circular and linear methods of drawing the same circuit.
- Figure 1-13. AC household power outlet (North America).
- Figure 1-14. Common symbols for earth ground (left), chassis ground (middle), and floating ground (right).
- Figure 1-15. Floating return, drawn as it would be wired (left), using circuit diagram format (middle), and linear format (right).
- Figure 1-16. Chassis return. Note that optionally, the chassis can also be connected to earth ground to reduce shock hazard in case of an accidental short circuit.
- Figure 1-17. Earth return. Even though most battery-powered devices are floating, circuit diagrams tend to use the earth ground symbol for them regardless.
- Figure 1-18. Voltage is additive, and current capacity remains constant when batteries are wired in series.
- Figure 1-19. Current capacity is additive, and voltage remains constant when batteries are wired in parallel.
- Figure 1-20. Three ways of connecting electronic components: basic, in series, and in parallel.
- Figure 1-21. Kirchoff's Voltage Law applied to resistors in series.
- Figure 1-22. Derivation of the voltage divider equation.
- Figure 1-23. The voltage divider equation. The voltage across each resistor is proportional to its resistance compared to the total resistance, R_1+R_2 .
- Figure 1-24. Worked example for the voltage divider equation.
- Figure 1-25. Kirchoff's Current Law example.
- Figure 1-26. KCL applied to resistors in parallel. Note that the drawing on the left is equivalent to the drawing on the right electrically. Resistors in parallel can be

represented either way.

Figure 1-27. KCL for two resistors in parallel.

Figure 1-28. The current divider equation, for two resistors in parallel.

Figure 1-29. Worked example for the current divider equation.

Figure 1-30. Calculating the resistance of a current-limiting resistor for an LED. The LED symbol is labelled D_1 (for diode 1) in the circuit diagram.

Figure 1-31. LED & resistor circuit, with anodes and cathodes labeled.

Figure 1-32. Circuit diagram symbol (left) and example (right) of a latching on/off switch.

Figure 1-33. Momentary switch connections.

Figure 1-34. Breadboard internal wiring. Transparent arrows show how the power rails are connected by row, and middle pins are connected by column.

Figure 1-35. Light theremin circuit diagram. If a legend is used, values next to the circuit diagram symbols may be omitted.

Figure 1-36. Schematic for Activity 1-1. Note: the longer wire on the LED is the positive side (anode).

Figure 1-37. Schematic for Activity 1-2.

Figure 2-1. Electron build-up and flow upon capacitor charging and discharging.

Figure 2-2. Circuit symbols for different types of capacitors.

Figure 2-3. Electrolytic (left) and ceramic (right) capacitors.

Figure 2-4. Calculating total capacitance of capacitors in parallel and in series.

Figure 2-5. A charge-discharge circuit. Holding down SW_1 charges the capacitor. Once charged, holding down SW_2 discharges the capacitor.

Figure 2-6. Capacitors can remove high frequencies from a signal (e.g. in an RC high-pass filter), low frequencies (e.g. in a CR low-pass filter), smooth out an AC signal to DC, and reduce fluctuations in a noisy power supply.

Figure 2-7. A circuit to illustrate charging and discharging a capacitor through a resistor.

Figure 2-8. It takes about three time constants ($3 \times \tau$) for a capacitor to charge through a resistor.

Figure 2-9. It takes about three time constants ($3 \times \tau$) for a capacitor to discharge through a resistor.

Figure 2-10. Solving for R_2 using the voltage divider equation.

Figure 2-11. Calculating the value of the bleed resistor, using the 10% Rule.

Figure 2-12. Solved circuit using the 10% Rule.

Figure 2-13. Split supply.

Figure 2-14. LM317 pin-out.

Figure 2-15. LM317 used as an adjustable regulator circuit with improved ripple rejection.

Figure 2-16. LM317 used as a current limiter.

Figure 2-17. Circuit diagram symbol for a current source.

Figure 2-18. Heat sink for TO-220 package. The silicon layer and plastic nut keep the body of the package insulated from the heat sink, to reduce the chances of a short circuit.

Figure 2-19. Thévenin's Theorem states that any complicated network of resistors, capacitors, and sources from the perspective of a single component (at connections a and b) may be represented as a voltage source in series with a resistor.

Figure 2-20. Example circuit for Thévenin's Theorem. First step: identify two terminals of interest, and label them a and b.

Figure 2-21. Remove the load from the example, and measure V_{ab} .

Figure 2-22. Measure i_{sc} across terminals a and b.

Figure 2-23. Thevenin Equivalent Circuit.

Figure 2-24. Thevenin Equivalent Circuit with load replaced.

Figure 2-25. Short all voltage sources, remove all current sources, then measure R_{TH} .

Figure 2-26. Norton Equivalent Circuit.

Figure 2-27. Example for the Mesh Current Method.

Figure 2-28. Number each inside loop, and label each junction.

Figure 2-29. Perform KCL on junction A.

Figure 2-30. Perform KCL on junction B.

Figure 2-31. Identify points a and b around the load.

Figure 2-32. Remove the load, and calculate the voltage difference between points a and b.

Figure 2-33. Circuit diagrams can be deceptive. These four circuits all depict two resistors in parallel, from points a to b.

Figure 2-34. Thévenin Equivalent circuit, with load replaced.

Figure 2-35. DIP chips are great for prototyping with breadboards.

Figure 2-36. Pin numbering for DIP chips runs counterclockwise, starting from the bottom left pin.

Figure 2-37. Logic symbol for an AND gate.

Figure 2-38. Pinout diagram for the 74HC08 AND chip.

Figure 2-39. Schematic to test out the functionality of an AND gate. Pull-down resistors protect against floating pin states when the DIP switches are open.

Figure 2-40. Venn diagram for a 2-input AND gate.

Figure 2-41. Logic symbol for a 2-input OR gate. The Y-terminal end of the gate can also be depicted as being more pointed than round.

Figure 2-42. Pinout diagram for the 74HC32 OR chip.

Figure 2-43. Schematic to test out the functionality of an OR gate.

Figure 2-44. Venn diagram for a 2-input OR gate.

Figure 2-45. Logic symbol for a NOT gate.

Figure 2-46. Pinout diagram for the 74HC04 NOT chip.

Figure 2-47. Schematic to test out the functionality of a NOT gate.

- Figure 2-48. Venn diagram for a NOT gate.
- Figure 2-49. An AND gate combined with a NOT gate is equivalent to a NAND gate.
- Figure 2-50. Venn diagram for a 2-input NAND gate.
- Figure 2-51. An OR gate combined with a NOT gate is equivalent to a NOR gate.
- Figure 2-52. Venn diagram for a 2-input NOR gate.
- Figure 2-53. XOR gate, meaning “exclusive OR”.
- Figure 2-54. Venn diagram for a 2-input XOR gate.
- Figure 2-55. XNOR gate, meaning “NOT exclusive OR”, or “exclusive NOR”.
- Figure 2-56. Venn diagram for a 2-input XNOR gate.
- Figure 2-57. Symbolic logical representation of Example 1(a).
- Figure 2-58. Venn diagram for Example 1(a).
- Figure 2-59. Symbolic logical representation of Example 2(a).
- Figure 2-60. Venn diagram for Example 2(a).
- Figure 2-61. Circuit diagram for Activity 2-1.
- Figure 2-62. Schematic for Activity 2-2. Note: Pin 3 of the 1K trim is not connected.
- Figure 2-63. Circuit diagram for Activity 2-3 (from Figures 2-38 and 2-39).
- Figure 3-1. Arduino Uno R3 board layout.
- Figure 3-2. Connecting the MCU to a serial LCD module.
- Figure 3-3. MCU connected to serial LCD module and laptop.
- Figure 3-4. Arduino IDE sketch window.
- Figure 3-5. Our first Arduino Uno sketch, properly commented.
- Figure 4-1. Digital and analog pins of the Arduino Uno MCU.
- Figure 4-2. Connecting an LED to digital pin 6, using a breadboard.
- Figure 4-3. Circuit diagram for schematic in Figure 4-2.
- Figure 4-4. Pulse width modulation (PWM), illustrated with different duty cycles.
- Figure 4-5. Gaining and shifting the sin function to the working PWM range [0-255].
- Figure 4-6. LED circuit (left) and momentary switch circuit (right).
- Figure 4-7. LM35 temperature sensor.
- Figure 4-8. A potentiometer can be set up as a variable resistor (rheostat), or a voltage divider. The wiper (middle pin) sweeps across a length of resistive material.
- Figure 4-9. LED circuit (left) and potentiometer set up as a voltage divider (right). As we are reading the voltage from the potentiometer, it is also called a rotary encoder in this context.
- Figure 4-10. The AREF pin lets you set the voltage that the highest div (1023) is equal to (range: 0-5V).
- Figure 4-11. Thermistor + LED circuit. Resistor R_1 is the sense resistor.
- Figure 5-1. A high voltage induces a magnetic field, pulling a switch closed.
- Figure 5-2. Anatomy of a relay circuit diagram symbol (Single Pole, Double Throw - SPDT).

Figure 5-3. Two common types of relays—SPDT, and DPDT.

Figure 5-4. Internal circuit diagram of a SPDT relay module, with an opto-coupler isolating the Arduino Uno from the relay supply. Some relay modules allow for a separate ground for the supply powering the relay, although in the lab we will power the relay module using the Arduino +5V pin.

Figure 5-5. Using a SPDT relay as a normally open (NO) switch vs. a normally closed (NC) switch. The switch terminals have been sketched into the relay module boxes to help illustrate switching direction.

Figure 5-6. The relay module on the left is a high side switch (above the load). The relay module on the right is a low-side switch (below the load).

Figure 5-7. The relay module on the left is powered using the Arduino Uno, which can introduce a lot of switching noise to your measurements. The relay module on the right is powered using a separate +5V DC adapter.

Figure 5-8. The Vin pin gives you access to the power supply used to power the Uno, before the on-board 5V voltage regulator.

Figure 5-9. Diode symbols: LED (left), and general diode symbol (right).

Figure 5-10. Silicon has 4 outer valence electrons. When doped with boron (3 outer valence electrons), a space is formed capable of temporarily accepting an electron (left). When doped with phosphorus (5 outer valence electrons), there is an extra electron, capable of flowing (right).

Figure 5-11. When conventional current flows from anode to cathode, a diode is forward biased, and electrons can jump across the P-N junction.

Figure 5-12. When conventional current flows from cathode to anode, a diode is said to be reverse biased, and electrons can no longer cross the P-N junction.

Figure 5-13. 1N4007 diode. A stripe indicates the brick wall (location of N-terminal).

Figure 5-14. Circuit to illustrate how a transistor works. Q₁ is a 2N2222 NPN transistor in a TO-92 package.

Figure 5-15. Worked example for the 10% Current Rule.

Figure 5-16. An NPN transistor configured as a common emitter amplifier.

Figure 5-17. Calculating the voltage drops around the three terminals of a bipolar junction transistor used as a common emitter amplifier in Figure 5-16.

Figure 5-18. A Darlington pair of NPN transistors.

Figure 5-19. An LED as a light sensor.

Figure 5-20. When V_{TH} is applied to the gate terminal, an N-channel MOSFET allows conventional current to flow from drain to source.

Figure 5-21. Pin-out diagram for 2N7000 N-channel MOSFET.

Figure 5-22. Circuit diagram symbol for a TRIAC.

Figure 5-23. A TRIAC, controlled by microcontroller digital pin.

Figure 5-24. Forward phase dimming (left) and reverse phase dimming (right) with a TRIAC. The TRIAC is switched on and off strategically to chop an AC supply into narrower widths to dim a resistive load.

Figure 5-25. Protection diode on a DC motor.

Figure 5-26. Noise-reducing capacitor on “Bob”, the laboratory video rover. These capacitors fixed his nasty resetting problem.

Figure 5-27. Circuit diagram for Activity 5-1: hot plate thermostat.

Figure 5-28. Schematic for Activity 5-2: NPN transistor-controlled DC motor.

Figure 5-29. Floppy drive connector of an ATX power supply, capable of supplying up to 3 amps. ATX power supplies are salvageable from old desktop computers.

Figure 5-30. Circuit diagram for Activity 5-3: MOSFET-controlled DC motor.

Figure 6-1. The rotor coil of a motor (left) will spin according to Fleming’s left hand rule (right).

Figure 6-2. Simple H-Bridge configuration.

Figure 6-3. Protection diodes for an H-bridge.

Figure 6-4. L298N H-bridge module. This module comes with an on-board regulator that you can also use to power the logic side of your circuit. Check the top underside of the module to confirm pin and screw terminal identities, as some modules may vary.

Figure 6-5. Pin jumpers in the open position (top), and in the closed position (bottom), L298n module.

Figure 6-6. A Nema17 bipolar stepper motor.

Figure 6-7. Circuit diagram symbols for unipolar (left) and bipolar (right) stepper motors.

Figure 6-8. SG90 hobby servo motor.

Figure 6-9. Functional block diagram of toaster control system.

Figure 6-10. Toaster setting dial—an example of open-loop control.

Figure 6-11. Adding a system to a sensor on the INPUT stream is a feed forward control strategy.

Figure 6-12. Example of a feed forward control system.

Figure 6-13. Example of a feedback control system.

Figure 6-14. Example of a proportional feedback control system.

Figure 6-15. Undamped feedback response.

Figure 6-16. Over-damped feedback response.

Figure 6-17. Under-damped feedback response, illustrating the concepts of rise time, overshoot, and settling time.

Figure 6-18. Finding a critically-damped feedback response.

Figure 6-19. Over-damped feedback response illustrating the effects of an integral gain. This response overshoots a little, because of integral wind-up.

Figure 6-20. Adjusting the derivative gain to attain the setpoint.

Figure 6-21. A simple PID control algorithm.

Figure 6-22. Circuit diagram for Activity 6-1.

Figure 6-23. Circuit diagram for Activity 6-2.

Figure 6-24. Circuit diagram for Activity 6-3.

Figure 6-25. Circuit diagram for Activity 6-4.

Figure 7-1. Circuit diagram symbol for an operational amplifier.

Figure 7-2. Open-loop configuration of an operational amplifier.

Figure 7-3. Buffer (or voltage follower) configuration of an op-amp.

Figure 7-4. The voltage output of an ideal buffer follows the voltage input ($V_{in}=V_{out}$).

Figure 7-5. Maximum peak output voltage vs. frequency for the TLO7x op-amp series ($R_L=2k\Omega$, $T=25^\circ\text{C}$).

Figure 7-6. The op-amp on the right can swing higher because a higher voltage is provided to the power rails.

Figure 7-7. Op-amp headroom means that the output of the op-amp can't swing all the way to the power rails. This is an illustration of the TLO7x series, when the op-amp is supplied with $\pm 5\text{V}$.

Figure 7-8. The output of a typical op-amp will not be able to swing all the way to ground, if the negative rail is connected to ground.

Figure 7-9. Rail-to-rail op-amps are able to swing their output within microvolts of the power rails (virtually no headroom).

Figure 7-10. Inverting amplifier configuration of an op-amp.

Figure 7-11. An op-amp's V_{out} is constrained by the voltage connected to its power rails.

Figure 7-12. Input and output of an inverting amplifier (gain $=-2$).

Figure 7-13. Configuration (left) and performance (right) of an inverting amplifier (gain $=1$).

Figure 7-14. Biasing an inverting amplifier.

Figure 7-16. Configuration (left) and example performance (right) of a non-inverting amplifier.

Figure 7-17. Calculating V_{out} for a non-inverting amplifier.

Figure 7-18. Configuration (left) and example performance (right) of a non-inverting operational amplifier, with a bias voltage.

Figure 7-19. Differential amplifier configuration (left) and equations (right).

Figure 7-20. Calculating V_{out} example for a differential amplifier.

Figure 7-21. Inverting summing amplifier configuration (left) and equations (right).

Figure 7-22. Calculating V_{out} example for an inverting summing amplifier.

Figure 7-23. Non-inverting summing amplifier configuration (left) and equations (right).

Figure 7-24. Calculating V_{out} example for a non-inverting summing amplifier.

Figure 7-25. This non-inverting summing amplifier shifts and gains the signal, V_{in} .

Figure 7-26. Voltage at the non-inverting input, $V_{in(+)}$.

Figure 7-27. Shifted and gained pH electrode voltage signal.

Figure 7-28. Solved non-inverting summing amplifier for pH meter.

Figure 7-29. The bias voltage is generated using a voltage divider, and then buffered before adding it to the (buffered) probe voltage.

Figure 7-30. How do you supply negative volts?

Figure 7-31. Splitting a power supply (top) and using a voltage divider (bottom) for supplying a negative volts to the bottom rail of an op-amp.

Figure 7-32. ICL7760 wired as a negative voltage generator.

Figure 7-33. Differentiator and integrator op-amp configurations.

Figure 7-34. Attenuating a probe signal using a voltage divider, then buffering the output.

Figure 7-35. Structure of a Wheatstone bridge.

Figure 7-36. Circuit diagram for Activity 7-1 (load cell scale).

Figure 7-37. Experimental setup for Activity 7-1 (load cell scale).

Figure 7-38. Circuit diagram for Activity 7-2 (pH meter).

Figure 8-1. The effect of a carefully designed low-pass filter (LPF).

Figure 8-2. Bode Magnitude Plot for a first-order LPF.

Figure 8-3. Calculation of the gain of a LPF, one decade higher than the cutoff frequency.

Figure 8-4. Worked example for a LPF, $f_c=200$ Hz. Left: passive LPF, right: active LPF with unity gain.

Figure 8-5. The effect of a carefully designed high-pass filter (HPF).

Figure 8-6. Bode Magnitude Plot for a first-order HPF.

Figure 8-7. An example band-pass filter (top) and Bode Magnitude Plot (bottom).

Figure 8-8. Twin T Notch Filter (left) and Bode Magnitude Plot (right).

Figure 8-9. Passive second-order HPF.

Figure 8-10. Non-inverting, amplifying second-order LPF.

Figure 8-11. Properties of an ideal op-amp.

Figure 8-12. Balancing the inputs of an inverting op-amp with a compensating resistor.

Figure 8-13. Matching the output impedance of an op-amp with the input impedance of the next stage (in this case, the Uno's analog pin).

Figure 8-14. Ironically, this newspaper did not start on January 1st, 1970.

Figure 8-15. The 16 MHz crystal oscillator on the Arduino Uno, responsible for microprocessor speed.

Figure 8-16. RTC module for keeping track of epoch time.

Figure 8-17. Circuit diagram (left) of a logic shifter (photo right), safely bridging a 3.3V microSD card module to the hotter 5V logic of the Arduino Uno.

Figure A-1. Troubleshooting flowchart: developing resilience.

Figure A-2. Potential areas to consider troubleshooting. Many problems are multifaceted and involve more than one area.

Figure A-3. Momentary switch connected to INT0 (pin 2).

Figure A-4. Voltage divider network at an op-amp input: solving for the input voltage.

Figure A-5. Pin-out diagram for the Uno. Pin numbers next to pins correspond to the pin numbers on the ATmega328P-PU chip.

List of Tables

Table 1-1. Typical voltages and current capacities for various battery types and chemistries.

Table 1-2. Current capacity for common wire gauges.

Table 1-3. Summary chart: resistors in series vs. resistors in parallel.

Table 1-4. Measured and calculated values for Activity 1-1.

Table 1-5. Measured and calculated values for Activity 1-2.

Table 2-1. Recommended operating conditions for the LM317 voltage regulator (TO-220 package).

Table 2-2. Some electrical characteristics of the NE555 chip.

Table 2-3. Logic tables for a 2-input AND gate. “0V” is the same as “ground”, and this table assumes that logic level for your circuit is +5V.

Table 2-4. Two-input AND logic table.

Table 2-5. Two-input OR logic table.

Table 2-6. NOT logic table.

Table 2-7. Two-input NAND logic table.

Table 2-8. Two-input NOR logic table.

Table 2-9. Two-input XOR logic table.

Table 2-10. Two-input XNOR logic table.

Table 2-11. Logic table for Example 1(a).

Table 2-12. Logic table for Example 2(a).

Table 2-13. Experimental results from Activity 2-1.

Table 3-1. Bit depth size chart.

Table 3-2. Mathematical operators in C++.

Table 3-3. Logical expressions in C++.

Table 3-4. Table of boolean operators.

Table 3-5. How to cast between common variable types, with examples.

Table 3-6. More complicated variable conversions involving String and char.

Table 3-7. C++ shorthand.

Table 3-8. Three major types of loops performing the same task.

Table 4-1. Wiring a momentary switch to a digital input pin.

Table 4-2. Digital pin summary table.

Table 4-3. Analog pin summary table.

Table 4-4. Selectable baud rates in the Arduino IDE serial monitor.

Table 4-5. Common types of functions.

Table 4-6. Example sketches for call-by-value (left) vs. call-by-reference (right).

Table 4-7. DOs and DON'Ts in writing functions.

Table 4-8. Advantages and disadvantages of thermistors.

Table 4-9. Table for Experimental Results in Activity 4-1.

Table 5-1. NPN and PNP Bipolar Junction Transistors (BJTs).

Table 5-2. h_{FE} values of P2N2222A (TO-92).

Table 5-3. N-Channel MOSFET vs. P-Channel MOSFET.

Table 6-1. An H-bridge can switch the polarity across a motor by changing the states of four switches.

Table 6-2. L298N H-bridge control of two independent DC motors.

Table 6-3. The process of toasting bread in a toaster, described without (left) and with (right) control system terminology.

Table 7-1. Calculating the expected op-amp output across the expected input range.

Table 7-2. Gaining and shifting a pH probe signal to a convenient range for the `analogRead()` function.

Table 8-1. First-order passive and active low-pass filters.

Table 8-2. First-order passive and active high-pass filters.

Table 8-3. Measuring the output impedance of a signal.

Table 8-4. Measuring the input impedance of the input pin of a device (e.g. a microprocessor).

Table 8-5. Ten practical tips for noise reduction and prevention.

Table 8-6. Calculating the median of a data set.

Table 8-7. Calculating the mode of a data set.

Table 8-8. Comparison of four data smoothing methods.

Table A-1. Troubleshooting questions and suggested follow-up actions.

Table A-2. Some C++ escape sequences you can use in strings and char variables.

Table A-3. Custom output formats for `Serial.print()` and `Serial.println()`.

Table A-4. Additional string commands that can help pre- or post-process strings.

Table A-5. Functions to convert other variable types to Strings.

Table A-6. ASCII table, with character, decimal, hexadecimal, and binary codes.

Table A-7. Commands that test the contents of a char variable.

Table A-8. Example bitwise AND comparison: finding out a pin state.

Table A-9. Example bitwise OR comparison: setting a pin HIGH.

Table A-10. Example bitwise AND comparison: setting a pin LOW.

Table A-11. XOR logic table.

Table A-12. Example bitwise XOR comparison: toggling a pin state HIGH.

Table A-13. Example bitwise XOR comparison: toggling a pin state LOW.

Table A-14. Boolean operators vs. bitwise operators in C++.

Table A-15. Short forms for common bitwise operations, with examples.

Table A-16. ATmega328P pin banks.

Table A-17. Trigger options for interrupt service routines for the Arduino Uno.

Table A-18. Variable sizes in bits, and the ranges of values they can store.

Table A-19. Some microprocessors and their memory limits.

Table A-20. Ohm's Law equations, re-arranged for each term.

Table A-21. Common fixed resistor values.

Table A-22. Common fixed capacitor values.

ACKNOWLEDGEMENTS

Many people assisted in the inception and development of this text.

Dr. Robert B. Macgregor, Jr. inspired this work by encouraging me to turn a fixed USP dissolution apparatus into an undergraduate course, and he helped me launch it within a short time span of six months.

Andrew Cooper taught me much of the tacit and practical knowledge, for example, *Practical Strategies to Reduce Signal Noise* (Table 8-5) and providing a virtual ground for a probe (Figure 7-31).

Dr. Will Cluett kindly reviewed and provided comments for Section 6.

There are a number of students who in taking this course have helped to refine and improve this work immensely: Celeste Vicente, Brittany Epp-Ducharme, and Nabeel Tariq. Your enthusiasm and keen eyes for detail are warmly appreciated.

Lastly, I would like to thank the institution in which I work, the Leslie Dan Faculty of Pharmacy at the University of Toronto for providing a positive and supportive educational environment and infrastructure, and plenty of batteries.

PREFACE

It's not supposed to work.

One of the things I hope you will discover throughout this course is that we take our technology for granted. This attitude is deeply embedded in our consumer culture. As end users, we are intentionally blinded to failures during product development. We demand quality, taking good design for granted. We purchase the latest gizmos at our favourite technology stores, and after a year or two of using them, either they break and we buy new ones, or we toss them aside because we are bored and done with them, wanting newer gizmos. We are *accustomed* to the idea that a device is supposed to work. Why shouldn't it? After all, we paid good money for it. It comes in a box, with a guarantee. If it doesn't work, we get a refund after a flustered sales person squints apologetically at the original store receipt.

These gizmos *work* because of careful design, quality parts, automated assembly processes, meticulous QC checks, and market research—something that engineers, scientists, manufacturers, and business people spend their entire careers working on. As a novice then, you will very likely become frustrated early on that the circuit you are building does not light up an LED, measure the temperature, or send text to a screen. Perhaps little components will produce strong-smelling smoke. Although a lay person might call this “broken”, “screwed up”, or “horrible” and feel discouraged about their abilities, this frustration is more appropriately called the *design and testing* phase—where things necessarily won't work. Why should they? The circuit doesn't even exist yet, let alone function flawlessly enough to sell in a store.

As a designer then, it is your job to put on an optimistic hat, and sharpen your troubleshooting skills. I can promise you that during the activities in this course, you will hit a dead end where you feel that you have tried everything you can, yet your circuit still doesn't work. You might feel frustrated, discouraged, or defeated. I can promise you this because it happens to me all the time.

However, I can also promise that if you hang in there, when you finally do figure out a critical piece of the puzzle that makes your project spring to life, all of that frustration will dissipate (after the urge to kick yourself passes). It will be replaced by a feeling of relief, satisfaction, pride, happiness, and perhaps the thrill that addicted me to this area of study in the first place. Electronics are SO COOL!!!

SECTION 0

INTRODUCTION AND COURSE OBJECTIVES

Introduction

We are taught at a very early age to start counting at 1. However, with computer programming in microprocessing, we need to get into the habit of starting to count from zero. It is only fitting then to begin the introduction of this course with Section 0 accordingly—starting *ex nihilo*, from nothing.

As scientists, we tend to use very sophisticated equipment, containing circuits that we barely understand and take for granted. Understanding and designing your own electronic circuits can allow you to control equipment on the microsecond scale. With sensors and switches, you can use electronics to build your own scientific equipment, and to record measurements. This course will introduce you to programming, electronics, data acquisition, system control strategies, and data analysis techniques.

The goal of this course is to introduce you to theoretical and applied concepts in electronic circuitry, for the purpose of collecting and analyzing experimental data. As the course was developed in the Leslie Dan Faculty of Pharmacy, many of the examples are rooted in pharmaceuticals, the science involved in drug formulation design. The concepts discussed are nonetheless applicable to all quantitative research contexts where measurement and data collection are important. The course uses mixed teaching methods, with the first portion of each lecture as small-group didactic teaching, and the second portion as laboratory exercises to experiment with and illustrate the concepts covered. The course discusses introductory circuit design, with an emphasis on how common components work (e.g. resistors, capacitors, diodes, transistors, operational amplifiers, and a variety of sensors) in scientific and manufacturing instrumentation.

Practical and mathematical aspects of circuit design are discussed (e.g. Ohm's Law, voltage dividers, analog *vs.* digital signals). There is a heavy emphasis on programming in C++, taught from an introductory level, which complements learning activities.

With the recent advent of low-cost, consumer-level microprocessors (e.g. ATtiny, Arduino, Raspberry Pi, ESP8266, and new platforms continually being developed), affordable and accessible microprocessing has empowered researchers with resources to take experimental designs to new heights. Such microprocessors are relatively simple compared to the

complexities of today's computers; however, are more than powerful enough and fast enough to control sophisticated equipment such as scientific instrumentation and 3D printing. Previously, DACs (Digital-to-Analog Converters) were thousands of dollars, requiring high programming aptitude to bridge the gap between instrument and computer. Serial communication ports were reliable only at slower speeds (e.g. 1200 bps). Serial communication was finicky, requiring access to equipment subroutines not always readily available. However, the climate has now changed for experimental design. Libraries are readily available, interfaces are more intuitive, and a large open source community has evolved to support scientists and hobbyists alike. Knowledge of programming and circuitry will provide a solid foundation not only in experimental design and analysis for research, but in many other areas as well.

The development and commercialization of prototype boards such as Arduino, Raspberry Pi, and ESP8266 have empowered the electronics community with quicker and easier circuit prototyping. Due to economies of scale, electronic components have become very inexpensive over websites such as eBay, Alibaba, and Amazon. The electronics hobbyist demand has driven the development of modular electronic components marketed for general purposes, compatible with open-source platforms at +3.3V and +5V logic levels (e.g. opto-isolator power relay circuits, H-bridge motor controllers, and frequency-matched RF transmitters/receivers). Circuits that would previously need to be built from scratch are readily available and packaged as low-cost, ready-to-use modules. This course will examine some of these modules and their usefulness in circuit design.

Course Objectives

This course is broken up into eight sections, which loosely follow different “themes” in electronics. Learning objectives are defined at the end of every section to help guide your learning. Over-arching these learning objectives are the following course objectives.

After taking this course, you will ...

- 1) Be able to properly interpret other people's circuit diagrams and draw your own diagrams in a manner that will enable you to accurately record and share your work with others. You will be able to draw circuit diagrams with enough detail so that others can build the same project. You will develop a foundation for “circuit diagram literacy”. Many diagrams in this manual have been intentionally hand-drawn to reinforce that drawing a circuit diagram doesn't require circuit diagram-drawing software.

- 2) Have a working knowledge of some basic building blocks in electronics: resistors, capacitors, relays, transistors, MOSFETs, op-amps, voltage regulators, etc. This course will not expose you to all of the fundamentals (e.g. inductors, Zener diodes, transformers, and diode bridges have been intentionally omitted). However, the course provides enough background to start you off designing your own circuits, hopefully pointing you in a helpful direction. Whereas cook follows recipes, a chef creates them. Understanding how electronic building blocks fit together will enable you to go beyond following other people's circuit diagrams, synthesizing your own modules and ideas.
- 3) Be able to program in C++, which is a wonderful segue into learning other programming languages. Once you learn how to code in one language, learning another is often a matter of translation, which is much faster to pick up. You will learn the basics of writing a computer program, and have opportunities to write and compile your own code. You will learn proper programming etiquette, including commenting, selecting and working with appropriate variable types, writing subroutines, functions, and declaring local and global variables. After you are finished this course, you will be able to write, compile, and upload code for an Arduino microprocessor. These skills are transferrable to programming other microprocessors (e.g. the ATtiny85).
- 4) Be able to design, create, assemble, and test your own projects and scientific equipment, from raw sensor output to data filtering and recording.
- 5) Be able to develop and refine your troubleshooting skills, and become more confident in your circuit building and testing abilities. You will be empowered to solve your own problems, and recognize the value and complexity in the commercial electronics around you.
- 6) Be able to design and build circuits safely and carefully, always being mindful of the dangers of high voltage.
- 7) Have fun with electronics and programming!

The activities in this course have been carefully prepared assuming that you will work independently. There will be lots of opportunities to help troubleshoot each other's work; however, you are expected to perform these exercises on your own to help develop your skills at circuit designing and programming.

SECTION 1

INTRODUCTION TO ELECTRICITY

What You'll Be Learning	Lecture: Introduction to electricity and circuit diagrams. Ohm's Law, General Power Law, Power (AC vs. DC). Voltage, current, Resistance, and how to measure them. Electrical ground. Kirchhoff's Laws. Voltage and current dividers. Anode vs. cathode. Switches. Introduction to breadboarding.
What You'll Be Doing	Activity 1-1: 9V LED - resistor circuit. Measuring voltage, current, resistance. Calculating power. Adding a momentary switch. Activity 1-2: Set up a simple voltage divider circuit. Confirm the voltage divider equation by measuring the voltage difference across each resistor. Demo: Light Theremin
Files you will need	Not applicable.

What is Electricity?

Electricity, in the sense that we will be using it, can be described as the movement of electrons. Metals that have a portable, or *free* electron, can conduct electricity. Copper is one such metal, and consequently is commonly used to make wire. It has one “free” electron per atom. A *cloud* of these free electrons holds the metal together, forming metallic bonds. In any conductive metal, the free electrons move in Brownian-like random pathways. When there is a charge gradient, electrons generally move from an area of negative (–) charge to an area of positive (+) charge, or thinking about it differently, from a higher to lower concentration of negative charge.

If electrons flow into one end of a copper wire, the outer-most electron leaves the orbit and flows to adjacent copper atoms, causing a chain of

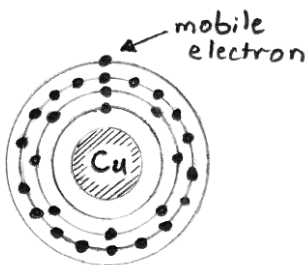


Figure 1-1. Mobile valence electron in outer shell of copper atom.