

Iguana Labs

800-297-1633 (US)

Beginner

Intermediate

Advanced

Beginner Tutorials

[Microcontroller Beginner Kit Instructions \(includes all tutorials below and intermediate tutorials\)](#)

[Basic Concepts](#)

[Basic Components \(Resistors, Diodes, LEDs, Switches\)](#)

[Finding the value of a resistor by reading it's color codes](#)

[Using a Breadboard \(Socket Board\)](#)

[Using Ohm's Law](#)

[Learning to use LEDs and Transistors \(Kit Available\)](#)

[Pulses, Oscillators, Clocks... \(Kit Available\)](#)

[Building a 5 Volt Power Supply \(Kit Available\)](#)

[Using a Multimeter - Some Simple Pointers](#)

[-5 Volt DC & -10 Volt DC Power Supply \(Kit Available\)](#)

[Home](#)

This page last updated August 1, 2002.

Iguana Labs

800-297-1633 (US)

[PG302 Microcontroller Programmer](#)

[Experimenter/Controller Board](#)

[Tutorials for Electronics](#)

[8051 Assembly Language Library](#)

[Chip Pinouts](#)

[Assemblers and other free tools](#)

[Other 8051 web sites](#)

[Prototyping Supplies](#)

[Catalog](#)

[Online Secure Order Form](#)

[Support](#)

[Information Privacy Policy](#)

[Click here to send email to support@iguanalabs.com](mailto:support@iguanalabs.com)

Iguana Labs

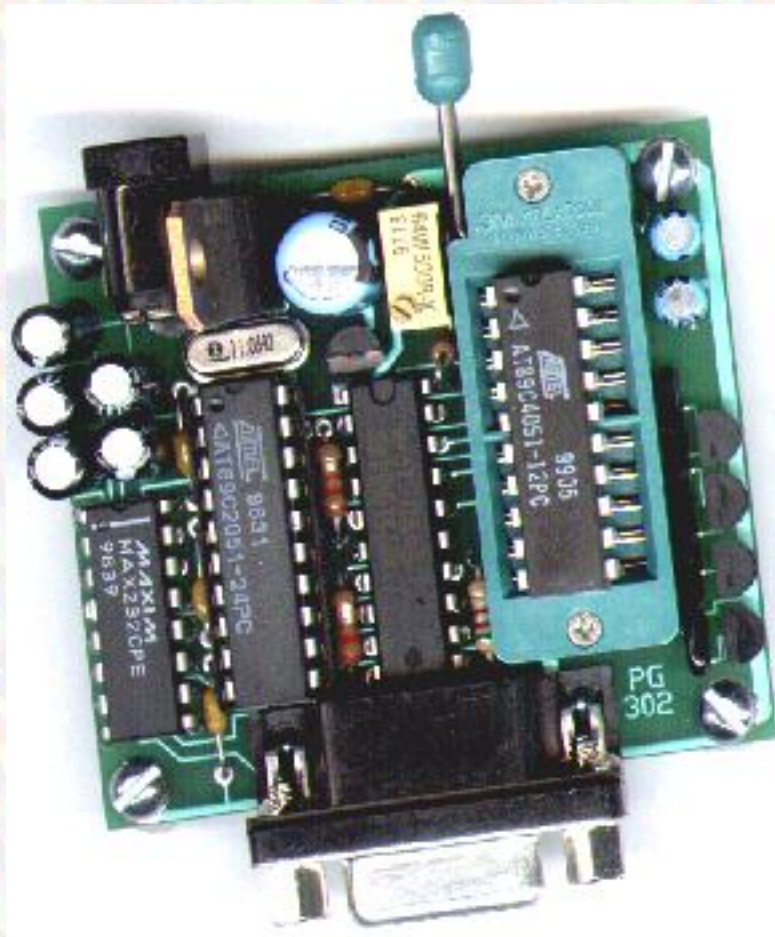
This page last updated on June 18, 2002.

Iguana Labs

800-297-1633 (US)

PG302 Device Programmer - \$69 (includes shipping)

[Features](#) - [Speeds](#) - [Device List](#) - [Software](#) - [PC Interface](#) - [Ordering Information](#)



Features:

-Programs many popular 8051 microcontrollers from Atmel, AMD, Dallas, Intel and Phillips, as well as AVR microcontrollers from Atmel.

-Comes with High Quality 3M Zero Insertion Force (ZIF) Socket. (Don't Buy a Programmer without a ZIF socket!)

-Complete control of power to socket. Power is only applied to target chip during an operation.

No need to turn off power when inserting/removing chip.

-Tested and Certified by Atmel.

-Price includes power supply (US and Canada), DB9 cable, software and USPS Priority Mail shipping (in the US). UPS and FedEx shipping are also available.

Supported Devices:

The PG302 with no adapters lets you program the following microcontrollers:

Atmel: [AT89C1051](#), [AT89C2051](#), [AT90S1200](#), [AT90S2313](#), AT89C4051

Add the [ADT87 Adapter](#) for \$21 to program the following microcontrollers:

AMD: 87C51, 87C52, 87C521, 87C541

Atmel: 89C51, 89C52, 89C55, 87F51, 87F52

Dallas: 87C520

Intel: 8751BH, 8752BH, 87C51, 87C51FA, 87C51FB, 87C52, 87C54

Phillips: 87C51, 87C51FA, 87C51FB, 87C52, 87C504, 87C524, 87C528, 87C550, 87C575, 87C576, 87C652, 87C654

Add the [ADT32K Adapter](#) for \$25 to program the following microcontrollers:

Atmel: 89C51RC, 89C55WD

Add the [ADT90 Adapter](#) for \$19 to program the following microcontrollers:

Atmel: 90S4414, [90S8515](#), 89S53, 89S8252

Other AVR chips (2323, 2333, 2343, 4433, 4434, 8535) can be programmed indirectly. [Click here for more information.](#)

The PG302 also supports in circuit programming with the [ISP302](#).

The PG302 communication protocol is provided in the document at

<http://www.iguanalabs.com/pg302pro.htm> . Custom programs can be written using this protocol for custom applications.

Speeds

Sample speeds for fully programming some microcontrollers.

Atmel AT89C2051, 11 seconds (2K file, [test.hex](#))

Atmel AT90S2313, 17 seconds (2K file, [test.hex](#))

Atmel AT89C52, 44 seconds (8K file, [89C52.hex](#))

PC Interface:

Connects to your computer's serial port (Comm 1, 2, 3 or 4) with a standard DB9 Male to DB9 Female cable. (Cable Included)

No PC Card Required.

Uses Intel Hex Data Format (Default output of most assemblers and compilers.)

The PG302 comes with everything you need:

6 Ft DB9 Serial Cable

[Windows 95, 98, Me, NT, 2000, XP Software on CD \(Free Updates From Web Site\)](#)

[CD also includes many tools and data sheets. Click here to see CD contents.](#)

Power Supply included for orders going to US and Canada

Linux Driver for 2051s (with C Source Code)

Free USPS Priority Mail Shipping! (US orders only. Allow 4 days for delivery)

The PG302 is carefully built and tested here at Iguana Labs. We are now in our 8th year of production!

Full Warranty - If you are not completely satisfied, you may return it within 30 days for a full refund. Parts and Labor are covered for three years.

[Links - Ordering Information - International Distributors - More About Iguana Labs and PG302 - Software](#)

If you would like to learn how to use microcontrollers, get our Microcontroller Beginner Kit which

includes the PG302. [Click here for more information.](#)

**Send any questions to tech@iguanalabs.com
or call us at 800-297-1633**

This page last updated on August 12, 2002.

Microcontroller Experimenter Board



(click picture for larger view)

The ExpBoard is a prototyping/experimenting board that comes completely assembled and ready to use. It includes a built in programmer for the target microcontroller which can be an 8051 based microcontroller or an AVR based microcontroller. The documentation works through some simple projects that show how to program the microcontroller and use it in some simple circuits with a speaker, a 7 segment display, input switches, the serial port, and the optional LCD Module. The web version of the instruction manual is at <http://www.iguanalabs.com/expboard>.

Features

Crystal socket allows for experimenting with different oscillator speeds.

Complete programming features for the target microcontroller as taken from our [PG302 device programmer](#).

Can be expanded with modules that plug into sockets (in place of 7 segment displays).

Works with the 8051 based AT89S8252 and the AVR based AT90S8515 microcontrollers

Several sample programs included.

2 Serial Ports - One exclusively for programming the chip and one just for the microcontroller

2 Status LEDs for each serial port. One for the transmit line and one for the receive line.

Power Indicator LED



The ExpBoard has:

2 - 7 Segment Displays

1 - 11.0592 MHz Crystal with 8252 or 8 MHz Crystal with 8515

1 - AT89S8252 Microcontroller (Advanced version of the popular 8051 microcontroller with 8K Flash Program Memory and 2K EEPROM (non-volatile RAM))

3 - Normally Open Push Button Switches

2 - Status LEDs for each Serial Port

1 - Speaker (plugs into 3.5 mm Jack)

Also Comes With:

Power Supply (US and Canada Orders Only)

6 Ft DB9 Cable

Printed Manual

CD with [Software for Windows \(Free Updates From Web Site\)](#) and ExpBoard Software

(includes compiler, assembly language editor, and more) [Click here to see list of CD contents.](#)

Free Shipping! (USPS Priority Mail (2-4 Days). US Orders Only. Faster shipping methods are also available.)

Technical Support (During business hours, call 800-297-1633 or send an email to support@iguanalabs.com)

PC Interface:

Connects to your computer's serial port (Comm 1, 2, 3 or 4) with a standard DB9 Male to DB9 Female cable. (Cable Included)

Uses Intel Hex Data Format (Default output of most assemblers and compilers.)

Expansion Modules

Currently there is one expansion module available for the ExpBoard. It is the LCD Module. The LCD Module plugs into U1 of the ExpBoard and can be mounted on the ExpBoard as shown below.



(click to enlarge)

For pricing [click here for the order information page.](#)

Send any questions to support@iguanalabs.com

This page last updated on August 1, 2002.

Iguana Labs

800-297-1633 (US)

Electronics Tutorials and Kits

Beginner

Intermediate

Advanced

[Home](#)

This page last updated June 18, 2002.

8051 Assembly Lanugage Library

Tools:

[Ay Pad: Text Editor for 8051 Assmebly Language Files](#)

[TASM: Assembler/Compiler for asm files](#)

Files:

Source File	Description	Schematic (click below)	Hex File
ledtest.asm	One Blinking LED	1stmicro.htm	ledtest.hex
ledproj2.asm	Eight Blinking LEDs	2nd2051.htm	ledproj2.hex
sounds.asm	500 Hz Tone	sound/index.htm	sounds.hex
sounds2.asm	250 Hz Tone	sound/index.htm	sounds2.hex
sound440.asm	440 Hz Tone (Musical A Note)	sound/index.htm	sound440.hex
sounds3.asm	Alternating Tone and Silence	sound/index.htm	sounds3.hex
sounds4.asm	Alternating Tones	sound/index.htm	sounds4.hex
sounds5.asm	Spaceship?	sound/index.htm	sounds5.hex
sounds6.asm	Spaceship?2	sound/index.htm	sounds6.hex
sounds7.asm	Weird	sound/index.htm	sounds7.hex
sounds8.asm	Laser Gun Fight?	sound/index.htm	sounds8.hex

7seg.asm	Numbers on 7 Segment Display	7segment.htm	7seg.hex
switch.asm	Switch as Input	switch.htm	switch.hex
light.asm	Light Sensor	light.htm	light.hex
adcproj.asm	Send Data to PC Serial Port	adc2051.htm	adcproj.hex
spexp.asm	Send Data to PC Serial Port	exp.htm	spexp.hex
spexp2.asm	Receive Data From PC Serial Port	exp.htm	spexp2.hex
lcdexp.asm	Write Data to LCD	exp.htm	lcdexp.hex

Email Addresses:

General Sales and Support: support@iguanalabs.com
Technical Questions and Support: tech@iguanalabs.com

Iguana Labs

This page last updated on August 1, 2002.

Chip Pinout Page

[1051 Pinout - Microcontroller](#)

[AT90S1200 Pinout and Description - Microcontroller](#)

[AT90S2313 Pinout and Description - Microcontroller](#)

[AT90S8515 Pinout and Description - Microcontroller](#)

[2051 Pinout and Description - Microcontroller](#)

[8051 Pinout - Microcontroller](#)

[89C51 Pinout and Description - Microcontroller](#)

[89C52 Pinout and Description - Microcontroller](#)

[STK12C68 Pinout and Description - NonVolatile 8k x 8 SRAM](#)

[7400 Pinout - Quad 2 Input NAND](#)

[7402 Pinout - Quad 2 Input NOR](#)

[7404 Pinout - Hex Inverter](#)

[7408 Pinout - Quad 2 Input AND](#)

[7414 Pinout - Hex Inverter Schmitt Trigger](#)

[7432 Pinout - Quad 2 Input OR](#)

[7447 Pinout - Seven Segment LED Driver](#)

[74138 - 1 of 8 Decoder/Demultiplexer](#)

[74193 Pinout - 4 Bit Binary Up/Down Counter](#)

[74367 Pinout - Noninverting Buffer](#)

[74393 Pinout - Dual 4 Bit Binary Counter](#)

[74573 Pinout - 8 Bit Latch \(3 State Output\)](#)

[ADC0800 Pinout - 8 Bit ADC](#)

[ADC0804 Pinout - 8 Bit ADC](#)

[LM741 - Standard OpAmp](#)

[LM7805 - 5 Volt Regulator](#)

[MAX232 - RS232 Receiver/Transmitter](#)

[Seven Segment LED Display](#)

[Suggest A New Entry](#)

Iguana Labs

This page last updated on June 20, 2002.

Iguana Labs

800-297-1633 (US)

Assemblers and other free tools

TASM Assembler 8051 / 2051 / 6502 / 6800 /
6805 / TMS32010 / TMS7000 / 8048 / Z80

Metalink Assembler

ASM Editor - Great Color syntax highlighting
for 8051 assembly language, AVR assembly
language and others!

**Acebus 8051 Integrated Editor, Simulator, and
Assembler** - Fully Functional Evaluation
Version

BascomLT Basic Compiler - Mostly Functional
Evaluation Version

8051 Basic Compiler - Allows you to write
programs in basic

8051 C Compiler - Franklin's Eval version is
fully functional (for small programs).

[8051 Simulator](#) - DOS based Simulator allows you to run hex files and see all register contents.

[8051 Disassemblers](#) - Two are included in this file

[About the 8051](#)

[8048 and 8049 Assembler](#)

[AVR Tools](#) - Assembler and Simulator for Atmel's AVR Microcontrollers

[AVRASM](#) - Command Line Assembler for AVR_s

[WinBoard PCB Schematic Software](#) - Free version is limited to boards with less than 100 pins.

[PKZIP Tools](#) - Popular tools for zipping and unzipping

[Chip Pinouts](#) - Easy to use (no big PDFs)

[Free Tutorials for Electronics](#)

[Click here to send email to
support@iguanalabs.com](mailto:support@iguanalabs.com)

Iguana Labs

This page last updated on June 20, 2002.

Iguana Labs

800-297-1633 (US)

Other 8051 web sites

[Paul's Page](#) - Lots of resources for 8051s/2051s

[Bascom LT](#) - Low Cost Basic Compiler for 8051 family

[8052.com](#) - Great source for 8051 based tutorials and code, etc.

[8051 Projects](#)



- Lots of Atmel and PIC products

[Robot Books, Kits, Etc.](#)

[Advanced Microcomputer Systems](#) - Tools and Kits for Microcontrollers

[Click here to send email to support@iguanalabs.com](mailto:support@iguanalabs.com)

Iguana Labs

Home Page

This page last updated on March 12, 2002.

Iguana Labs

800-297-1633 (US)

Prototyping Supplies

Alberta - 1 day 'Quick Turn' circuit boards

Express PCB - Easy PCB Production

Mental Automation - CAD schematic tools

NOHAU - 8051 Emulator

CEIBO - 8051 Emulator

Iguana Labs

Home Page

[Click here to send email to support@iguanalabs.com](mailto:support@iguanalabs.com)

This page last updated on June 27, 2002.

Iguana Labs

800-297-1633 (US)

Ordering Information and Price List



[Click here for the Online Secure Order Form](#)

To order by phone:

Monday - Friday, 7:00 AM to 4:00 PM Mountain Time

(800) 297-1633

(303) 823-5557

Current Availability Status

No Minimum Order!

Shipping is free for United States (First Class Mail, Allow 7-10 days for delivery). USPS Priority Mail is free (in the US) for orders over \$100 or over 1 pound (PG302 or Microcontroller Beginner Kit for example).

FedEx and UPS are also available.

Orders received before 11:00 AM Mountain Time (1:00 PM Eastern and 10:00 AM Pacific) are shipped same day.

To order, use the [secure order form](#) or call us at 1-800-297-1633 or (303) 823-5557.

[Click here to view our Information Privacy Policy](#)

To contact us:

Email: support@iguanalabs.com

Phone: 800-297-1633 or (303) 823-5557

Mail: Iguana Labs Inc

47 Iroquois Ct
Lyons CO 80540

Device Programmer

PG302, \$69.00

Adapters For PG302

ADT87, \$21.00

ADT32K, \$25.00

ADT90, \$19.00

ISP302, \$8.00

Kits

Beginner's Kit, \$24.00, - Learning to use Transistors and LEDs + Pulses, Oscillators, Clocks...

Microcontroller Beginner Kit, \$89.00 - Beginner's Kit + PG302 and More

ExpBoard, \$79.00 - Experimenter Board for 8051s and 8515s (assembled and tested)

LCD Module, \$25.00 - LCD Display for ExpBoard (assembled and tested)

555Kit, \$3.00, - Pulses, Oscillators, Clocks...

5voltKit, \$1.50, - 5 Volt Regulator Kit

-5&10Kit, \$4.00, - Negative 5 Volts DC and Negative 10 Volts DC Kit

2051Kit, \$7.50, - Learning to use a 2051 Microcontroller

8951Kit, \$7.50, - Learning to use a 8951 Microcontroller

8952Kit, \$8.00

8515Kit, \$8.50 - Basic parts needed for an 8515 system

MAX232Kit, \$5.00 - Basic parts needed for communicating to a PC

ADC2051Kit, \$15.00 - Data Collection (Analog to Digital Conversion and sending data to a PC)

LightKit, \$15.00 - Light Sensor

Printed Copy of Kit Instructions, \$1.00 each

Microcontrollers

AT89C2051-24PC, \$4.50

AT89C4051-24PC, \$5.00

AT89C51-24PC, \$4.50

AT89C52-24PC, \$5.00

AT89S8252-24PC, \$7.00

AT90S1200-12PC, \$2.00

AT90S2313-10PC, \$5.00

AT90S8515-8PC, \$7.00

ATmega161-8PC, \$12.00

24LC256-I/P, \$3.00

Other Components

Mating Header for ISP302, \$ 0.50

7 Segment Display, \$1.50

8 pin Socket, \$0.50

16 pin Socket, \$0.50

20 pin Socket, \$0.50

40 pin Socket, \$0.50

20 pin ZIF Socket (Made by 3M), \$11.00

40 pin ZIF Socket (Made by ARIES), \$11.00

6.0000 MHz Crystal (HC-49/US), \$1.50

8.0000 MHz Crystal (HC-49/U), \$1.50

10.0000 MHz Crystal (HC-49/US), \$1.50

11.0592 MHz Crystal (HC-49/US), \$1.50

24.0000 MHz Crystal (HC-49/US), \$1.50

MAX232, \$2.50

ADC0804, \$3.00

Accessories

6 Ft DB9 Male to DB9 Female Serial Cable, \$6.00

IC Extractor (8 to 40 pins), \$2.00

Small Breadboard, \$9.00

Large Breadboard, \$25.00

Iguana Labs Software CD, \$3.00

Printed Tutorials

Our tutorials are available as full color printouts for \$ 1.00 each.

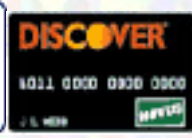
[Click here for the Online Secure Order Form](#)

Iguana Labs Inc
47 Iroquois Ct
Lyons CO 80540

Iguana Labs

Iguana Labs

Secure Order Form



Current Availability Status

Complete all of the necessary information below and then click the Submit button at the bottom of the page to submit your order to us. Orders received before 11:00 AM Mountain Time (1:00 PM Eastern and 10:00 AM Pacific) are shipped same day. If you have any problems with this order form please call us at 1-800-297-1633 or send an email to support@iguanalabs.com.

[Click here for our Information Privacy Policy](#)

Credit Card Information

Select Card Type

Card Number

Expiration Date: Month Year

Name on Card

Credit Card Billing Address (If different than Shipping Address)

Shipping Information

Name

Company

Address Line 1

Address Line 2

City

State/Province

Zipcode/Postal Code

Country

Contact Information

Email Address

Phone Number

Select the items you want to order

Item 1

Quantity

Item 2

Quantity

Item 3

Quantity

Item 4

Quantity

Item 5

Quantity

Item 6

Quantity

Item 7

Quantity

Item 8

Quantity

Item 9

Quantity

Item 10

Quantity

Shipping for United States Orders

Orders received before 11:00 AM Mountain Time (1:00 PM Eastern and 10:00 AM Pacific) are shipped same day.

Delivery times are from the day the order is shipped.

First Class USPS Shipping is free for United States (Allow 7 to 10 days for delivery). USPS Priority Mail is free for orders over \$100 or over 1 pound (PG302 or Microcontroller Beginner Kit for example). Other options are also available. For these, shipping charges are based on the weight of the order. A typical range for each service is shown below. If you have a FedEx account that you would like to use, enter the number in the shipping comments section below.

Shipping Comments for US Orders

Shipping for International Orders

Allow one business day for international orders to be shipped.

Delivery times are from the day the order is shipped.

A range of prices is given for each service listed in the options below. The exact price is based on the weight of the order. If you have a FedEx account that you would like to use, enter the number in the shipping comments section below.

Shipping Comments for International Orders

Order Confirmation

Print out a copy of this page for your records before submitting it. You will receive an email confirmation within one business day.

Review all of the order information entered above and then press the submit button below to complete the order. This should make another screen appear saying "Thank you for your order" to let you know the order was received. If you get a screen that says "Page Can Not Be Found", go back and press submit again or call us at 800-297-1633.

Iguana Labs

Troubleshooting Guide and FAQ for PG302 device programmer and adapters.

Testing the PG302.

Here is a sample hex file that can be used for 2051s or the various versions of 8051s. [ledtest.hex](#)

The tutorial at <http://www.iguanalabs.com/1st2051.htm> gives the basic steps of how to program a chip.

Error Messages

'Comm Port Not Available'

- 1.) Do you get the same message if the PG302 is not plugged into the PC? If so, the problem is probably in the PC system set up or PC hardware.
- 2.) If this occurs before any programming occurs, it may mean another program or device (modem?) may be using the comm port.
- 3.) If this occurs during programming, it probably indicates a problem with the hex file. Try the test file above. If the file above works but you can't get your hex file to work, try sending it to us and we will check to see if it is a valid hex file.

'Programmer Not Responding'

- 1.) If the programmer was working but seems to have died, try unplugging the power and plugging it back in. The on board firmware may have gotten confused due to a bad hex file, power glitch, etc...
- 2.) No power to unit. Check if there is a DC voltage coming from the power supply. (The voltage should be around 15 to 16 Volts) Check if there is 5 Volts on pin 20 of the AT89C2051 (with respect to ground on pin 10).
- 3.) Hex File is empty. If the Hex file is blank, the software may give this error.
- 4.) Comm port not set up correctly

Check the settings under Control Panel, Ports, ...
Try setting it to 9600 baud, No Parity, 8 bits, 1 stop bit.

- 5.) Try a terminal program

In the terminal program, set up the Comm port for 9600, N, 8, 1. Type a '1' in the window (to send a 1 to the programmer). The programmer should respond with a Y.

- 6.) To check if the adapter may be causing a problem, you can try running the unit with out an adapter or chip in the 20 pin ZIF socket. If the programmer is working correctly, it should go through the programming sequence (0 to 100% in the progress bar) and then say 'Device Not Equal to File'.

'Device Not Equal to File'

- 1.) Make sure the correct chip type is selected. The current chip type is shown in the upper right corner of the PC software.
- 2.) The file may be too large. Try a small test program like the one at the top of the page.
- 3.) The chip you are trying to program may be a bad chip (unlikely but possible). Try another chip if you have one.
- 4.) The chip may be backwards. Pin 1 should go next to the handle. When using an adapter, Pin 1 should go next to the handle of the adapter.

Adapters

Which adapter is which?

The ADT87 is the adapter with two 14 pin chips on the bottom (usually 74HC573 or 74HCT573).

The ADT90 is almost bare on the bottom except for two small brown capacitors and a crystal.

Frequently Asked Questions

- 1.) Can the PG302 read chips that have already been programmed?

Yes as long as the lock bits have not been set. If the chip is from a commercial product, the lock bits have probably

been set.

2.) How do I read a chip that has already been programmed?

You should be able to read a chip by using the read option in the PG302 software. Press Read and then type in the name of a new file and then press Save.

If you are trying to read a chip that was programmed by someone else, it may have the lock bits set (especially chips from commercial products). If the lock bits are set, the checksum will be the same as it is for a blank chip. We don't know of any way to read a chip that has had the lock bits set.

3.) I can program the Flash on an AVR chip but I can't seem to program the EEPROM.

The EEPROM is much smaller. Make sure the file is not too big. (If the file is too big it probably just says 'Device Not Equal to File')

4.) Can the PG302 program Atmel's 29Cxxx chips?

No.

5.) How do I erase the lock bits on an AT90S2313-10PC?

Due to a manufacturing defect, some 2313 chips will not let you erase the lock bits once the lock bits have been set. However it is possible to erase the lock bits if Vcc is lowered to less than 4 volts. To accomplish this with the PG302,

Put the 2313 on a breadboard,

Connect jumper wires from the 20 pin ZIF socket to the 2313 on the breadboard.

International

What are the power supply specifications?

The power supply should provide at least 15VDC (but not more than 24VDC). Most power supplies put out more than the voltage rating on the box. We have seen power supplies that are rated at 24VDC but actually put out 33VDC even when running the PG302. If the power supply is putting out more than 24VDC, the power regulator (the 7805) will probably overheat and may fail. An unregulated power supply of 12VDC usually works.

The power supply should be able to provide at least 200mA of current.

Plug size should be 2.1 mm x 5.5mm. (where 2.1 is the diameter and 5.5 is the length.)

Center positive, Outside negative

For further technical assistance send email to support@iguanalabs.com

Iguana Labs

Iguana Labs Inc Information Privacy Policy

We do not use cookies of any kind. The only personal information collected is the information you submit to us on the order form when you place an order. We collect data about how our web pages are used (for example, which pages and downloads are most popular) but this information is no way linked to specific users. There is no way for us to tell what any user has viewed or downloaded.

We have never and will never sell your email address, phone number or other information to anyone.

We have never and will never share your email address, phone number or other information with anyone (except what is required for shipping your order) unless you request it for some type of support.

We will not use any information collected (email addresses, phone numbers, mailing addresses) to contact you except for the following:

- A copy of your order is sent to you by email for verification and for your records
- Tracking information (if available) is sent to your email address
- If there is a problem with your order we will try to contact you by email, or possibly by phone if it seems urgent.

We will not call you to try to sell you more stuff. If you want to talk to us while you are eating dinner, you will have to call us :)

We will not send you any newsletters, advertisements, or other junk.

The information collected for orders will not be used for any other purposes than to complete the sale.

All information collected for each sale is securely encrypted when you send it to us.

No customer information is stored in any type of online database. (This means you have to reenter all of your data each time you order, but the information is not there for someone else to get.)

If you have any questions, call or write to us.

Contact Information:

Iguana Labs Inc
47 Iroquois Ct
Lyons CO 80540

800-297-1633
(303)-823-5557
support@iguanalabs.com

[Click here to send email to support@iguanalabs.com](mailto:support@iguanalabs.com)

Iguana Labs

This page last updated on January 17, 2002.

Iguana Labs

800-297-1633 (US)

Beginner

Intermediate

Advanced

Intermediate Tutorials

Truth Tables

First Microcontroller Project - 8951 (Kit Available)

Microcontroller Beginner Kit Instructions (includes most beginner tutorials and the tutorials listed below)

First Microcontroller Project - 2051 (Kit Available)

Second MC Project - 2051 as 8 bit Counter (Kit Available)

Making Sound With The 2051 Microcontroller

Using a 7 Segment LED Display with a 2051

Using Switches as Inputs

This page last updated August 1, 2002.

Iguana Labs

800-297-1633 (US)

Beginner

Intermediate

Advanced

Advanced Tutorials

[Data Collection - Analog to Digital Conversion and Communicating with a PC through the Serial Port \(Kit Available\)](#)

[Light Sensor Kit](#)

[LCD Instruction Set](#)

[Home](#)

This page last updated August 1, 2002.

Truth Tables

Introduction

Calculators look at a list of inputs and produce an output. Computers do the same thing. We use truth tables to show this list of inputs and outputs. Inputs are limited to two possibilities, either 0 or 1. This is the meaning of digital. The name digital (or binary) means two values. A 0 is called False and a 1 is called True.

Theory

The simplest truth table is for one input and one output. In this case there are two possible truth tables. These are shown below.

Input	Output
0	0
1	1

Figure A

Input	Output
0	1
1	0

Figure B

For Figure A it is the same as having a wire connecting the input and the output. Whatever appears on the input is transferred to the output.

For Figure B, the opposite happens. This is called a NOT operation (NOT gate). The output is not the input. The output is True (1) if the input is not True (1). The output is False (0) if the input is not False (0). To learn how to build a NOT gate, go to [LED and Transistor Kit](#).

Another way to look at a truth table is as a list of possible events and what will happen in each case. Suppose your friend John might come to visit. If he comes you will not watch Matlock. If he does not come you will watch Matlock. Then the table would look like the following.

Event	Result
-------	--------

John Comes	You Don't Watch Matlock
John Doesn't Come	You Watch Matlock

NOT GATE

We will build a device with one input and one output. We will give the device an input, either True (1) or False (0). The device will give us an output (either True or False) to tell us what happened as a result of our input. Later we will build devices with more inputs and outputs so it can do more complex things for us.

When we build this first device we will use 0 volts for False (0) and we will use 5 volts for True (1). If the input to our device is 0 volts then the output will be 5 volts. If the input is 5 volts then the output will be 0 volts. This will be our first logic gate, the not gate. It is also called an inverter. Logic gates are devices that are built to do truth tables. Computer chips like the Pentium are made of logic gates.

AND GATE

Now lets look at a device with two inputs and one output. First, we will look at a device that does an AND operation. Our inputs will be A and B. Our output will be C.

If A and B are both True (1) then the output, C, will be True (1). Otherwise C will be False (0). This operation is shown in the truth table given below.

Input A	Input B	Output	Input A	Input B	Output
False	False	False	0	0	0
False	True	False	0	1	0
True	False	False	1	0	0
True	True	True	1	1	1

The table gives a list of all possible combinations of inputs and the resulting output for each combination. This is not the only possible combination of outputs, but this particular combination is called an AND gate.

OR GATE

Another possibility for a device with two inputs and one output is an OR operation. For inputs A and B and output C : If A or B is True (1) then C is True (1). Otherwise C is False (0).

To build the truth table, we first look at the inputs. A can be True or False. B can be True or False. First we make a table of all possible combinations of A and B.

A	B
False	False

False	True
True	False
True	True

Now we can determine the output by saying for each combination, if A or B is true then the output, C, is True. Then we have the following table.

A	B	C
False	False	False
False	True	True
True	False	True
True	True	True

Then to convert this to a form we can build (using 5 Volts for True and 0 Volts for False) we make the following table.

A	B	C	A	B	C
False (0V)	False (0V)	False (0V)	0V	0V	0V
False (0V)	True (5V)	True (5V)	0V	5V	5V
True (5V)	False (0V)	True (5V)	5V	0V	5V
True (5V)	True (5V)	True (5V)	5V	5V	5V

Boolean Algebra

These three gates (the NOT gate, the OR gate, and the AND gate) are the basic building blocks of digital design. They are all that is needed to build the most complex computers that exist. To build complex designs, a type of math has been developed to deal with binary numbers. It is called Boolean Algebra. It gives you a way to combine these three gates into bigger designs.

The three basic operations have symbols. A NOT operation is represented by a line over a letter. Instead of using this line we will just say 'bar'. For example \bar{A} means 'not A'. So if \bar{A} is 1 then A is 0. If \bar{A} is 0 then A is 1.

An OR operation is represented by a + sign. For example, $A + B = A \text{ OR } B$.

An AND operation is represented by a *. For example, $A * B$ means A AND B.

Examples

1.) If $A = 1$ (5V) and $B = 0$ (0V) then what is $A * B$?

Since 1 is a True and 0 is a False then we can say the problem is $\sim \text{True} * \text{False} = ?$. If we put in AND for * then the problem

is $\sim \text{True AND False} = ?$. To answer this we can look back at the truth table for the AND operation and see that True AND False is False. Another way to find the answer is to look at the definition of the AND operation. It says if A and B are True then the output is True. Otherwise the output is false. In the above example since B is False the output is False so $A * B = \text{False}$ ($A * B = 0$).

2.) If $A = 1$ (5V) and $B = 0$ (0V) then what is $A + B$?

To do this problem we can say that $A = 1 = \text{True}$ and $B = 0 = \text{False}$. Then the problem is $\sim \text{True OR False} = ?$ We can either look at the truth table for the OR operation or we can look at the definition of the OR operation which says. If A or B is True then the output is True. Otherwise the output is False. Since A is True in our question then the output is True. So $\text{True} + \text{False} = \text{True}$ ($1 + 0 = 1$).

Try doing the following problems using the truth tables and the definitions above. The answers are given below.

1.) $A = 1$ (5V) and $B = 0$ (0V). What are the answers for the following problems?

- a.) $A + B = ?$
- b.) $A * B = ?$
- c.) $\overline{A} = ?$
- d.) $\overline{B} = ?$

2.) $A = 0$ (0V) and $B = 0$ (0V). What are the answers to the following problems?

- a.) $A + B = ?$
- b.) $A * B = ?$
- c.) $\overline{A} = ?$
- d.) $\overline{B} = ?$

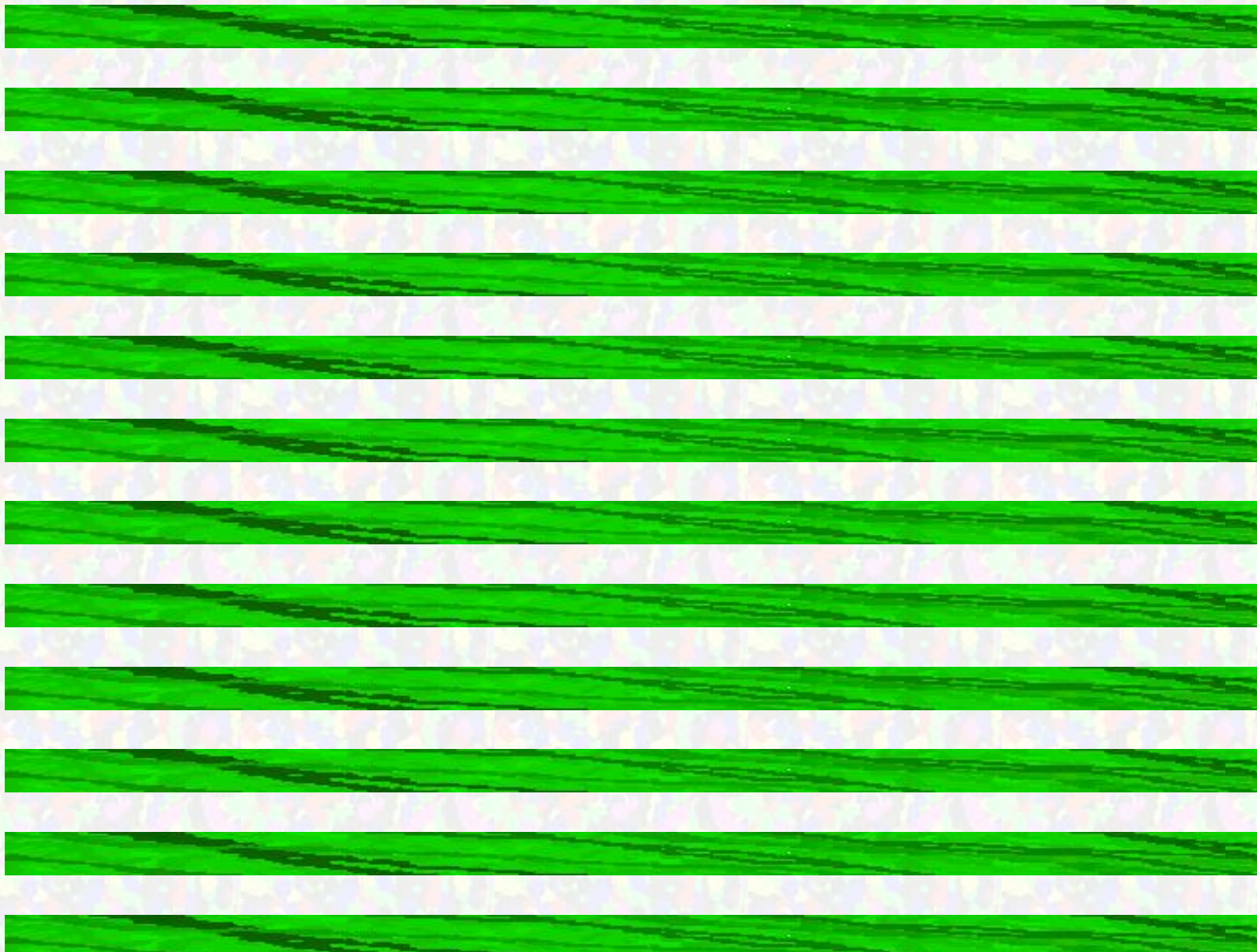
3.) $A = 1$ and $B = 1$. What are the answers to the following problems?

- a.) $A + B = ?$
- b.) $A * B = ?$
- c.) $\overline{A} = ?$
- d.) $\overline{B} = ?$

4.) $A = 0$ and $B = 1$. What are the answers to the following problems?

- a.) $A + B = ?$
- b.) $A * B = ?$
- c.) $\overline{A} = ?$
- d.) $\overline{B} = ?$





Answers:

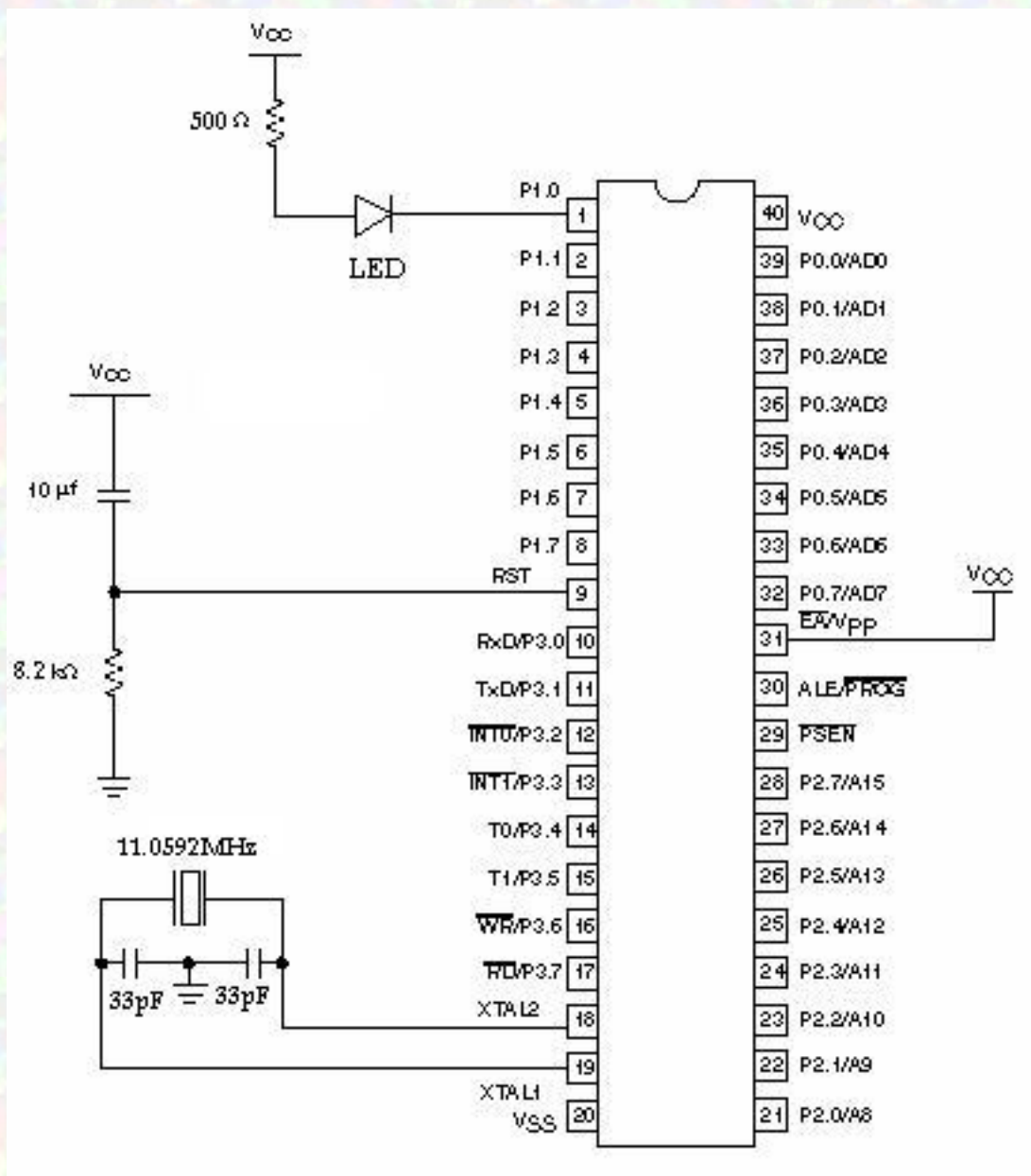
- 1. a.) 1 (5V) b.) 0 (0V) c.) 0 (0V) d.) 1 (5V)
- 2. a.) 0 (0V) b.) 0 (0V) c.) 1 (5V) d.) 1 (5V)
- 3. a.) 1 (5V) b.) 1 (5V) c.) 0 (0V) d.) 0 (0V)
- 4. a.) 1 (5V) b.) 0 (0V) c.) 1 (5V) d.) 0 (0V)

Iguana Labs

Making an LED Blink - 8951

Step 1.) The first step is to build the circuit. At this point you should be familiar with the parts used. (2 resistors, 3 capacitors, 1 LED). You can either put these parts together using a breadboard or wirewrap. This design is intended for use with an Atmel 89C51. Most microcontrollers (such as a normal 8051 or 8751) can not handle the current required to turn an LED on and off but the ATMEL part has this capability.

$V_{cc} = 5V$ and $Gnd = 0V$



The only thing we want to do with this project is to make the LED blink. By doing this, you will be able to learn the basic process of compiling a program written in assembly language and then programming the resulting file into the microcontroller.

First we will assume we already have the assembly code written. [ledtest.asm](#) is the assembly language program we are going to use. (This file is included with [TASM](#))

Step 2.) Compiling the Code

Move the assembly language program ([ledtest.asm](#)) to the directory where you have [TASM](#). Bring up a DOS prompt, change to the directory where the TASM files are, and compile the code using the command

```
tasm -51 ledtest.asm ledtest.hex
```

This will create a file called ledtest.hex.

Close the DOS prompt window now.

Step 3.) Downloading the code to the Microcontroller.

Make sure the serial cable and the power supply are connected to the [PG302 programmer](#).

Put your microcontroller into the PG302 programmer.

Run [PG302](#).

From the Setup Menu, select the type of device (microcontroller) you are using

From the Setup Menu, select the Comm port you are using.

Press PROGRAM DEVICE.

Press BROWSE.

Find ledtest.hex and click on it (single click).

Press OK to select the file.

Press OK to program the file into the microcontroller.

Now the program should be loaded into the microcontroller.

Make sure the power is off for the circuit you have built.

Move the microcontroller back to the circuit you have built.

Turn on the power to the circuit. If the LED starts blinking, then you have successfully built your first microcontroller project. For more information on using microcontrollers, look at the [Microcontroller Beginner Kit \(click here\)](#).

The parts for this project are available. See [8951Kit.htm](#) for the parts list and look for **8951Kit** on [Order Form](#) for pricing information or call 800-297-1633.

You may also be interested in the ExpBoard. It is a fully assembled board designed for experimenting with microcontrollers. [Click here for more information.](#)

Send Questions to support@iguanalabs.com

Iguana Labs

This page last updated on May 1, 2002.

Microcontroller Beginner Kit

[To see the sales ad for the Microcontroller Beginner Kit, click here.](#)
[For price and ordering information, click here or call 800-297-1633.](#)

Table of Contents

[Microcontroller Beginner Kit, Parts List](#)

[Overview of Instruction Book](#)

[Chapter 1: Basic Electronics](#)

[1.1 Basic Definitions and Concepts](#)

[1.2 Basic Components](#)

[1.3 Finding the Value of a Resistor by Color Codes](#)

[1.3.1 Resistor Color Codes](#)

[1.3.2 Resistor Rules](#)

[1.4 Finding Voltage and Current Using Ohm's Law](#)

[1.5 Using a Bread Board](#)

[1.6 Transistors and LEDs](#)

[1.6.1 The Transistor](#)

1.6.2 Introduction to Digital Devices - The Inverter

1.7 Oscillators, Pulse Generators, Clocks... Capacitors and the 555 Timer IC

1.7.1 The Capacitor

1.7.2 The 555 Timer

Chapter 2: Microcontrollers

2.0.1 PG302 Setup

2.0.2 AY Pad Software Setup

2.0.3 TASM Software Setup

2.0.4 Chapter 2 Summary

2.1 Building a 5 Volt Power Supply

2.2 The 2051 Microcontroller

2.3 Making an LED Blink

2.3.1 Partial Instruction Set for 2051 Microcontroller

2.4 A Simple Microcontroller System

2.5 Making Sound with a Speaker

2.6 Using a 7 Segment Display

2.7 Using a Switch as an Input to the Microcontroller

Microcontroller Beginner Kit, Parts List

1 PG302, including DB9 Cable, Software (on CD), and Power Supply.

1 Breadboard

1 IC Extractor (4 inch metal gadget)

1 Power Supply Adapter (Use with the PG302 power supply to run the projects on the breadboard.)

Resistors: (light tan with colored stripes) (Section 1.3 shows how to find the resistor value from the stripes)

10 - 100 ohm resistors

10 - 330 ohm resistors

5 - 510 ohm resistors

5 - 2,200 ohm resistors

5 - 8,200 ohm resistor

5 - 10,000 ohm resistors

5 - 15,000 ohm resistor

5 - 100,000 ohm resistors

Capacitors

2 - 33pf capacitors (small brown flat disks)

2 - 10uf capacitor (small can shaped component)

2 - 220uf capacitor (larger can shaped component)

10 - LEDs (small green components)

5 - NPN Transistors (small black components with 3 legs)

1 - 11.0592 MegaHertz Crystal (metal, silver colored component)

1 - LM7805, 5 Volt Regulator (black 3 leg component)

2 - 555, Timers (black 8 leg components)

2 - AT89C2051, Microcontrollers (black 20 leg component)

1 - Speaker (with wires for breadboard)

1 - 7 Segment Display

2 - Normally Open Push Button Switches

Jumper Wires for Breadboard

The CD has all the software needed for [PG302 Device Programmer](#) and the microcontroller projects. [Click here to see list of CD contents.](#)

Overview of Instruction Book

The Microcontroller Beginner Kit includes all the parts necessary to do most of the projects offered by Iguana Labs. If you have no experience with electronics, start by reading the tutorials listed below. If you are already familiar with basic electronics you can skip to Chapter 2.

1.1 Basic Concepts

1.2 Basic Components

1.3 Finding the Value of a Resistor

1.4 Using Ohm's Law

1.5 Using a Breadboard

After reading Sections 1.1 through 1.5 you can start playing with the components by working through the first project in Section 1.6, *LEDs and Transistors*. Next work through section 1.7, the *Pulses, Oscillators, Clocks...* project.

Chapter 2 starts with an introduction to microcontrollers and then explains how to set up the software we will use. This chapter assumes that you are familiar with computers (creating folders and files, using the CD drive, copying and pasting files, etc.) Section 2.1 builds the 5 volt power supply. You will use the 5 volt power supply for the microcontroller projects. Section 2.3 is a simple project that shows the process we go through to program a microcontroller. Section 2.4 goes through a slightly more complex project. Section 2.5 shows how to make sound with the microcontroller using a speaker. Section 2.6 shows how to use a 7 segment display to display numbers. Section 2.7 introduces inputs to the microcontroller using switches.

Chapter 1: Basic Electronics

1.1 Basic Definitions and Concepts

Introduction

Welcome to the exciting world of electronics. Before we can build anything we need to look at a couple of concepts. Anytime you have an electrical circuit, you have voltage and current. We build circuits to control voltage and current.

Current

Current is what flows through a wire. Think of it as water flowing in a river. The current flows through wires from one point to another point just like water in a river. Current flows from points of high voltage to points of low voltage. Current can be shown in circuit diagrams by using arrows as in Figure 1. The arrow shows which way the current is flowing. An I is included beside the arrow to indicate current.

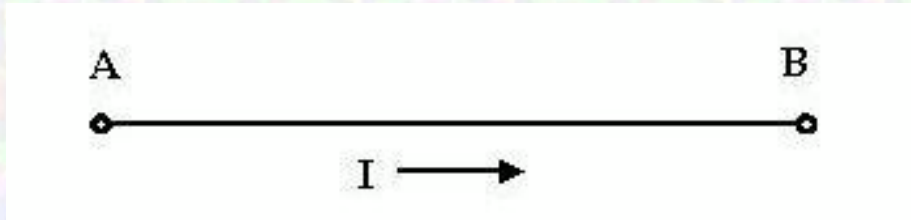


Figure 1

The unit of measurement for current is the Ampere, or Amp for short, and abbreviated as A. Common currents are 0.001 Amps (0.001A) to 0.5 Amps (0.5A). Since currents are usually small, they are often given in the form of milliAmps (abbreviated mA.) The milli means divided by 1000, so 0.001 Amps equals 1 milliAmp (1 mA) since $1 / 1000 = 0.001$. Also, 0.5 Amps equals 500 milliAmps (500mA) since $500 / 1000 = 0.5$.

Voltage

Voltage is a measure of how much electricity is at a point. If we continue the river comparison, a point at the top of a hill would be at a high voltage level and a point at the bottom of a hill would be at a low voltage level. Then, just as water flows from a high point to a low point, current flows from a point of high voltage to a point of low voltage. If one point is at 5 volts and another point is at 0 volts then when a wire is connected between them, current will flow from the point at 5 volts to the point at 0 volts. (Voltage is measured in volts.)

A measurement of voltage is much like a measurement of height. It gives you the difference in voltage between those two points. If point A is at 10 volts and point B is at 2 volts then the voltage measured between A and B is 8 volts ($10 - 2$). This is

similar to measuring height. We measure the height of hills the same way. We say the sea level is at zero feet and then compare other points to that level. On top of Mary's Peak you are 4000 ft high (compared to sea level). In the same way we call the lowest voltage in a circuit zero volts and give it the name ground. Then all other points in the circuit are compared to that ground point. Rivers always flow towards sea level and currents always flow towards ground.

A battery is similar to a dam. On one side is a lot of stored up energy. When a path is formed from that side to the other side then current flows. If there is no path then current does not flow and the energy just stays there waiting for a path to form to the other side. The path can be a big path with lots of current flowing or a small path with just a little bit of current flowing. With a dam, a little bit of water flow could go on for a long time, but flow through a big path that lets all the water go at once would only last a short while. A battery is the same. If there is a big path from the high voltage side to the low voltage side then the battery will not last long.

Open Circuit

An open circuit is when two points are not connected by anything. No current flows and nothing happens. If a wire in your vacuum cleaner breaks it can cause an open circuit and no current can flow so it does not do anything. There may be a voltage between those two points but the current can not flow without a connection.

Short Circuit

A short circuit (or short) is when two points with different voltage levels are connected with no resistance (see resistors) between two points. This can cause a large amount of current to flow. If a short circuit happens in your house, it will usually cause a circuit breaker to break or a fuse to blow. If there is nothing to limit the current, the wires may melt and cause a fire. This situation is something like a dam breaking. There is a large amount of energy suddenly free to flow from a high point to a low point with nothing to limit the current.

There are two basic ways that components can be connected. One is in **series** and the other is in **parallel**. We will refer to these types of connections later when we are building circuits.

Series Connection

A series connection is when two components are joined together by a common leg and nothing else is connected to that point as shown in Figure 2.

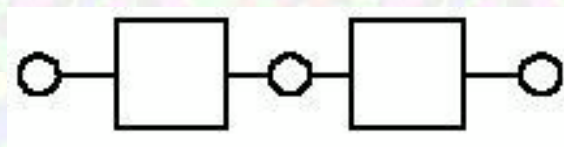


Figure 2

Parallel Connection

A parallel connection is when two components are joined together by both legs as shown below.

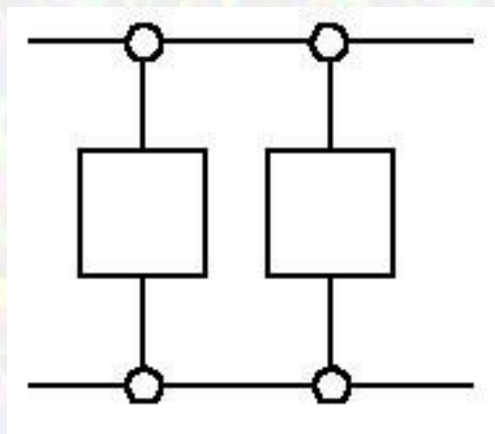


Figure 3

1.2 Basic Components

Resistors

Resistors are components that have a predetermined resistance. Resistance determines how much current will flow through a component. Resistors are used to control voltages and currents. A very high resistance allows very little current to flow. Air has very high resistance. Current almost never flows through air. (Sparks and lightning are brief displays of current flowing through air. The light is created as the current burns parts of the air.) A low resistance allows a large amount of current to flow. Metals have very low resistance. That is why wires are made of metal. They allow current to flow from one point to another point without any resistance. Wires are usually covered with rubber or plastic. This keeps the wires from coming in contact with other wires and creating short circuits. High voltage power lines are covered with thick layers of plastic to make them safe, but they become very dangerous when the line breaks and the wire is exposed and is no longer separated from other things by insulation.

Resistance is given in units of ohms. Common resistor values are from 100 ohms to 100,000 ohms. The letter k is often used with resistors to mean '1000'. For example, a 10,000 ohm resistor is written as 10k ohms. Each resistor is marked with colored stripes to indicate its resistance. To learn how to calculate the value of a resistor by looking at the stripes on the resistor, go to the resistor values tutorial, Section 1.3, which includes more information about resistors.

Variable Resistors

Variable resistors are also common components. They have a dial or a knob that allows you to change the resistance. This is very useful for many situations. Volume controls are variable resistors. When you change the volume you are changing the resistance which changes the current. Making the resistance higher will let less current flow so the volume goes down. Making the resistance lower will let more current flow so the volume goes up. The value of a variable resistor is given as its highest resistance value. For example, a 500 ohm variable resistor can have a resistance of anywhere between 0 ohms and 500 ohms. A variable resistor may also be called a potentiometer (pot for short). No variable resistors are included in this kit.

Diodes

Diodes are components that allow current to flow in only one direction. They have a positive side (leg) and a negative side. When the voltage on the positive leg is higher than on the negative leg then current flows through the diode (the resistance is very low). When the voltage is lower on the positive leg than on the negative leg then the current does not flow (the resistance

is very high). The positive leg of a diode is the one with the line closest to it.

Usually when current is flowing through a diode, the voltage on the positive leg is 0.65 volts higher than on the negative leg. There are no standard diodes in this kit but a special kind of diode called an LED is included.

LEDs

Light Emitting Diodes are great for projects because they provide visual entertainment. LEDs use a special material which emits light when current flows through it. Unlike light bulbs, LEDs never burn out unless their current limit is passed. A current of 0.002 Amps (2 mA) to 0.02 Amps (20 mA) is a good range for LEDs. They have a positive leg and a negative leg just like regular diodes. To find the positive side of an LED, look for a line in the metal inside the LED. It may be difficult to see the line. This line is closest to the positive side of the LED. Another way of finding the negative side is to find a flat spot on the edge of the LED. This flat spot is on the negative side. Also, the positive leg is usually longer than the negative leg (this is true for the LEDs in this kit).

When current is flowing through an LED the voltage on the positive leg is about 1.4 volts higher than the voltage on the negative side. Remember that there is no resistance to limit the current so a resistor must be used in series (see Series Connection in Section 1.1) with the LED to avoid destroying it.

Switches

Switches are devices that let you control whether two points in a circuit are connected or not, depending on the position of the switch. For a light switch, ON means connected (current flows through the switch, lights light up and people dance.) There is practically no resistance through the switch when it is turned on. When the switch is OFF, that means there is an open circuit (no current flows, lights go out and people settle down. This effect on people is used by some teachers to gain control of loud classes.)

When the switch is ON it looks and acts like a wire. When the switch is OFF there is no connection.

1.3 Finding the Value of a Resistor by Color Codes

To calculate the value of a resistor using the color coded stripes on the resistor, use the following procedure.

Step One:

Turn the resistor so that the gold or silver stripe is at the right end of the resistor.

Step Two:

Look at the color of the first two stripes on the left end. These correspond to the first two digits of the resistor value. Use the table below to determine the first two digits.






Step Three:

Look at the third stripe from the left. This corresponds to a multiplication value. Find the value using the table below.

Step Four:

Multiply the two digit number from step two by the number from step three. This is the value of the resistor in ohms. The fourth stripe indicates the accuracy of the resistor. A gold stripe means the value of the resistor may vary by 5% from the value given by the stripes.

1.3.1 Resistor Color Codes (with gold or silver strip on right end)

Color	Color	First Stripe	Second Stripe	Third Stripe	Fourth Stripe
Black		0	0	x1	
Brown		1	1	x10	
Red		2	2	x100	
Orange		3	3	x1,000	
Yellow		4	4	x10,000	
Green		5	5	x100,000	
Blue		6	6	x1,000,000	
Purple		7	7		
Gray		8	8		
White		9	9		
Gold					5%
Silver					10%

Follow the procedure above with the examples below and soon you will be able to quickly determine the value of a resistor by just a glance at the color coded stripes.

Example 1:

You are given a resistor whose stripes are colored from left to right as brown, black, orange, gold. Find the resistance value.

Step One: The gold stripe is on the right so go to Step Two.

Step Two: The first stripe is brown which has a value of 1. The second stripe is black which has a value of 0. Therefore, the first two digits of the resistance value are 10.

Step Three: The third stripe is orange which means $\times 1,000$.

Step Four: The value of the resistance is found as $10 \times 1000 = 10,000$ ohms (10 kilo ohms = 10 k ohms). The gold stripe means the actual value of the resistor may vary by 5% meaning the actual value will be somewhere between 9,500 ohms and 10,500 ohms. (Since 5% of $10,000 = 0.05 \times 10,000 = 500$)

Example 2:

You are given a resistor whose stripes are colored from left to right as orange, orange, brown, silver. Find the resistance value.

Step One: The silver stripe is on the right so go to Step Two.

Step Two: The first stripe is orange which has a value of 3. The second stripe is orange which has a value of 3. Therefore, the first two digits of the resistance value are 33.

Step Three: The third stripe is brown which means $\times 10$.

Step Four: The value of the resistance is found as $33 \times 10 = 330$ ohms. The silver stripe means the actual value of the resistor may vary by 10% meaning the actual value will be between 297 ohms and 363 ohms. (Since 10% of $330 = 0.10 \times 330 = 33$)

Example 3:

You are given a resistor whose stripes are colored from left to right as blue, gray, red, gold. Find the resistance value.

Step One: The gold stripe is on the right so go to Step Two.

Step Two: The first stripe is blue which has a value of 6. The second stripe is gray which has a value of 8. Therefore, the first two digits of the resistance value are 68.

Step Three: The third stripe is red which means $\times 100$.

Step Four: The value of the resistance is found as $68 \times 100 = 6800$ ohms (6.8 kilo ohms = 6.8 k ohms). The gold stripe means the actual value of the resistor may vary by 5% meaning the actual value will be somewhere between 6,460 ohms and 7,140 ohms. (Since 5% of $6,800 = 0.05 \times 6,800 = 340$)

Example 4:

You are given a resistor whose stripes are colored from left to right as green, brown, black, gold. Find the resistance value.

Step One: The gold stripe is on the right so go to Step Two.

Step Two: The first stripe is green which has a value of 5. The second stripe is brown which has a value of 1. Therefore, the first two digits of the resistance value are 51.

Step Three: The third stripe is black which means $\times 1$.

Step Four: The value of the resistance is found as $51 \times 1 = 51$ ohms. The gold stripe means the actual value of the resistor may vary by 5% meaning the actual value will be somewhere between 48.45 ohms and 53.55 ohms. (Since 5% of $51 = 0.05 \times 51 = 2.55$)

1.3.2 Resistor Rules

There are some more rules that may be useful when working with resistors. You do not need to know them but if you need a resistor with a value that you do not have, you may be able to use the following information to create the value of resistor you need.

First Rule for Resistors : Series Connection

When two resistors are connected in series, as shown in Figure 4, the new resistance between points A and B is $R_1 + R_2$.

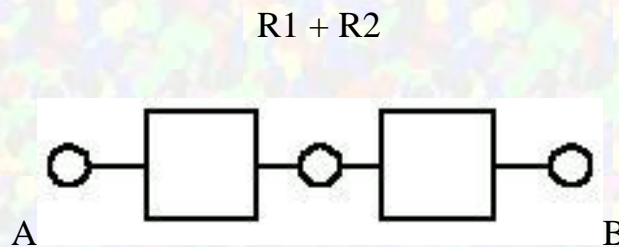


Figure 4

The resistors add together. For example if $R_1 = 500$ ohms and $R_2 = 250$ ohms then the resistance between points A and B would be $R_1 + R_2 = 500 + 250 = 750$ ohms. Section 1.5 has a picture that shows what a series connection of two resistors looks like on a breadboard.

Second Rule for Resistors : Parallel Connection

When two resistors are connected in parallel, as shown in Figure 5, the new resistance is smaller than either R1 or R2. The new resistance between points A and B is $(R1 \times R2) / (R1 + R2)$.

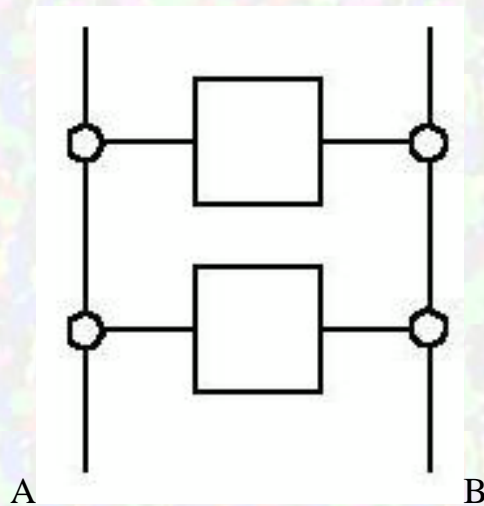


Figure 5

For example, if $R1 = 500$ and $R2 = 250$ then the resistance between points A and B = $(500 \times 250) / (500 + 250) = (125,000) / (750) = 167$ ohms. If $R1 = R2$ then the new resistance is just $R1 / 2$. Using these two rules, resistors can be combined to form new resistance values. Section 1.5 has a picture that shows what a parallel connection of two resistors looks like on a breadboard.

1.4 Finding Voltage and Current Using Ohm's Law

Now that you are familiar with resistance, we can explore how resistors are used to control voltage and current in electric circuits. There is a simple relationship between current, voltage and resistance. This relationship is called Ohm's Law.

The formula is the following.

$$\text{Difference in Voltage} = \text{Current} * \text{Resistance}$$

$$\text{or } DV = I * R$$

(in units of measurement, it looks like Volts = Amps * Ohms)

This is **Form 1** of Ohm's Law.

To find current and resistance the following forms can be used. They are the same as the above formula but in a different form.

$$\text{Form 2: Current} = \text{Difference in Voltage} / \text{Resistance}$$

$$\text{or } I = DV / R$$

(units are Amps = Volts / Ohms)

Form 3: Resistance = Difference in Voltage / Current

$$\text{or } R = DV / I$$

(units are Ohms = Volts / Amps)

These formulas are always used for situations where there are two points with a resistor between them. DV is the difference in voltage between the two points and current is what flows between the two points. These simple relationships allow us to calculate many things. Given any two of the three values (Current, Resistance, and Difference in Voltage) the third can be found. The most common calculation is for current. Voltage is easy to measure and the resistance can be found from the resistor (see color codes). Once these values are known, current can be calculated using Form 2 of Ohm's law, $I = DV / R$.

For example, consider the problem shown in Figure 6. One side is at 0 volts (ground) and the other side is at 5 volts (with a multimeter, black probe on right side, red probe on left side). The squiggly line is the symbol we use to represent a resistor.

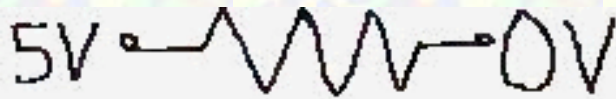


Figure 6

The voltage difference between Point A and Point B is $5 - 0 = 5$ volts ($DV=5$). Assume the resistor between the two points has a value of 500 ohms ($R=500$). We know that current flows from a point of high voltage to a point of low voltage so we can draw an arrow from the higher voltage to the lower voltage.



Figure 7

Now we can find the current flowing through the resistor by using Form 2 of Ohm's Law.

$$I = DV / R$$

If we put in values for DV and R we get

$$I = DV / R = 5 / 500$$

$$I = 5 / 500 = 0.01 \text{ Amps}$$

$$I = 0.01 \text{ Amps} = 10 \text{ milliAmps}$$

10 milliamps can be abbreviated as 10 mA

This means the current is 10 mA. ($I = 10\text{mA}$)

Now to check our answer we can use Form 1 and Form 3 of Ohm's law. We have to use the value of current in Amps for these formulas. So if we have $I = 0.01$ Amps and Resistance = 500 ohms then by using Form 1 of Ohm's law we can find:

$$DV = I * R$$

$$DV = I * R = 0.01 * 500$$

$$DV = 0.01 * 500 = 5 \text{ volts}$$

$$\text{So } DV = 5 \text{ volts}$$

5 volts is the voltage we started with so the value we found for the current must be correct. We can also check the answer with Form 3 by using $I = 0.01$ Amps and $DV = 5$ volts.

$$R = DV / I$$

$$R = DV / I = 5 / 0.01$$

$$R = 5 / 0.01 = 500 \text{ ohms}$$

$$\text{So } R = 500 \text{ ohms}$$

Now consider the problem shown in Figure 8. The voltage on one side is 10 volts and the voltage on the other side is 3 volts. Therefore the voltage difference between the two points is $10 - 3 = 7$ volts ($DV = 7 \text{ V}$). The resistor is 400 ohms ($R = 400$).

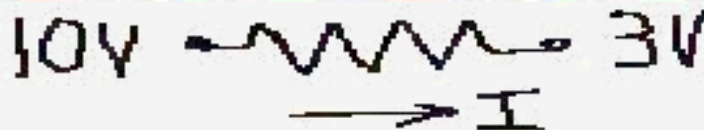


Figure 8

Then the current flowing from left to right is

$$I = DV / R$$

$$I = DV / R = 7 / 400$$

$$I = 7 / 400 = 0.0175 \text{ Amps}$$

$$I = 0.0175 \text{ Amps} = 17.5 \text{ milliAmps}$$

$$I = 17.5 \text{ milliAmps} = 17.5 \text{ mA}$$

This means the current flowing from the left to the right is 17.5 mA.

Now suppose we have two points with a voltage difference of 5 volts. Point A is at 5 volts and Point B is at 0 volts (ground). (Notice that the voltage difference is the important part. If Point A is at 7 volts and Point B is at 2 volts then the voltage difference is the same, $7 - 2 = 5$ volts.) Now suppose we want a current to flow between Points A and B and we want the current to be 0.02 Amps ($I = 0.02 \text{ Amps} = 20 \text{ mA}$). Now we need to find the value of the resistor so we use Form 3 of Ohm's Law.

$$\text{Resistance} = \text{Difference in Voltage} / \text{Current} \text{ or } R = DV / I \text{ (Ohms} = \text{Volts} / \text{Amps)}$$

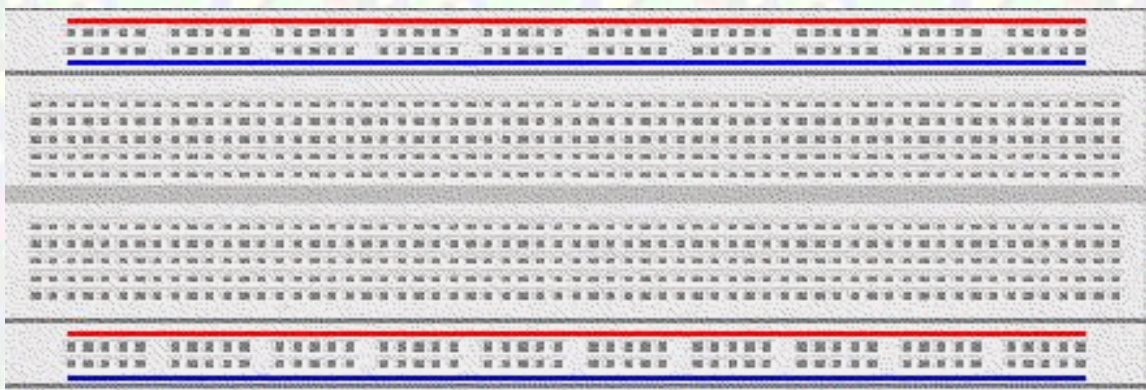
$$DV / I = 5 / 0.02 = 250 \text{ ohms}$$

This means that putting a resistor with a value of 250 ohms between Points A and B will make a current flow from Point A to Point B and the current will be 0.02 Amps (20 mA). Now using the values of voltage and resistance, check the value of the current using Form 2 of Ohm's law.

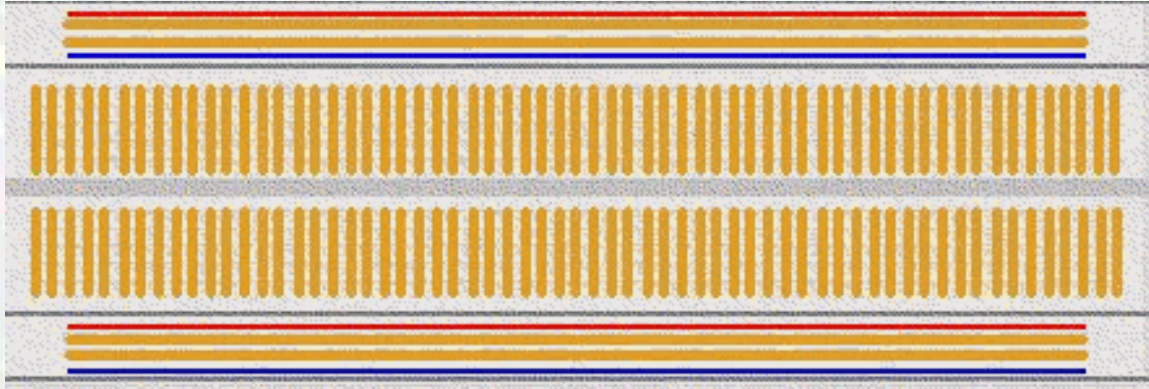
$$0.2 \text{ Amps} = 5 \text{ Volts} / 250 \text{ Ohms (Amps} = \text{Volts} / \text{Ohms)}$$

1.5 Using a Bread Board

To build our projects, we will use a breadboard like the one shown below.



The bread board has many strips of metal (copper usually) which run underneath the board. The metal strips are laid out as shown below.



These strips connect the holes on the top of the board. This makes it easy to connect components together to build circuits. To use the bread board, the legs of components are placed in the holes. The holes are made so that they will hold the component in place. Each hole is connected to one of the metal strips running underneath the hole.

Each strip forms a node. A node is a point in a circuit where two components are connected. Connections between different components are formed by putting their legs in a common node. On the bread board, a node is the row of holes that are connected by the strip of metal underneath.

The long top and bottom row of holes are usually used for power supply connections. The row with the blue strip beside it is used for the negative voltage (usually ground) and the row with the red strip beside it is used for the positive voltage.

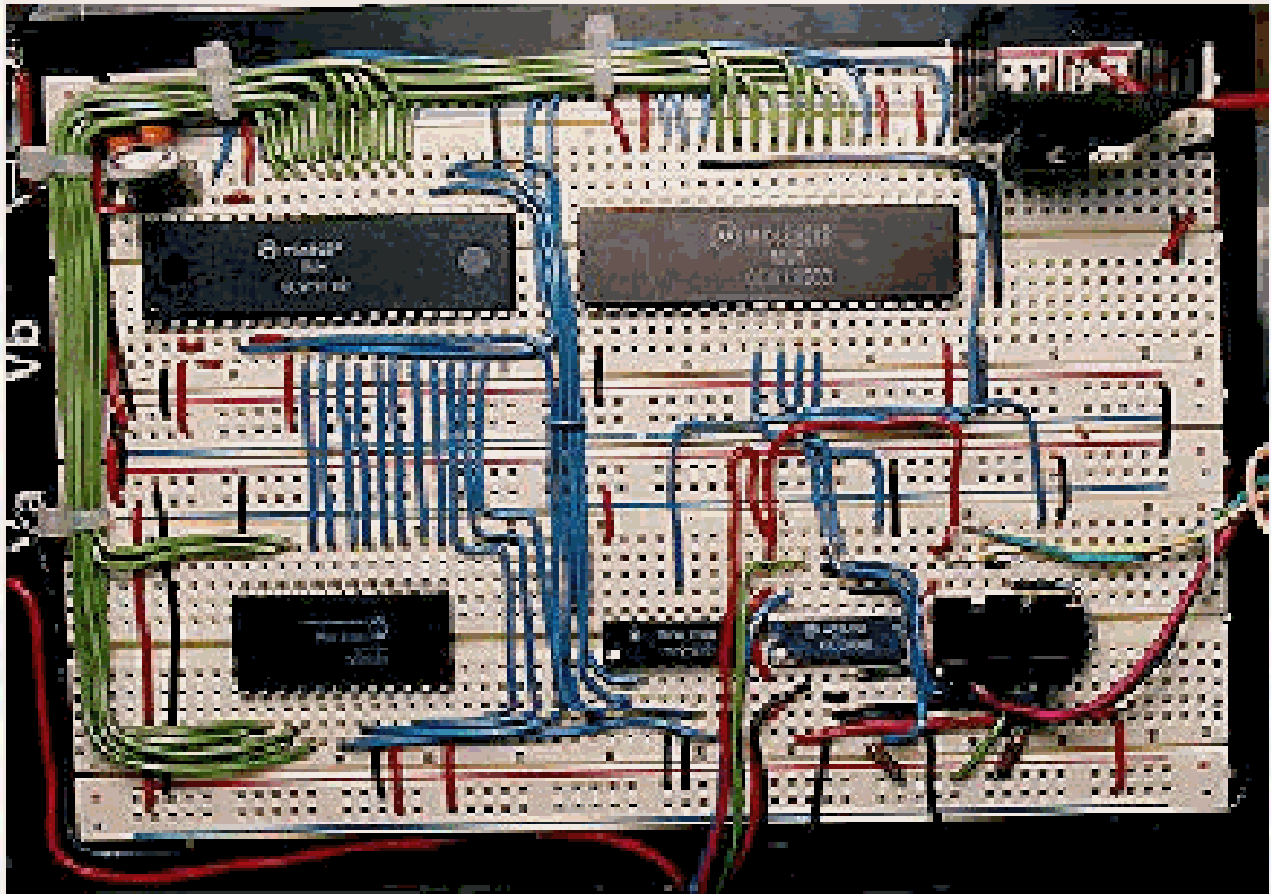
The circuit is built by placing components and connecting them together with jumper wires. Then when a path is formed from the positive supply node to the negative supply node through wires and components, we can turn on the power and current flows through the path and the circuit comes alive.

A series connection of 2 resistors on a breadboard looks like the picture below on the left and a parallel connection of 2 resistors looks like the picture below on the right.



For chips with many legs (ICs), place them in the middle of the board (across the middle dividing line) so that half of the legs are on one side of the middle line and half are on the other side.

A completed circuit might look like the following. This circuit uses two small breadboards.



1.6 Transistors and LEDs

Now we know enough that we can start to build circuits. But first we will look a little closer at a component that was introduced in Section 1.2.

The LED

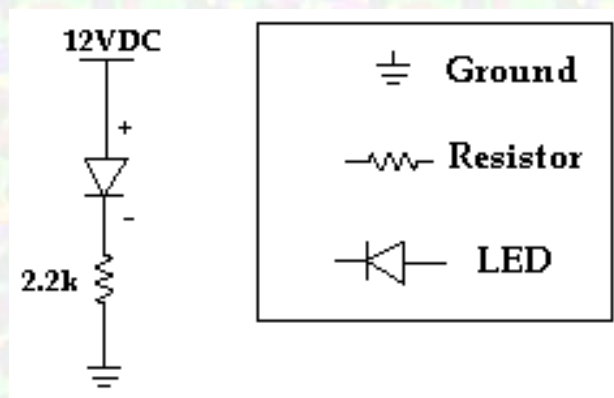


An LED is the device shown above. Besides red, they can also be yellow, green and blue. The letters LED stand for Light Emitting Diode. If you are unfamiliar with diodes, take a moment to review the components in Basic Components, Section 1.2. The important thing to remember about diodes (including LEDs) is that current can only flow in one direction.

To make an LED work, you need a voltage supply and a resistor. If you try to use an LED without a resistor, you will probably burn out the LED. The LED has very little resistance so large amounts of current will try to flow through it unless

you limit the current with a resistor. If you try to use an LED without a power supply, you will be highly disappointed.

So first of all we will make our LED light up by setting up the circuit below.



Step 1.) First you have to find the positive leg of the LED. The easiest way to do this is to look for the leg that is longer.

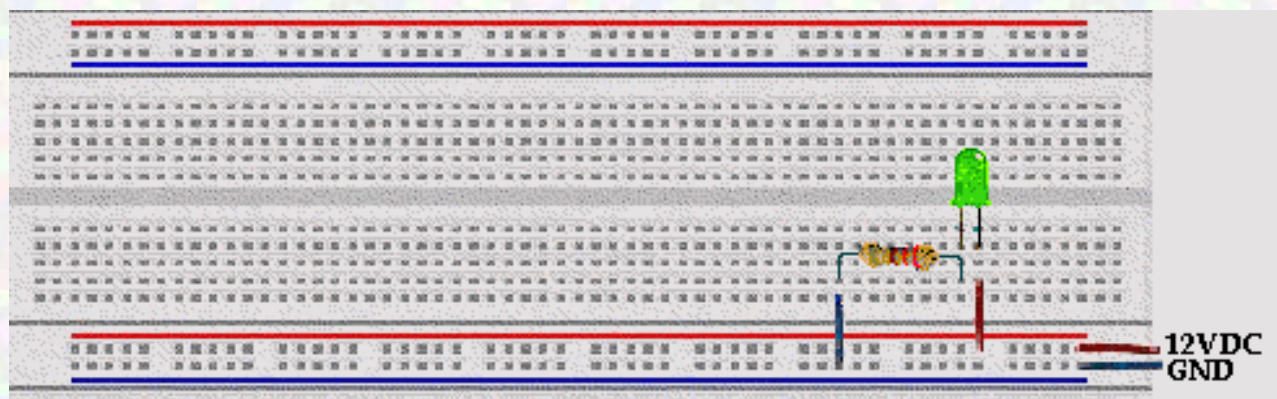
Step 2.) Once you know which side is positive, put the LED on your breadboard so the positive leg is in one row and the negative leg is in another row. (In the picture below the rows are vertical.)

Step 3.) Place one leg of a 2.2k ohm resistor (does not matter which leg) in the same row as the negative leg of the LED. Then place the other leg of the resistor in an empty row.

Step 4.) Unplug the power supply adapter from the power supply. Next, put the ground (black wire) end of the power supply adapter in the sideways row with the blue stripe beside it. Then put the positive (red wire) end of the power supply adapter in the sideways row with the red stripe beside it.

Step 5.) Use a short jumper wire (use red since it will be connected to the positive voltage) to go from the positive power row (the one with the red stripe beside it) to the positive leg of the LED (not in the same hole, but in the same row). Use another short jumper wire (use black) to go from the ground row to the resistor (the leg that is not connected to the LED). Refer to the picture below if necessary.

The breadboard should look like the picture shown below.



Now plug the power supply into the wall and then plug the other end into the power supply adapter and the LED should light

up. Current is flowing from the positive leg of the LED through the LED to the negative leg. Try turning the LED around. It should not light up. No current can flow from the negative leg of the LED to the positive leg.

People often think that the resistor must come first in the path from positive to negative, to limit the amount of current flowing through the LED. But, the current is limited by the resistor no matter where the resistor is. Even when you first turn on the power, the current will be limited to a certain amount, and can be found using ohm's law.

Revisiting Ohm's Law

Ohm's Law can be used with resistors to find the current flowing through a circuit. The law is $I = VD/R$ (where I = current, VD = voltage across resistor, and R = resistance). For the circuit above we can only use Ohm's law for the resistor so we must use the fact that when the LED is on, there is a 1.4 voltage drop across it. This means that if the positive leg is connected to 12 volts, the negative leg will be at 10.6 volts. Now we know the voltage on both sides of the resistor and can use Ohm's law to calculate the current. The current is $(10.6 - 0) / 2200 = 0.0048$ Amperes = 4.8 mA

This is the current flowing through the path from 12V to GND. This means that 4.8 mA is flowing through the LED and the resistor. If we want to change the current flowing through the LED (changing the brightness) we can change the resistor. A smaller resistor will let more current flow and a larger resistor will let less current flow. Be careful when using smaller resistors because they will get hot.

Next, we want to be able to turn the LED on and off without changing the circuit. To do this we will learn to use another electronic component, the transistor.

1.6.1 The Transistor

Transistors are basic components in all of today's electronics. They are just simple switches that we can use to turn things on and off. Even though they are simple, they are the most important electrical component. For example, transistors are almost the only components used to build a Pentium processor. A single Pentium chip has about 3.5 million transistors. The ones in the Pentium are smaller than the ones we will use but they work the same way.

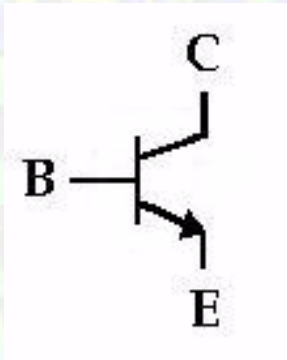
Transistors that we will use in projects look like this:



The transistor has three legs, the Collector (C), Base (B), and Emitter (E). Sometimes they are labeled on the flat side of the transistor. Transistors always have one round side and one flat side. If the round side is facing you, the Collector leg is on the left, the Base leg is in the middle, and the Emitter leg is on the right.

Transistor Symbol

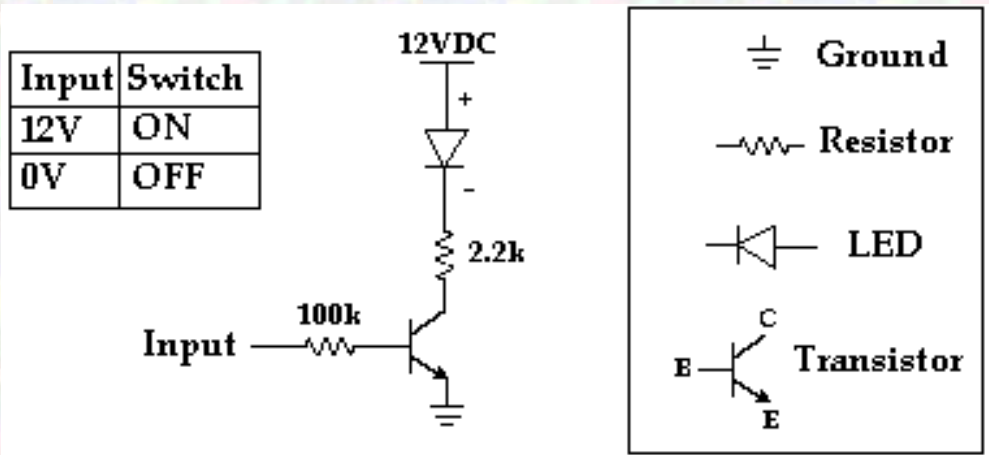
The following symbol is used in circuit drawings (schematics) to represent a transistor.



Basic Circuit

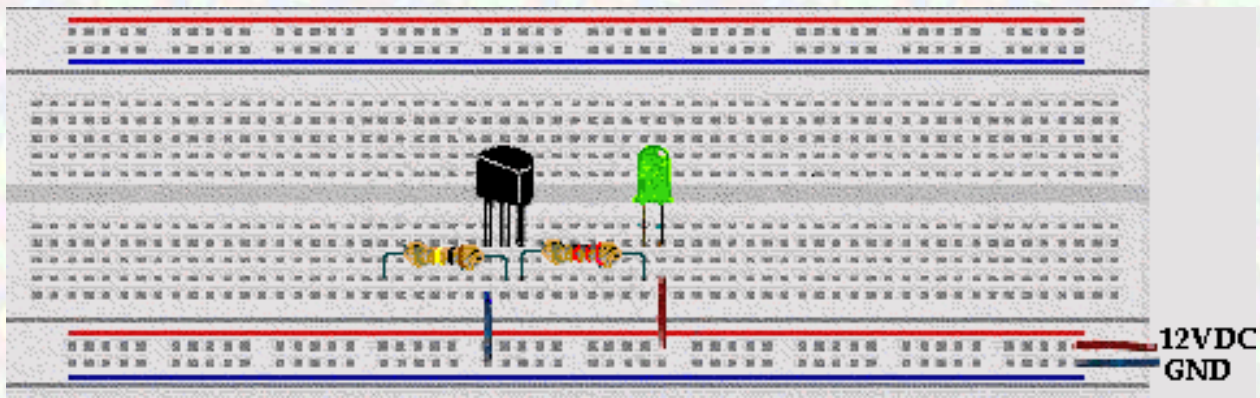
The Base (B) is the On/Off switch for the transistor. If a current is flowing to the Base, there will be a path from the Collector (C) to the Emitter (E) where current can flow (The Switch is On.) If there is no current flowing to the Base, then no current can flow from the Collector to the Emitter. (The Switch is Off.)

Below is the basic circuit we will use for all of our transistors.



To build this circuit we only need to add the transistor and another resistor to the circuit we built above for the LED. Unplug the power supply from the power supply adapter before making any changes on the breadboard. To put the transistor in the breadboard, separate the legs slightly and place it on the breadboard so each leg is in a different row. The collector leg should be in the same row as the leg of the resistor that is connected to ground (with the black jumper wire). Next move the jumper wire going from ground to the 2.2k ohm resistor to the Emitter of the transistor.

Next place one leg of the 100k ohm resistor in the row with the Base of the transistor and the other leg in an empty row and your breadboard should look like the picture below.



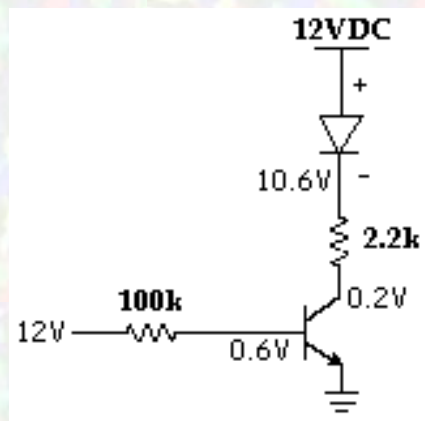
Now put one end of a yellow jumper wire in the positive row (beside the red line) and the other end in the row with the leg of the 100k ohm resistor (the end not connected to the Base). Reconnect the power supply and the transistor will come on and the LED will light up. Now move the one end of the yellow jumper wire from the positive row to the ground row (beside the blue line). As soon as you remove the yellow jumper wire from the positive power supply, there is no current flowing to the base. This makes the transistor turn off and current can not flow through the LED. As we will see later, there is very little current flowing through the 100k resistor. This is very important because it means we can control a large current in one part of the circuit (the current flowing through the LED) with only a small current from the input.

Back to Ohm's Law

We want to use Ohm's law to find the current in the path from the Input to the Base of the transistor and the current flowing through the LED. To do this we need to use two basic facts about the transistor.

- 1.) If the transistor is on, then the Base voltage is 0.6 volts higher than the Emitter voltage.
- 2.) If the transistor is on, the Collector voltage is 0.2 volts higher than the Emitter voltage.

So when the 100k resistor is connected to 12VDC, the circuit will look like this:



So the current flowing through the 100k resistor is $(12 - 0.6) / 100000 = 0.000114 \text{ A} = 0.114 \text{ mA}$.

The current flowing through the 2.2k ohm resistor is $(10.6 - 0.2) / 2200 = 0.0047 \text{ A} = 4.7 \text{ mA}$.

If we want more current flowing through the LED, we can use a smaller resistor (instead of 2200) and we will get more

current through the LED without changing the amount of current that comes from the Input line. This means we can control things that use a lot of power (like electric motors) with cheap, low power circuits. Soon you will learn how to use a microcontroller (a simple computer). Even though the microcontroller can not supply enough current to turn lights and motors on and off, the microcontroller can turn transistors on and off and the transistors can control lots of current for lights and motors.

For Ohm's law, also remember that when the transistor is off, no current flows through the transistor.

1.6.2 Introduction to Digital Devices - The Inverter

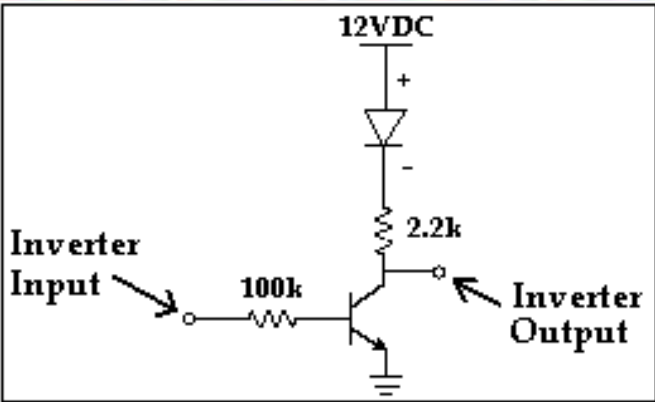
In digital devices there are only two values, usually referred to as 0 and 1. 1 means there is a voltage (usually 5 volts) and 0 means the voltage is 0 volts.

An inverter (also called a NOT gate) is a basic digital device found in all modern electronics. So for an inverter, as the name suggests, it's output is the opposite of the input (Output is NOT the Input). If the input is 0 then the output is 1 and if the input is 1 then the output is 0. We can summarize the operation of this device in a table.

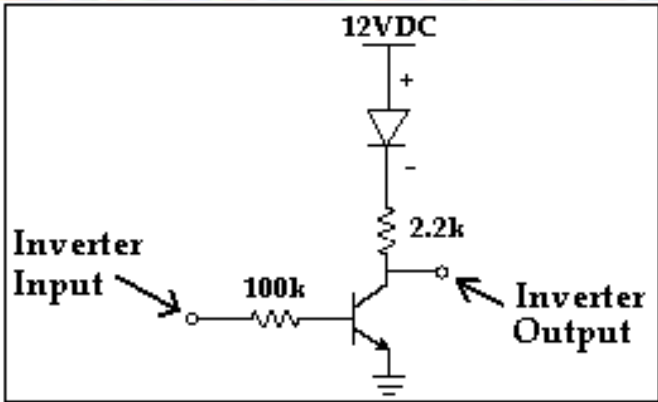
Input	Output
1	0
0	1

To help us practice with transistors we will build an inverter. Actually we have already built an inverter. The transistor circuit we just built is an inverter circuit. To help see the inverter working, we will build a circuit with two inverters. The circuit we will use is shown below.

First Inverter (already built)



Second Inverter



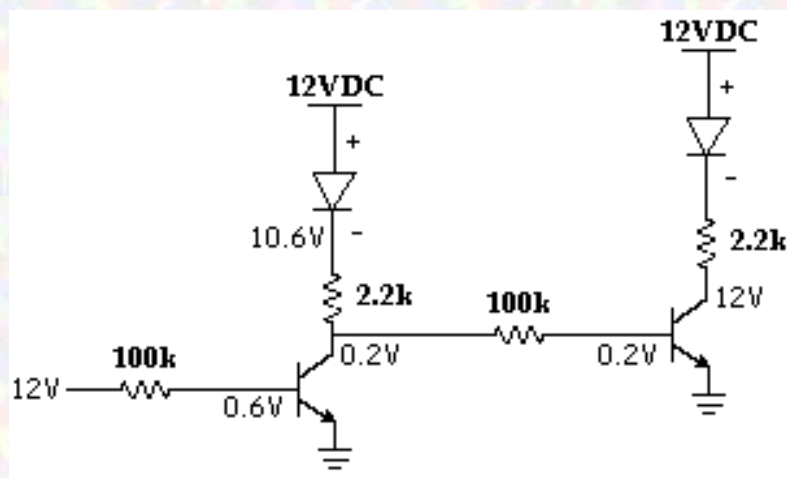
To build the circuit, use the transistor circuit we just built as the first inverter. The first inverter input is the end of the 100k ohm resistor connected to the yellow jumper wire. Build another circuit identical to the first (the basic transistor circuit from Section 1.6.1) except leave out the yellow jumper wire connected to the 100k ohm resistor (the inverter input). This circuit is

the second inverter.

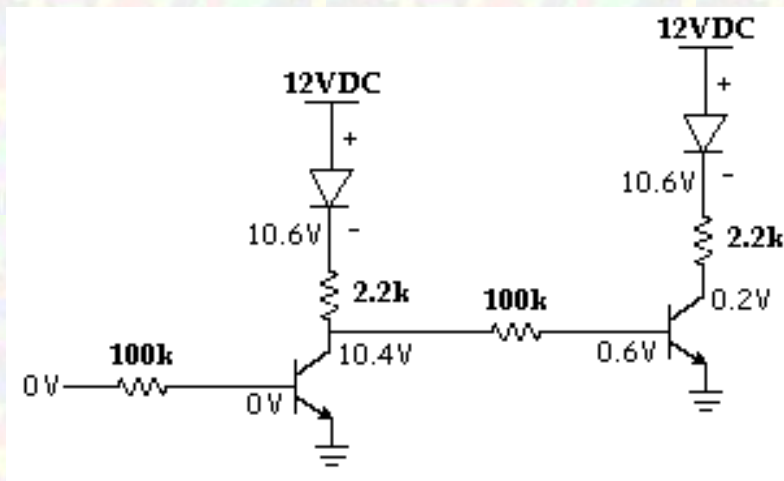
Connect the output of the first inverter to the input of the second inverter by putting one end of a jumper wire in the same row of holes as the 2.2k ohm resistor and the Collector of the transistor (the output of the first inverter) and putting the other end in the same row of holes as the leg of the 100k ohm resistor of the second inverter (the input to the second inverter).

Here is how to check if you built it correctly. Connect the first inverter input (the yellow jumper wire) to 12V (the positive row). The LED in the first inverter should come on and the LED in the second inverter should stay off. Then connect the first inverter input to 0V (the ground row). (You are turning off the switch of the first inverter.) The first LED should go off and the second LED should come on. If this does not happen, check to make sure no metal parts are touching. Check to make sure all the parts are connected correctly.

The input can either be connected to 12V or 0V. When the Inverter Input is 12V, the transistor in the first inverter will turn on and the LED will come on and the Inverter Output voltage will be 0.2V. The first Inverter Output is connected to the input of the second inverter. The 0.2V at the input of the second inverter is small enough that the second transistor is turned off. The circuit voltages are shown in the diagram below.

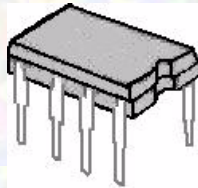


When the Inverter Input is connected to 0V, the transistor in the first inverter is turned off and the LED will get very dim. There is a small amount of current still flowing through the LED to the second inverter. The voltage at the first Inverter Output will go up, forcing the second inverter transistor to come on. When the second inverter transistor comes on, the second inverter LED will come on. To find the voltage at the output of the first inverter (10.4V), use Ohm's law. There is no current flowing through the transistor in the first inverter so the path of the current is through the first LED, through the 2.2k resistor, through the 100k resistor, through the second transistor to ground. The voltage at the negative side of the first LED is fixed at 10.6V by the LED. The voltage at the second transistor base is fixed at 0.6V by the transistor. Then given those two voltages, you should be able to find the voltage at the point in the middle (10.4V) using Ohm's law. (Hint: First find the current and then work through Form 1 of ohm's law to find the voltage at the point between the 2.2k resistor and the 100k resistor.)



Switch the input back and forth from 0V to 12V and you can see that when the first stage is on, the second stage is off. This demonstrates the inverting action of the Inverter.

1.7 Oscillators, Pulse Generators, Clocks... Capacitors and the 555 Timer IC

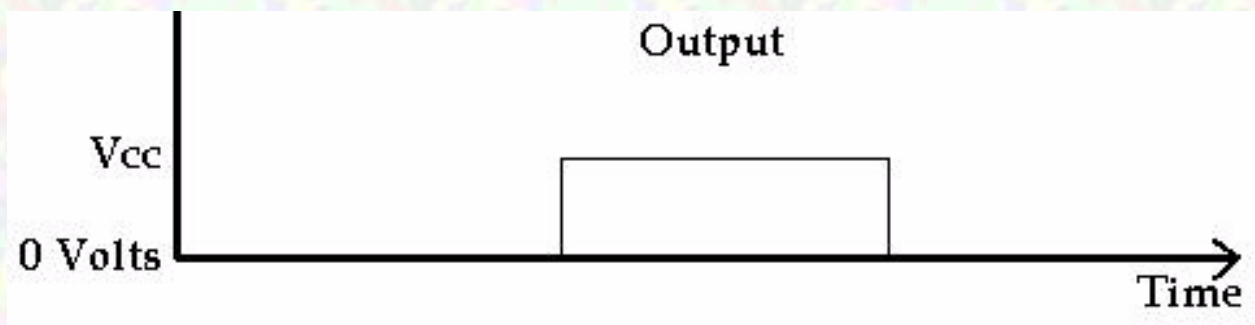


Introduction

As electronic designs get bigger, it becomes difficult to build the complete circuit. So we will use prebuilt circuits that come in packages like the one shown above. This prebuilt circuit is called an IC. IC stands for Integrated Circuit. An IC has many transistors inside it that are connected together to form a circuit. Metal pins are connected to the circuit and the circuit is stuck into a piece of plastic or ceramic so that the metal pins are sticking out of the side. These pins allow you to connect other devices to the circuit inside. We can buy simple ICs that have several inverter circuits like the one we built in the LED and Transistor section or we can buy complex ICs like a Pentium Processor.

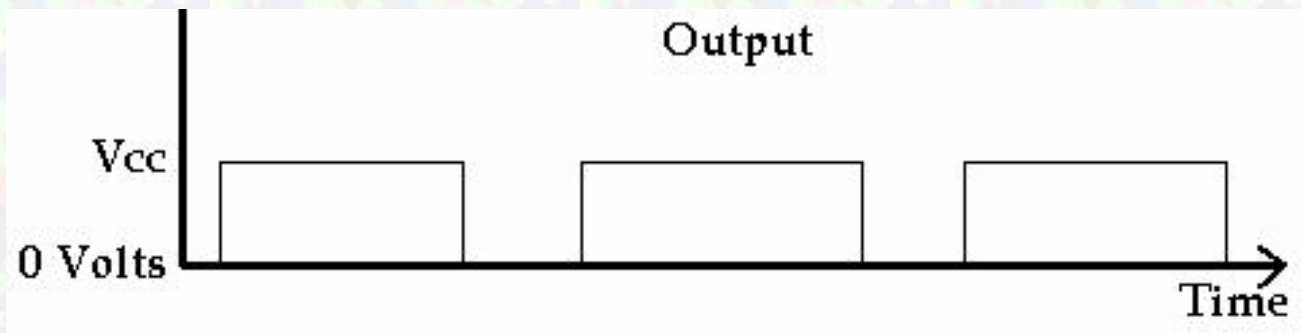
The Pulse - More than just an on/off switch

So far the circuits we have built have been stable, meaning that the output voltage stays the same. If you change the input voltage, the output voltage changes and once it changes it will stay at the same voltage level. The 555 integrated circuit (IC) is designed so that when the input changes, the output goes from 0 volts to V_{cc} (where V_{cc} is the voltage of the power supply). Then the output stays at V_{cc} for a certain length of time and then it goes back to 0 volts. This is a pulse. A graph of the output voltage is shown below.



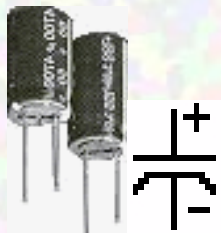
The Oscillator (A Clock) - More than just a Pulse

The pulse is nice but it only happens one time. If you want something that does something interesting forever rather than just once, you need an oscillator. An oscillator puts out an endless series of pulses. The output constantly goes from 0 volts to V_{cc} and back to 0 volts again. Almost all digital circuits have some type of oscillator. This stream of output pulses is often called a clock. You can count the number of pulses to tell how much time has gone by. We will see how the 555 timer can be used to generate this clock. A graph of a clock signal is shown below.



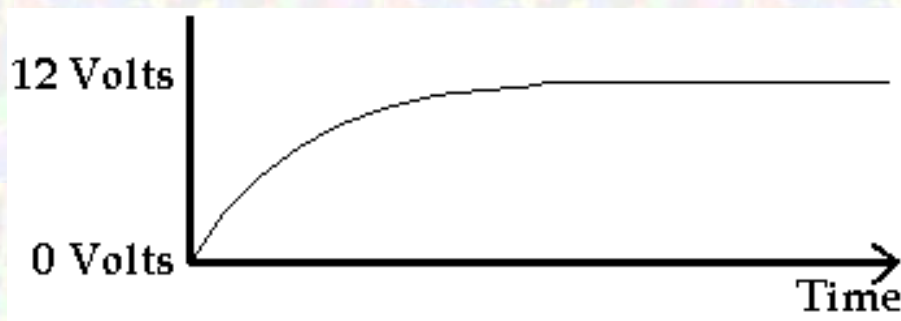
1.7.1 The Capacitor

If you already understand capacitors you can skip this part.

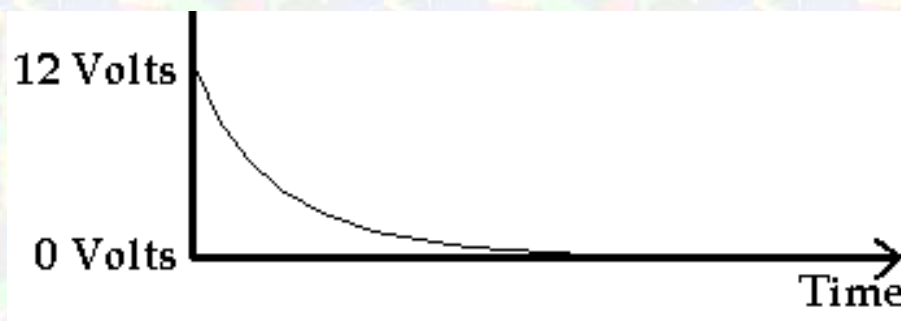


The picture above on the left shows two typical capacitors. Capacitors usually have two legs. One leg is the positive leg and the other is the negative leg. The positive leg is the one that is longer. The picture on the right is the symbol used for capacitors in circuit drawings (schematics). When you put one in a circuit, you must make sure the positive leg and the negative leg go in the right place. Capacitors do not always have a positive leg and a negative leg. The smallest capacitors in this kit do not. It does not matter which way you put them in a circuit.

A capacitor is similar to a rechargeable battery in the way it works. The difference is that a capacitor can only hold a small fraction of the energy that a battery can. (Except for really big capacitors like the ones found in old TVs. These can hold a lot of charge. Even if a TV has been disconnected from the wall for a long time, these capacitors can still make lots of sparks and hurt people.) As with a rechargeable battery, it takes a while for the capacitor to charge. So if we have a 12 volt supply and start charging the capacitor, it will start with 0 volts and go from 0 volts to 12 volts. Below is a graph of the voltage in the capacitor while it is charging.



The same idea is true when the capacitor is discharging. If the capacitor has been charged to 12 volts and then we connect both legs to ground, the capacitor will start discharging but it will take some time for the voltage to go to 0 volts. Below is a graph of what the voltage is in the capacitor while it is discharging.



We can control the speed of the capacitor's charging and discharging using resistors.

Capacitors are given values based on how much electricity they can store. Larger capacitors can store more energy and take more time to charge and discharge. The values are given in Farads but a Farad is a really large unit of measure for common capacitors. In this kit we have 2 33pf capacitors, 2 10uf capacitors and 2 220uF capacitors. Pf means picofarad and uf means microfarad. A picofarad is 0.000000000001 Farads. So the 33pf capacitor has a value of 33 picofarads or 0.000000000033 Farads. A microfarad is 0.000001 Farads. So the 10uf capacitor is 0.00001 Farads and the 220uF capacitor is 0.000220 Farads. If you do any calculations using the value of the capacitor you have to use the Farad value rather than the picofarad or microfarad value.

Capacitors are also rated by the maximum voltage they can take. This value is always written on the larger can shaped capacitors. For example, the 220uF capacitors in this kit have a maximum voltage rating of 25 volts. If you apply more than 25 volts to them they will die. We don't have to worry about that with this kit because our power supply can only put out 12 volts.

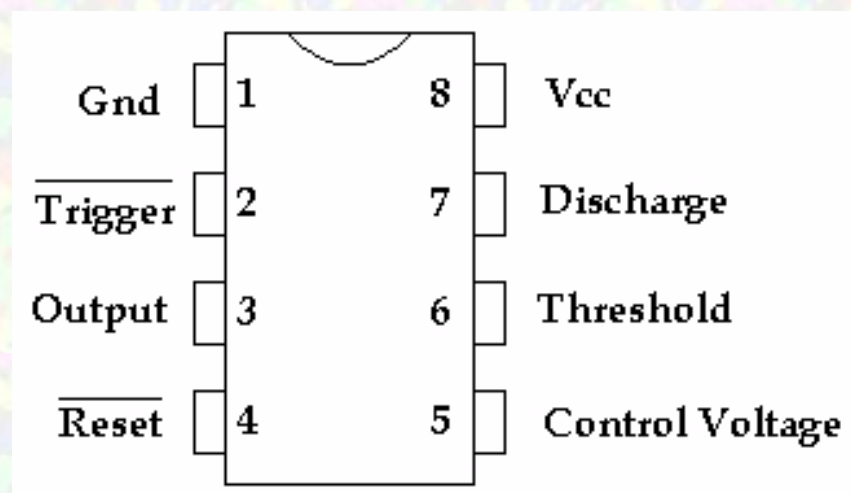
1.7.2 The 555 Timer

Creating a Pulse

The 555 is made out of simple transistors that are about the same as on / off switches. They do not have any sense of time. When you apply a voltage they turn on and when you take away the voltage they turn off. So by itself, the 555 can not create a pulse. The way the pulse is created is by using some components in a circuit attached to the 555 (see the circuit below). This circuit is made of a capacitor and a resistor. We can flip a switch and start charging the capacitor. The resistor is used to control how fast the capacitor charges. The bigger the resistance, the longer it takes to charge the capacitor. The voltage in the capacitor can then be used as an input to another switch. Since the voltage starts at 0, nothing happens to the second switch. But eventually the capacitor will charge up to some point where the second switch comes on.

The way the 555 timer works is that when you flip the first switch, the **Output** pin goes to Vcc (the positive power supply voltage) and starts charging the capacitor. When the capacitor voltage gets to $\frac{2}{3} V_{cc}$ (that is $V_{cc} * \frac{2}{3}$) the second switch turns on which makes the output go to 0 volts.

The pinout for the 555 timer is shown below



Deep Details

Pin 2 (Trigger) is the 'on' switch for the pulse. The line over the word Trigger tells us that the voltage levels are the opposite of what you would normally expect. To turn the switch on you apply 0 volts to pin 2. The technical term for this opposite behavior is 'Active Low'. It is common to see this 'Active Low' behavior for IC inputs because of the inverting nature of transistor circuits like we saw in the LED and Transistor Tutorial.

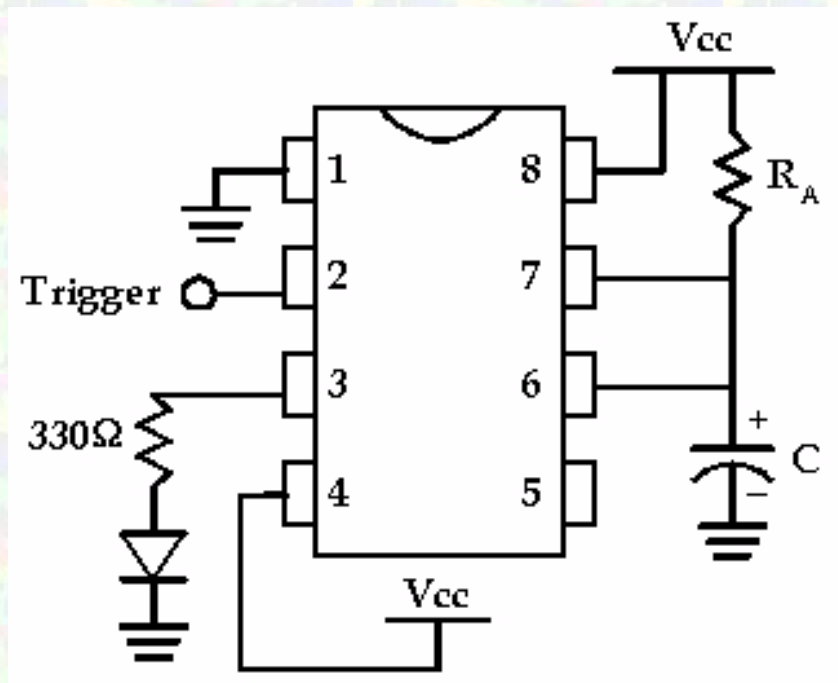
Pin 6 is the off switch for the pulse. We connect the positive side of the capacitor to this pin and the negative side of the capacitor to ground. When Pin 2 (Trigger) is at V_{cc} , the 555 holds Pin 7 at 0 volts (Note the inverted voltage). When Pin 2 goes to 0 volts, the 555 stops holding Pin 7 at 0 volts. Then the capacitor starts charging. The capacitor is charged through a resistor connected to V_{cc} . The current starts flowing into the capacitor, and the voltage in the capacitor starts to increase.

Pin 3 is the output (where the actual pulse comes out). The voltage on this pin starts at 0 volts. When 0 volts is applied to the trigger (Pin 2), the 555 puts out V_{cc} on Pin 3 and holds it at V_{cc} until Pin 6 reaches $2/3$ of V_{cc} (that is $V_{cc} * 2/3$). Then the 555 pulls the voltage at Pin 3 to ground and you have created a pulse. (Again notice the inverting action.) The voltage on Pin 7 is also pulled to ground, connecting the capacitor to ground and discharging it.

Seeing the pulse

To see the pulse we will use an LED connected to the 555 output, Pin 3. When the output is 0 volts the LED will be off. When the output is V_{cc} the LED will be on.

Building the Circuit



Place the 555 across the middle line of the breadboard so that 4 pins are on one side and 4 pins are on the other side. (You may need to bend the pins in a little so they will go in the holes.) Leave the power disconnected until you finish building the circuit. The diagram above shows how the pins on the 555 are numbered. You can find pin 1 by looking for the half circle in the end of the chip. Sometimes instead of a half circle, there will be a dot or shallow hole by pin 1.

Before you start building the circuit, use jumper wires to connect the red and blue power rows to the red and blue power rows on the other side of the board. Then you will be able to easily reach Vcc and Ground lines from both sides of the board. (If the wires are too short, use two wires joined together in a row of holes for the positive power (Vcc) and two wires joined together in a different row of holes for the ground.)

Connect Pin 1 to ground.

Connect Pin 8 to Vcc.

Connect Pin 4 to Vcc.

Connect the positive leg of the LED to a 330 ohm resistor and connect the negative end of the LED to ground. Connect the other leg of the 330 ohm resistor to the output, Pin 3.

Connect Pin 7 to Vcc with a 10k resistor ($R_A = 10K$).

Connect Pin 7 to Pin 6 with a jumper wire.

Connect Pin 6 to the positive leg of the 220uF Capacitor ($C = 220\mu F$). (You will need to bend the positive (long leg) up and out some so that the negative leg can go in the breadboard.)

Connect the negative leg of the capacitor to ground.

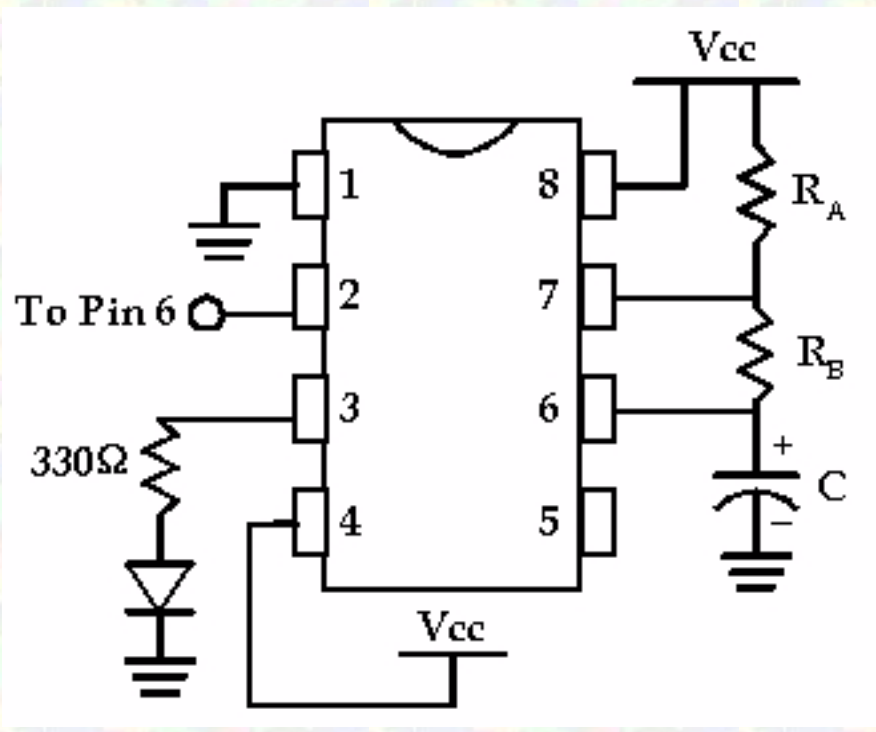
Connect a wire to Pin 2 to use as the trigger. Start with Pin 2 connected to Vcc.

Now connect the power. The LED will come on and stay on for about 2 seconds. Remove the wire connected to Pin 2 from Vcc. You should be able to trigger the 555 again by touching the wire connected to pin 2 with your finger or by connecting it to ground and removing it. (It should be about a 2 second pulse.)

Making it Oscillate

Next we will make the LED flash continually without having to trigger it. We will hook up the 555 so that it triggers itself. The way this works is that we add in a resistor between the capacitor and the discharge pin, Pin 7. Now, the capacitor will charge up (through R_A and R_B) and when it reaches $2/3$ Vcc, Pin 3 and Pin 7 will go to ground. But the capacitor can not discharge immediately because of R_B . It takes some time for the charge to drain through R_B . The more resistance R_B has, the longer it takes to discharge. The time it takes to discharge the capacitor will be the time the LED is off.

To trigger the 555 again, we connect Pin 6 to the trigger (Pin 2). As the capacitor is discharging, the voltage in the capacitor gets lower and lower. When it gets down to $1/3$ Vcc this triggers Pin 2 causing Pin 3 to go to Vcc and the LED to come on. The 555 disconnects Pin 7 from ground, and the capacitor starts to charge up again through R_A and R_B .



To build this circuit from the previous circuit, do the following.

Disconnect the power.

Take out the jumper wire between Pin 6 and Pin 7 and replace it with a 2.2k resistor ($R_B = 2.2K$).

Use the jumper wire at pin 2 to connect Pin 2 to Pin 6.

Now reconnect the power and the LED should flash forever (as long as you pay your electricity bill).

Experiment with different resistor values of R_A and R_B to see how it changes the length of time that the LED flashes. (You are changing the amount of time that it takes for the Capacitor to charge and discharge.)

Formulas

These are the formulas we use for the 555 to control the length of the pulses.

$$t_1 = \text{charge time (how long the LED is on)} = 0.693 * (R_A + R_B) * C$$

$$t_2 = \text{discharge time (how long the LED is off)} = 0.693 * R_B * C$$

$$T = \text{period} = t_1 + t_2 = 0.693 * (R_A + 2 * R_B) * C$$

$$\text{Frequency} = 1 / T = 1.44 / ((R_A + 2 * R_B) * C)$$

t1 and t2 are the time in seconds. C is the capacitor value in Farads. $220\mu\text{F} = 0.000220 \text{ F}$. So for our circuit we have:

$$t1 = 0.693 * (10000 + 2200) * 0.000220 = 1.86 \text{ seconds}$$

$$t2 = 0.693 * 2200 * 0.000220 = 0.335 \text{ seconds}$$

$$T = 1.86 + 0.335 = 2.195 \text{ seconds}$$

$$\text{Frequency} = 0.456 \text{ (cycles per second)}$$

Chapter 2: Microcontrollers

A microcontroller is an integrated circuit (IC) that is programmable. When you turn on the power to the microcontroller it goes through a series of commands. These commands are put in the chip by you. You can make it do different things by changing the commands (usually called the program). To change the commands you need a device like the [PG302](#). The [PG302](#) lets you download the program from your computer to the microcontroller. This chapter will show you some simple programs and how to download those to the microcontroller.

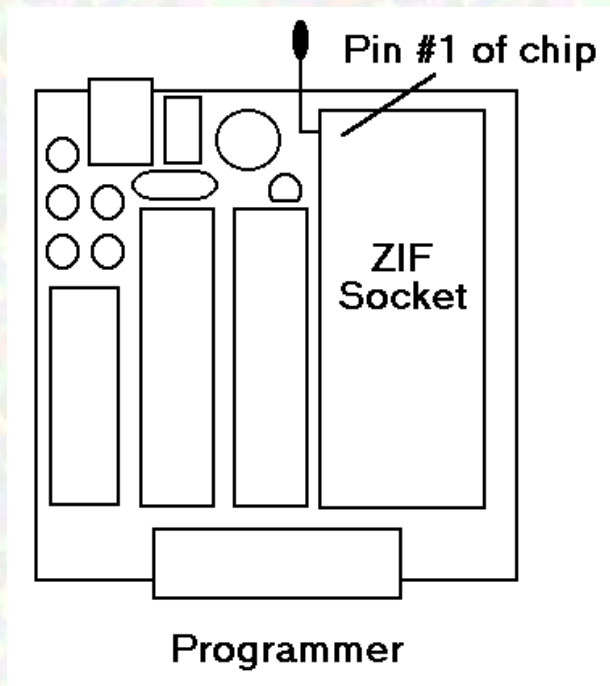
2.0.1 [PG302](#) Setup

The Software CD contains a setup.exe file in the PG302 folder. Double click on this file to run it and follow the instructions.

[\(Click here to download the software from our website.\)](#)

There are two setup options that you may want to change at some point in the future. The first option is Auto Erase. In most situations you will want to leave this on so that the chip is erased before programming it with new code. The second option is Auto Verify. This option automatically verifies that the code was programmed correctly. If a chip has not been erased before programming, this option will find that error. Leave these options on while working through the projects in this chapter.

When programming a chip, insert it in the green socket so that pin 1 of the chip is in the same corner as the handle of the socket. The handle of the green socket should be in the up (open) position. Then lock the chip into the socket by pushing the handle down into the closed position.



2.0.2 AY Pad Software Setup

AY Pad is the software we will use to view and edit the commands (programs) for the microcontroller. It is a fancy text editor that colorizes text files. Look for a file on the CD in the MBKit/AY Pad folder called `setup.exe` ([or click here](#)). Double click the `setup.exe` file. This will install AY Pad.

To run AY Pad, go to Start->Programs->AY Software->AY Pad

2.0.3 TASM Software Setup

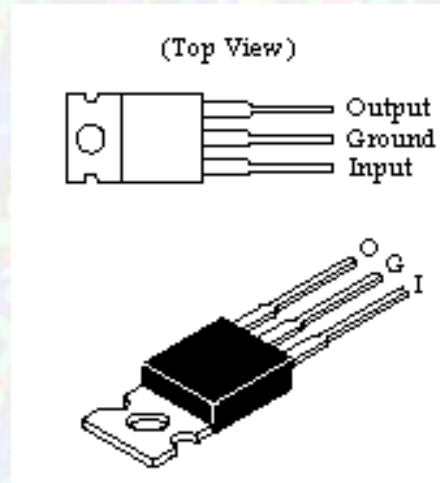
Make a new folder called *tasm* on your C hard drive that you can use for these projects. Look for a file on the CD in the MBKit folder called `tasminst.exe` ([or click here](#)). Copy this file and paste it in the new *tasm* folder on your C drive. Now double click on `tasminst.exe`. This will extract all the files we need for the projects. The main file we will use is TASM. It is called a compiler or assembler. It converts the microcontroller commands from a text version (that we can understand) to a number version (that the microcontroller can understand).

2.0.4 Chapter 2 Summary

Build the 5 volt power supply as shown in Section 2.1. You should now be ready to work through section 2.3, *Making an LED Blink*, and section 2.4, *A Simple Microcontroller System*. These show the basic steps of compiling programs and downloading them to the 2051 microcontroller. After doing these two projects you should be able to study the assembly language code and modify it to do various things with the LEDs.

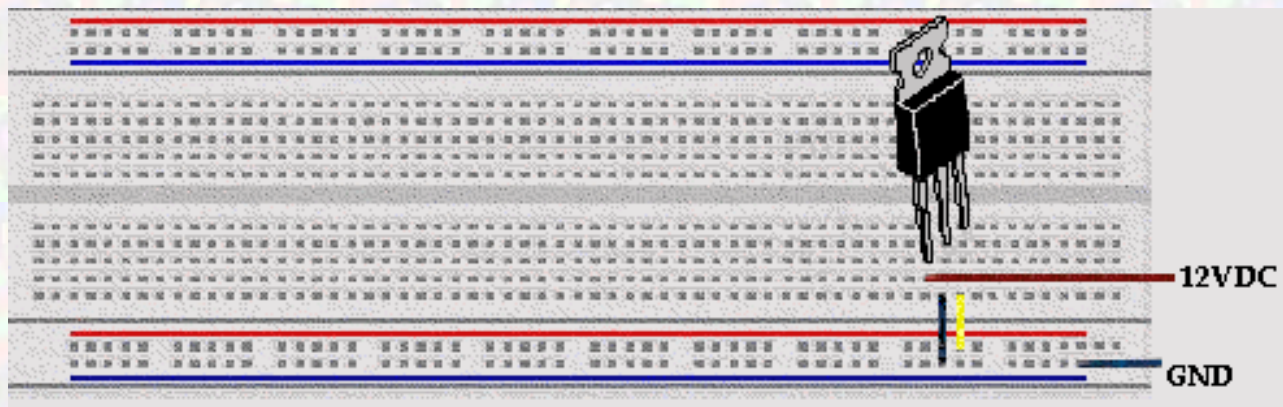
2.1 Building a 5 Volt Power Supply

Most digital logic circuits and processors need a 5 volt power supply. To use these parts we need to build a regulated 5 volt source. Usually you start with an unregulated power supply ranging from 9 volts to 24 volts DC. To make a 5 volt power supply, we use a LM7805 voltage regulator IC (Integrated Circuit). The IC is shown below.



The LM7805 is simple to use. You simply connect the positive lead of your unregulated DC power supply (anything from 9VDC to 24VDC) to the Input pin, connect the negative lead to the Ground pin and then when you turn on the power, you get a 5 volt supply from the Output pin. **This 5 volt output will be used as Vcc in the following projects.**

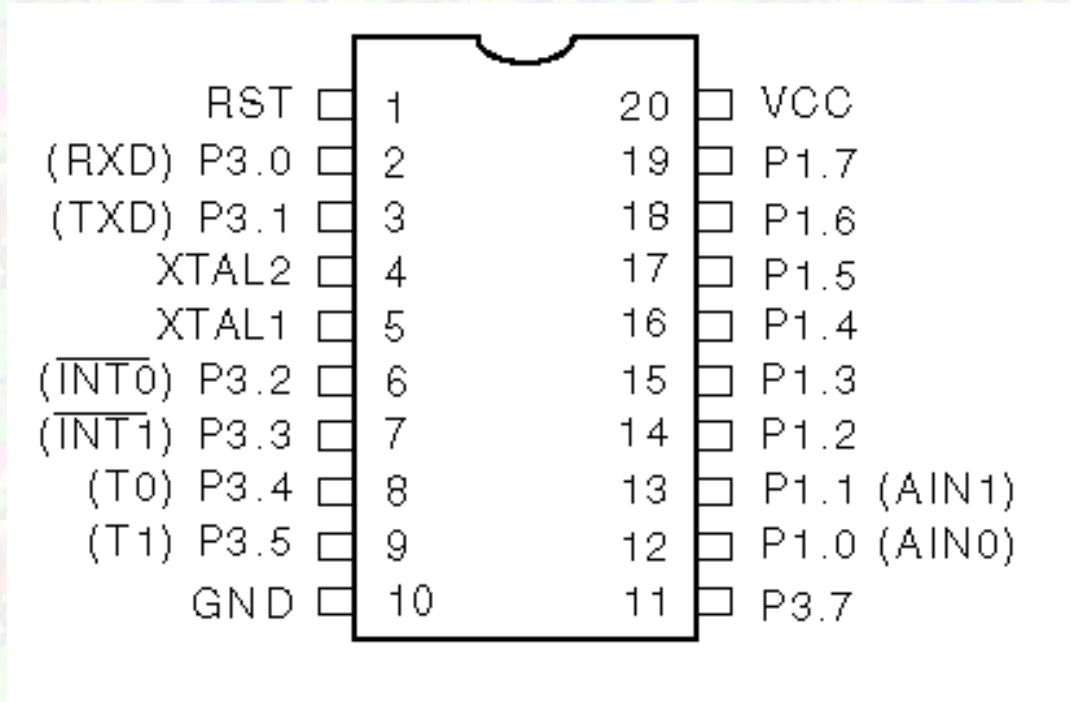
Connect the red wire from the power supply adapter to the input of the 7805. Connect the black wire from the power supply adapter to the ground row (with the blue line beside it). Run a black jumper wire from the ground row to the ground of the 7805. Then use a yellow jumper to connect the 5 volt output to the row of holes with the red stripe beside it. The breadboarded circuit is shown below.



Sometimes the input supply line (the 12VDC above) may be noisy. To help smooth out this noise and get a better 5 volt output, a capacitor is usually added to the circuit, going between the input and ground (GND). Find the 220 uF capacitor and put the long leg (positive leg) in the row of holes with the 12VDC line and put the short leg (negative leg) in ground (the row of holes next to the blue line).

2.2 The 2051 Microcontroller

The [2051 microcontroller](#) is a complex integrated circuit that is programmable. You can give it a set of commands to follow and it will run through those commands and do exactly what you want it to do. This section will give a quick overview of the pins of the 2051 and then Section 2.3 will show how to program the 2051. The 2051 is shown below.



Pin 1 is Reset. This pin can be used to force the 2051 to start over at the beginning of the program.

Pin 2 and Pin 3 can be used to communicate with the computer or other devices (RXD is receive and TXD is transmit). Pin 2 and Pin 3 are also part of Port 3. Port 3 includes P3.0, P3.1, P3.2, P3.3, P3.4, P3.5 and P3.7 (there is no P3.6). These pins are usually used as general input/output pins. They can be connected to LEDs to turn them on and off (this would be using them as outputs). Or they can be connected to switches so that the 2051 can look and see if a user has turned a switch on or off (this would be using them as inputs).

Pins 4 and 5 are connected to the 11.0592 MHz crystal. The 2051 uses this crystal to create a clock. The speed of the crystal determines the speed that the 2051 runs at. You can make programs run faster by using a faster crystal such as 24 MHz.

Pin 10 is the ground connection for the 2051.

Pins 12 to 19 make up Port 1. This is another set of pins that can be used as general inputs and outputs.

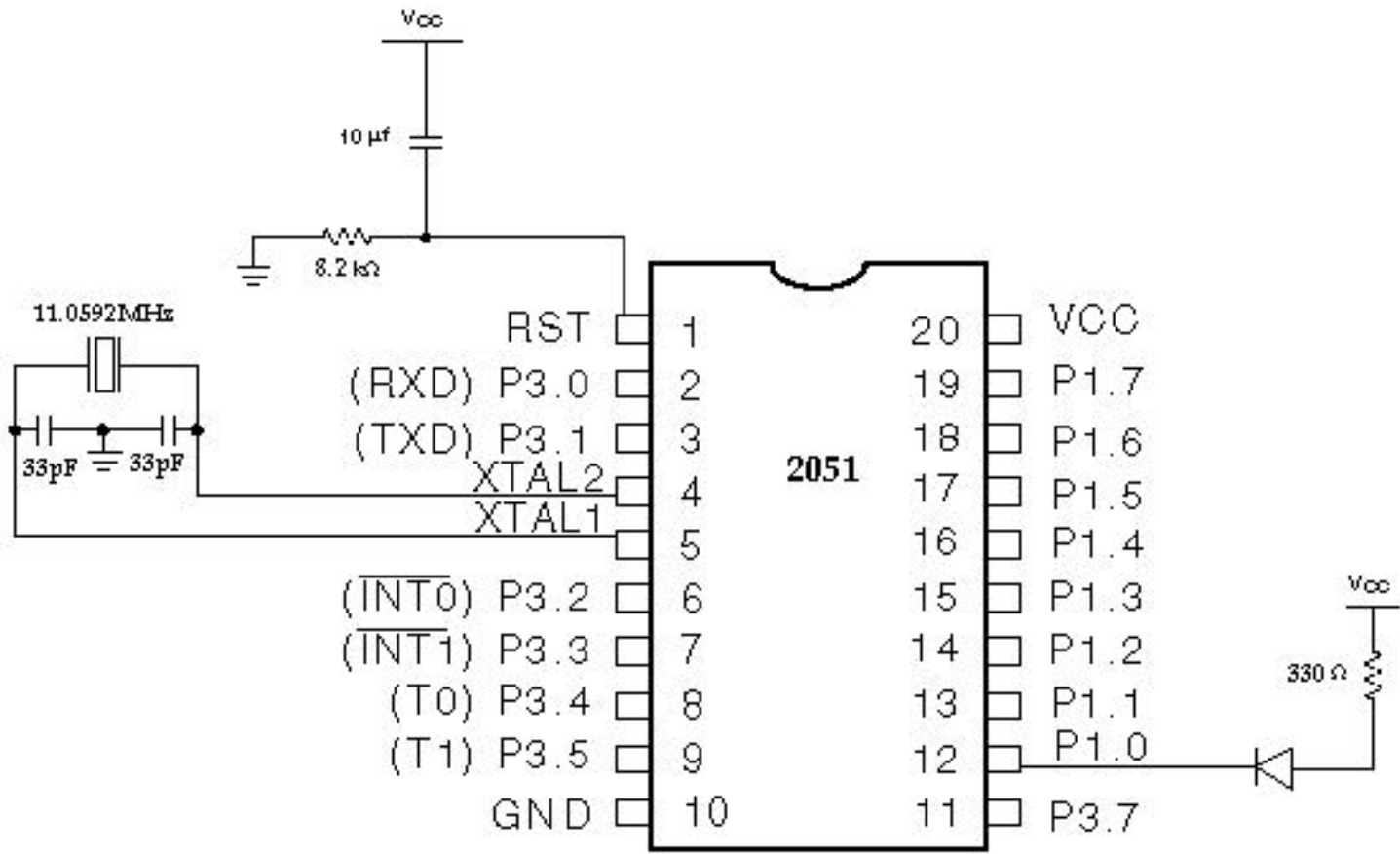
Pins 6, 7, 8, 9, 12 and 13 also have other possible uses that we will not get into here. You can learn more about them in the documents listed at the end of Section 2.3.1






2.3 Making an LED Blink

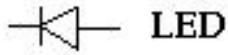
Step 1.) Building the circuit.

The first step is to build the circuit. At this point you should be familiar with the parts used. (2 resistors, 3 capacitors, 1 LED). The 2051 chip should be placed across the middle of the breadboard (like the 555 IC was) so each leg will have it's own row of holes. You will probably need to bend the legs in slightly so the chip will go in the holes. Do this by placing the chip sideways on a hard flat surface so 10 pins from one side of the chip are laying flat against the surface. Then bend all the chips at once by pressing gently on the chip.

This project will require you to remove the 2051 from the board to program it. Be careful when removing the 2051 from the board. Always use the chip extractor to pull the chip off the board. Try not to bend the pins (legs) when removing the chip.



 Ground	 Capacitor
 Resistor	
 LED	 Crystal (XTAL)



LED



Crystal (XTAL)

Vcc = 5V (from the 5 Volt power supply we built earlier)

The only thing we want to do with this project is to make the LED blink. By doing this, you will be able to learn the basic process of compiling a program and then downloading (programming) the resulting file into the microcontroller.

First we will assume we already have the program written. The file ledtest.asm is the program we are going to use. (It should be in the folder with the TASM software at c:\tasm) The program is printed out at the end of this section. We will take a closer look at the program after making the LED blink.

Step 2.) Compiling the Code

Start an MS-DOS window. (Look for MS-DOS Command Prompt under Programs in the Start Menu in Windows.) Change to the directory where the TASM files are (type `cd c:\tasm`), and compile the code using the command

```
tasm -51 ledtest.asm ledtest.hex
```

This will create a file called ledtest.hex.

Step 3.) Downloading the code to the Microcontroller.

Make sure the serial cable and the power supply are connected to the PG302 programmer. Put the 2051 into the PG302 programmer (look at Section 2.0.1 for a diagram of how to put the chip in the programmer).

Start the PG302 software.

From the Setup Menu, select the microcontroller you are using, the AT89C2051.

From the Setup Menu, select the Comm port (serial port) you are using. (It is ok to guess if you are not sure which one is which.)

Press PROGRAM DEVICE.

Press BROWSE.

Find ledtest.hex and click on it (single click). (This file should be at C:\tasm\ledtest.hex)

Press OK to select the file.

Press OK to program the file into the microcontroller. The software should say 'Device Equals File'. If it gives you a different message, check the troubleshooting tips at <http://www.iguanalabs.com/troublei.htm>. (If you had to guess about which Comm

port you are using, try choosing the other Comm ports).

Now the program should be loaded into the microcontroller.

Leave the power disconnected from the circuit you have built.

Move the microcontroller back to the circuit you have built.

Plug the power supply into the power supply adapter (which should be connected to the board). If the LED starts blinking, then you have successfully built your first microcontroller project.

The assembly language program is shown below. Everything in green is just comments (created by using a ; in front of the comment). The real commands are in blue. Black is used for numbers and also for names that we create. Red is also used for numbers. If the number actually starts with a number then it is red. Otherwise it will be black.

Numbers in the microcontroller are stored in registers. The register names show up in purple. In this program we use the register **A** (the main register), and the registers **R5**, **R6**, and **R7**. We can also use **R0**, **R1**, **R2**, **R3**, and **R4**.

Take a look at the program and see if it makes sense. The program starts with the first command **AJMP START** at the top which tells the 2051 to jump to the point labeled **START:** at the bottom. Next, the **ACALL INITIALIZE** command jumps back up to **INITIALIZE:**. Here we just move (**MOV**) the default number 0 into the control registers and then return (**RET**) to the next command after the **ACALL INITIALIZE** command. (You can move different values into those control registers to change how the 2051 operates, but don't worry about that yet.) The next command, **CPL P1.0**, turns the LED on or off (makes it the opposite of what it was). Then we go to the **DELAYHS** routine to make a delay before we turn the LED on or off again. (Without a delay, the LED would go on and off so fast that it would just look like it was staying on all the time.) Finally we use the **AJMP LOOP** command to jump back up to the point labeled **LOOP:** to turn the LED on or off. The microcontroller will keep going through this loop until you turn the power off.

The commands we use the most are shown in section 2.3.1. For a complete list of commands and descriptions, look at the file **Programmers Guide and Instruction Set.pdf** in the **MBKit** folder. See if you can follow the program through, using the green comments for guidance.

Start up the AY Pad software. Open the **ledtest.hex** file and try changing the command **MOVR6, #00H** (in the **DELAYHS** loop) to **MOVR6, #70H**. Save the file and then go back and do steps 2 and 3 to see how this changes the program.

```
*****
;* LED Blinker *
;* Iguana Labs *
;* 4/1/97 *
*****
#include "8051EQU.INC" ;include predefined constants
*****
; RESET ;reset routine
.ORG0H ;locate routine at 00H
AJMP START ;jump to START
*****
```

```

; INTERRUPTS (not used) ;place interrupt routines at appropriate
;memory locations
.ORG03H ;external interrupt 0
RETI
.ORG0BH ;timer 0 interrupt
RETI
.ORG13H ;external interrupt 1
RETI
.ORG1BH ;timer 1 interrupt
RETI
.ORG23H ;serial port interrupt
RETI
.ORG25H ;locate beginning of rest of program
;*****
INITIALIZE: ;set up control registers
MOVTCON,#00H
MOVTMOD,#00H
MOVPSW,#00H
MOV IE,#00H ;disable interrupts
RET
;*****
; Real code starts below. The first two routines are for delays so we
; can slow down the blinking so we can see it. (Without a delay, it
; would blink so fast it would look like it was always on.
;*****
DELAYMS: ;millisecond delay routine
MOVR7,#00H ;put value of 0 in register R7
LOOPA:
INCR7 ;increase R7 by one (R7 = R7 +1)
MOVA,R7 ;move value in R7 to Accumlator (also known as A)
CJNEA,#0FFH,LOOPA ;compare A to FF hex (256). If not equal go to LOOPA
RET ;return to the point that this routine was called from
;*****
DELAYHS: ;half second delay above millisecond delay
MOVR6,#00H ;put 0 in register R6 (R6 = 0)
MOV R5,#002H ;put 2 in register R5 (R5 = 2)
LOOPB:
INC R6 ;increase R6 by one (R6 = R6 +1)
ACALL DELAYMS ;call the routine above. It will run and return to here.
MOVA,R6 ;move value in R6 to A
JNZ LOOPB ;if A is not 0, go to LOOPB
DEC R5 ;decrease R5 by one. (R5 = R5 -1)
MOVA,R5 ;move value in R5 to A
JNZ LOOPB ;if A is not 0 then go to LOOPB.
RET
;*****

```

START: ;main program (on power up, program starts at this point)
 ACALL INITIALIZE ;set up control registers
 LOOP:
 CPLP1.0 ;ComPLement (invert) P1.0 (this makes LED change)
 ACALL DELAYHS ;go to above routine that causes a delay
 AJMP LOOP ;go to LOOP(always jump back to point labeled LOOP)
 .END ;end program (never reaches this point)

2.3.1 Partial Instruction Set for 2051 Microcontroller

Command	Description
ADD	Add Two Numbers
SUBB	Subtract Two Numbers
INC	Increment Data (Add 1 to existing amount)
DEC	Decrement Data (Subtract 1 from existing amount)
MUL	Multiply Two Numbers
DIV	Divide Two Numbers
ANL	Logical AND Two Numbers
ORL	Logical OR Two Numbers
XRL	Logical XOR Two Numbers
CLR	Clear (reset) data to 0
CPL	Complement (Flip) Data (1s go to 0 and 0s go to 1)
MOV	Move data around in the chip from register to register
SETB	Set a number to 1
JB	Jump to a new location if a number is 1
JNB	Jump to a new location if a number is 0

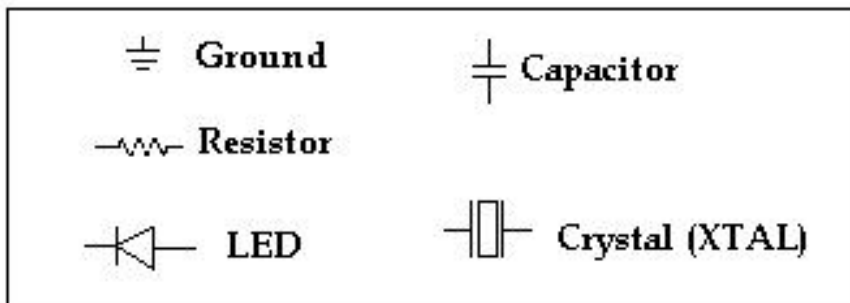
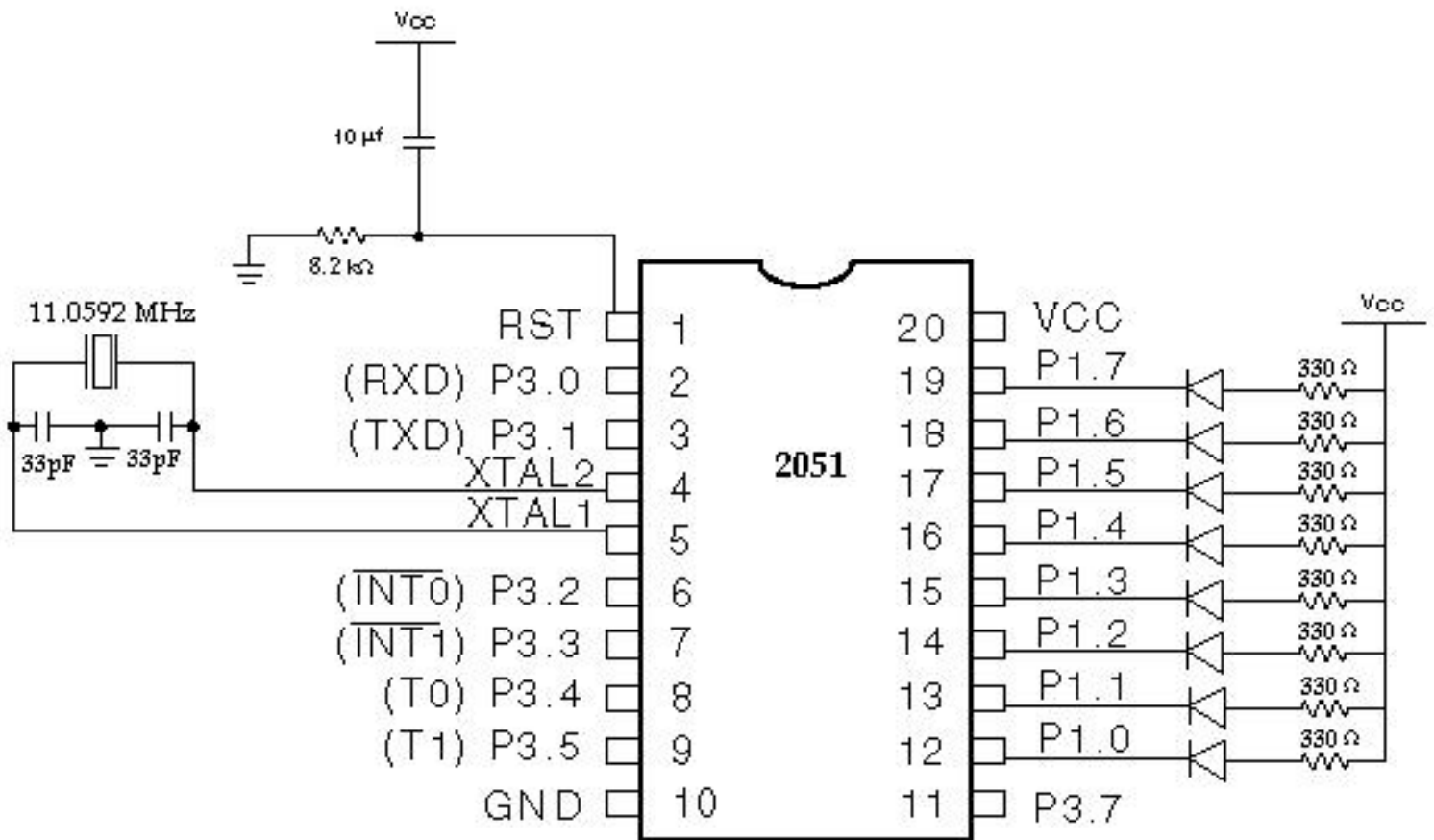
ACALL	Jump to a subroutine, then return to this point when done with subroutine
RET	Return from subroutine (used at end of subroutine)
AJMP	Jump to a new location
JZ	Jump if a number is 0
JNZ	Jump if a number is not 0
CJNE	Compare two numbers and jump if they are not equal
DJNZ	Decrease a number by 1 and jump if result is not zero
NOP	No Operation (used to waste time when waiting for something to happen)

For a complete list of commands and descriptions, look at the file `Programmers Guide and Instruction Set.pdf`. You may also want to look at the file `8051mem.pdf` for more information on the memory structure of the 8051 (for storing data). The file `8051arch.pdf` has more information on the chip architecture. The file `8051hardware.pdf` has more information on the hardware (registers, port structure, etc.). The above files have general information for the Atmel family of 8051 microcontrollers. For information specific to the 2051, look at the file `AT89C2051.pdf`.

2.4 A Simple Microcontroller System

Next we will add more LEDs and make a slightly more complex program.

Add in the LEDs and the resistors as shown in the diagram below.



Vcc = 5V

In this project the program is similar to the previous one but instead of just making one LED blink, we are using all of Port 1. The program for this is ledproj2.asm. (Look in the TASM folder) Once you have compiled the program, download it to the 2051. Make sure the power is off for the circuit you have built. Then move the microcontroller to the circuit.

Turn on the power to the circuit. The LEDs should all come on. Then they should start blinking on and off as the counter counts up from 0 to 255 (all on to all off). The numbers are in digital (binary) format. For more information on working with digital numbers, look at http://webster.cs.ucr.edu/Page_asm/ArtofAssembly/CH01/CH01-1.html

Use AY Pad to look at this program and see how it is different from ledtest.hex.

[Click here to see the next tutorial on using the 2051 to make sound with a speaker.](#)

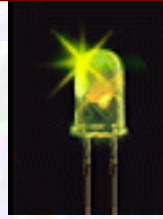
[Click here to see the tutorial on using the 2051 with a 7 segment display.](#)

[Click here to see the tutorial on using a switch as an input to the microcontroller.](#)

[Click here to see the tutorial on using Light Sensors with the microcontroller.](#)

[**To see the sales ad for the Microcontroller Beginner Kit, click here.**](#)

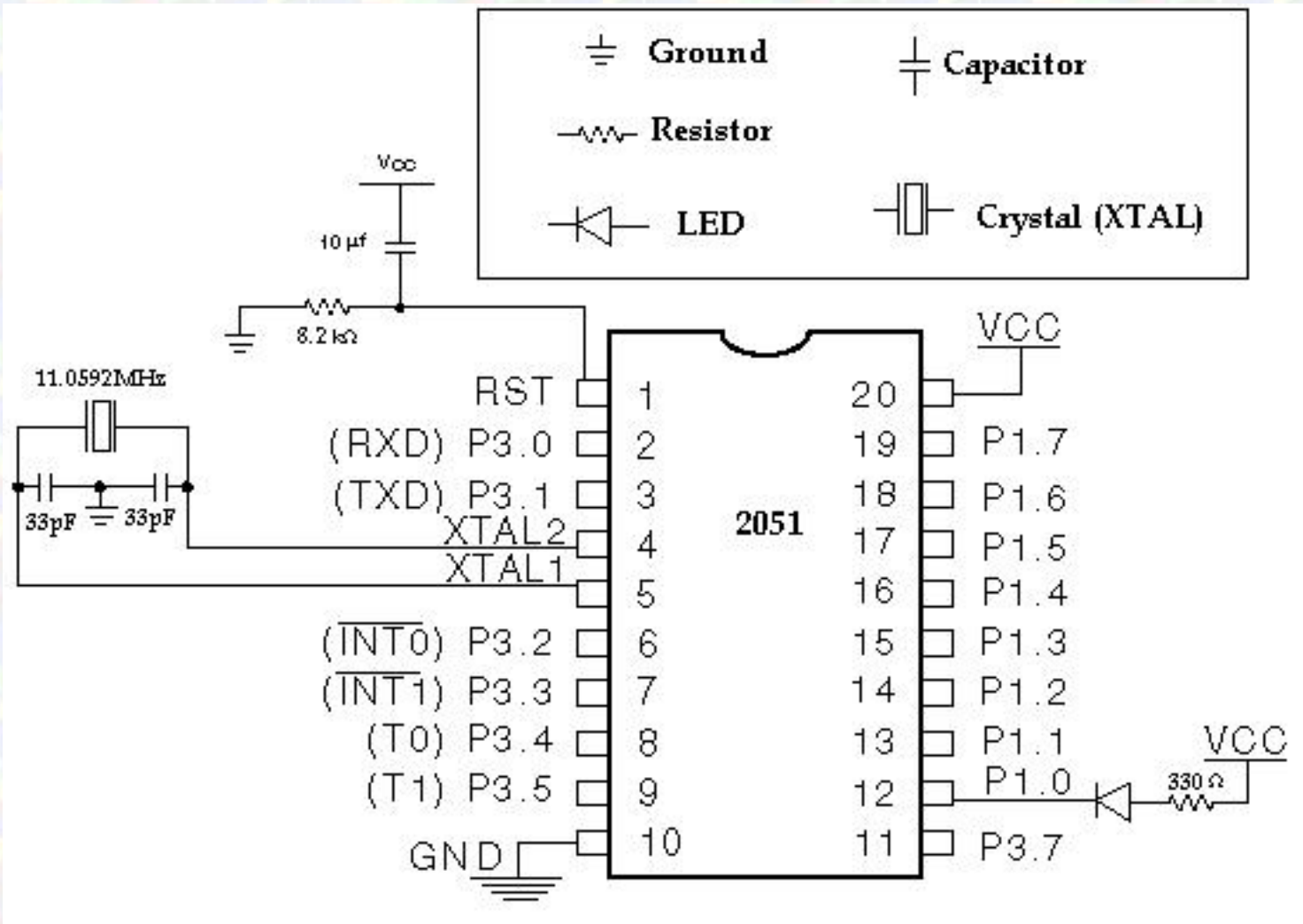
[**For pricing, ordering information, other kits and products, click here or call 800-297-1633.**](#)



Making an LED Blink - 2051

Step 1.) The first step is to build the circuit. At this point you should be familiar with the parts used. (2 resistors, 3 capacitors, 1 LED). You can either put these parts together using a breadboard or wirewrap. This design is intended for use with an Atmel 2051 microcontroller (a 20 pin version of the 8051). Most microcontrollers (such as a normal 8051 or 8751) can not handle the current required to turn an LED on and off but the ATMEL part has this capability.

$V_{cc} = 5V$ and $Gnd = 0V$



The only thing we want to do with this project is to make the LED blink. By doing this, you will be able to learn the basic process of compiling a program written in assembly language and then programming the resulting file into the microcontroller.

First we will assume we already have the assembly code written. [ledtest.asm](#) is the assembly language program we are going to use. (This file is included with [TASM](#).)

Step 2.) Compiling the Code

Move the assembly language program ([ledtest.asm](#)) to the directory where you have [TASM](#). Bring up a DOS prompt, change to the directory where the TASM files are, and compile the code using the command

```
tasm -51 ledtest.asm ledtest.hex
```

This will create a file called ledtest.hex.

Close the DOS prompt window now.

Step 3.) Downloading the code to the Microcontroller.

Make sure the serial cable and the power supply are connected to the [PG302 programmer](#).

Put your microcontroller into the PG302 programmer.

Run [PG302](#).

From the Setup Menu, select the type of device (microcontroller) you are using

From the Setup Menu, select the Comm port you are using.

Press PROGRAM DEVICE.

Press BROWSE.

Find ledtest.hex and click on it (single click).

Press OK to select the file.

Press OK to program the file into the microcontroller.

Now the program should be loaded into the microcontroller.

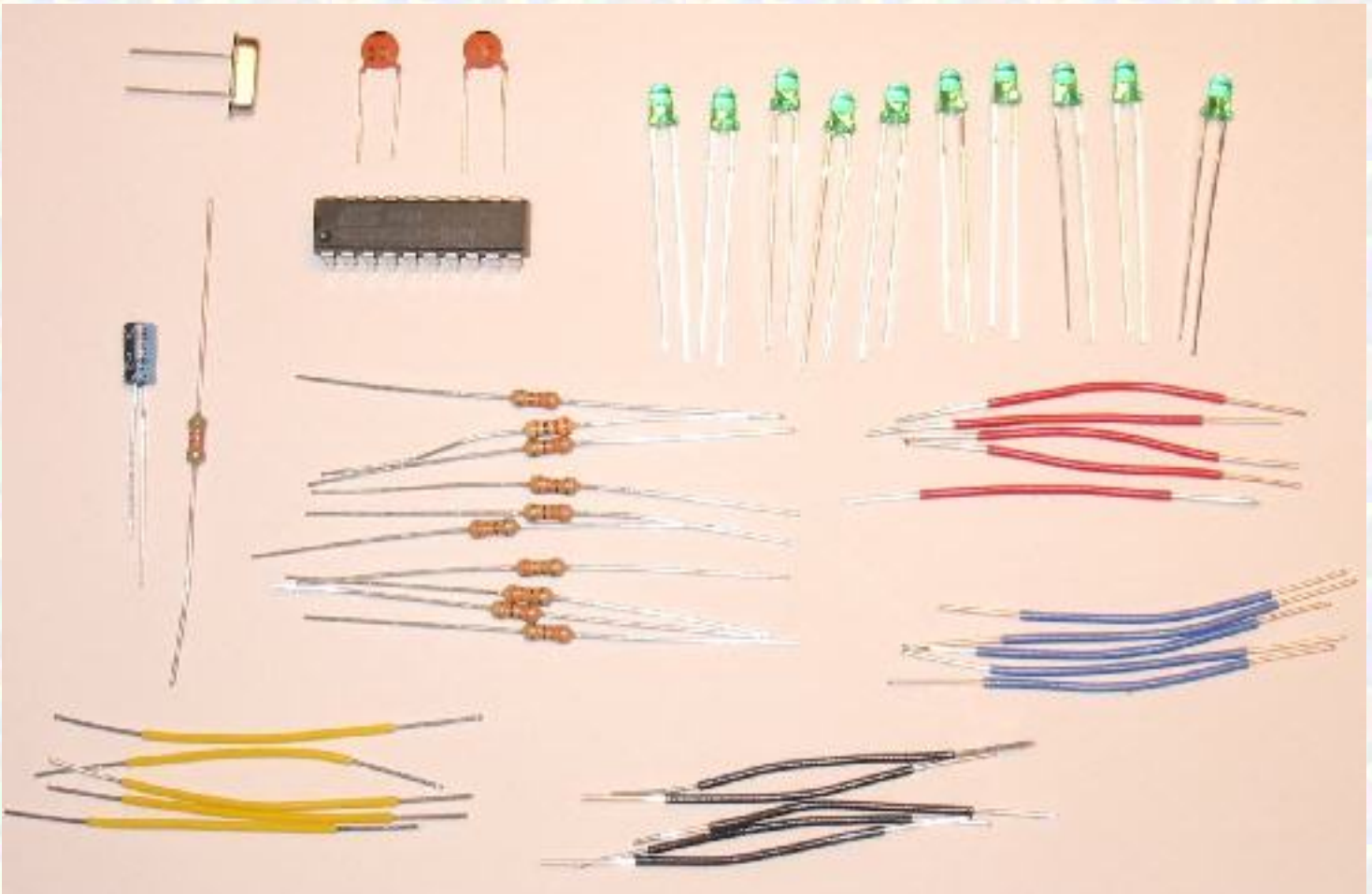
Make sure the power is off for the circuit you have built.

Move the microcontroller back to the circuit you have built.

Turn on the power to the circuit. If the LED starts blinking, then you have successfully built your first microcontroller project.

The parts for this kit are included in 2051Kit and the [Microcontroller Beginner Kit](#). See [part 2](#) for parts list and see [Order Form](#) for price and ordering information or call 800-297-1633.

You may also be interested in the ExpBoard. It is a fully assembled board made for experimenting with microcontrollers. [Click here for more information on the ExpBoard.](#)



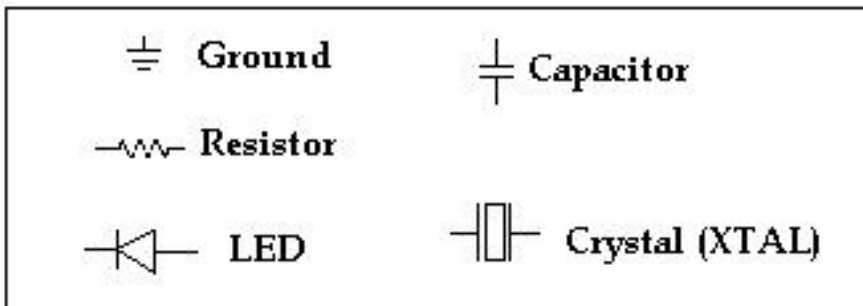
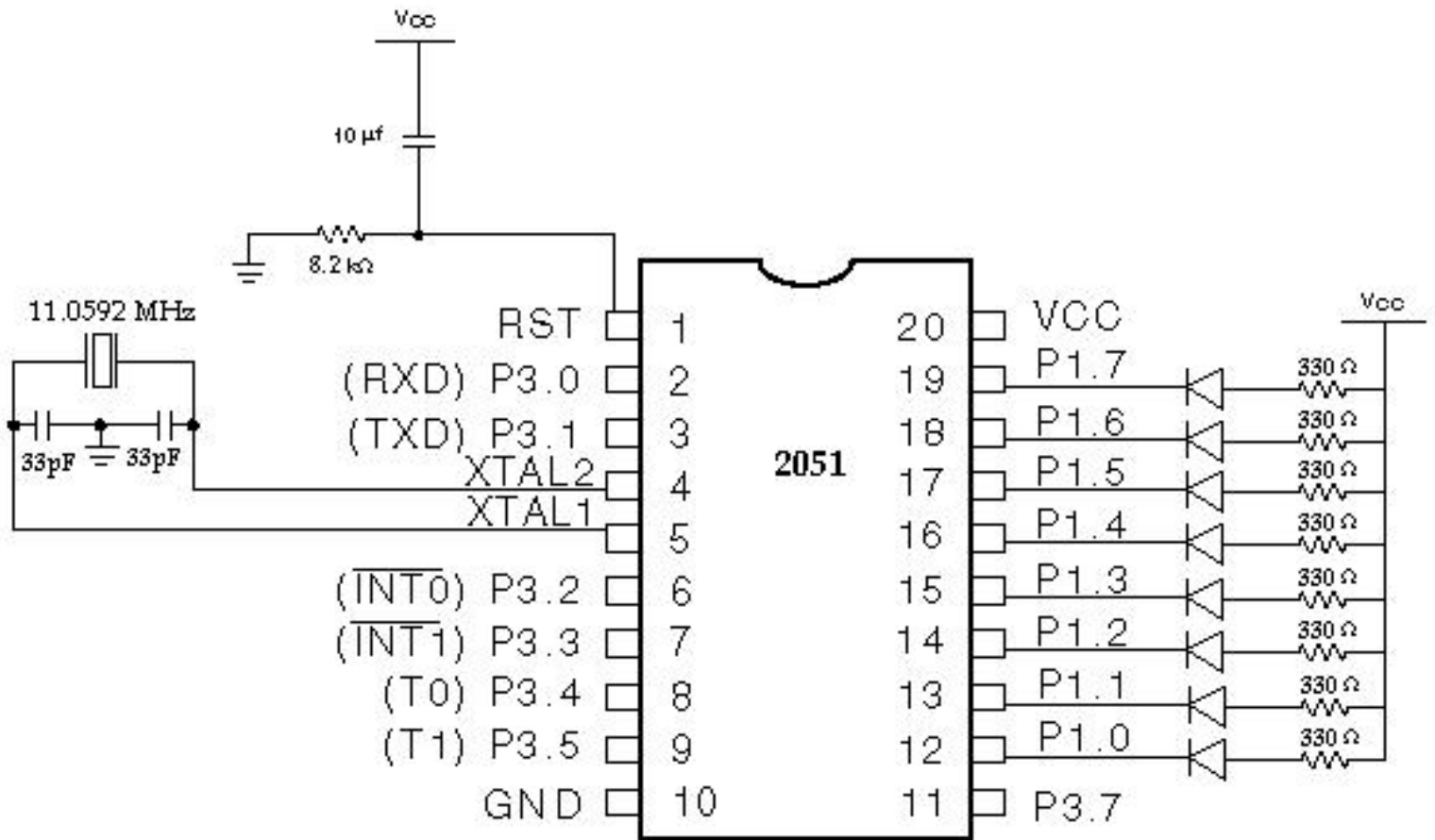
Send Questions to support@iguanalabs.com

Iguana Labs

A Simple Embedded Microcontroller System

Step 1.) The first step is to build the circuit. At this point you should be familiar with the parts used. You can either put these parts together using a breadboard or wirewrap. This design is intended for use with an Atmel 2051. Most microcontrollers (such as a normal 8051 or 8751) can not handle the current required to turn an LED on and off but the ATMEL part has this capability.

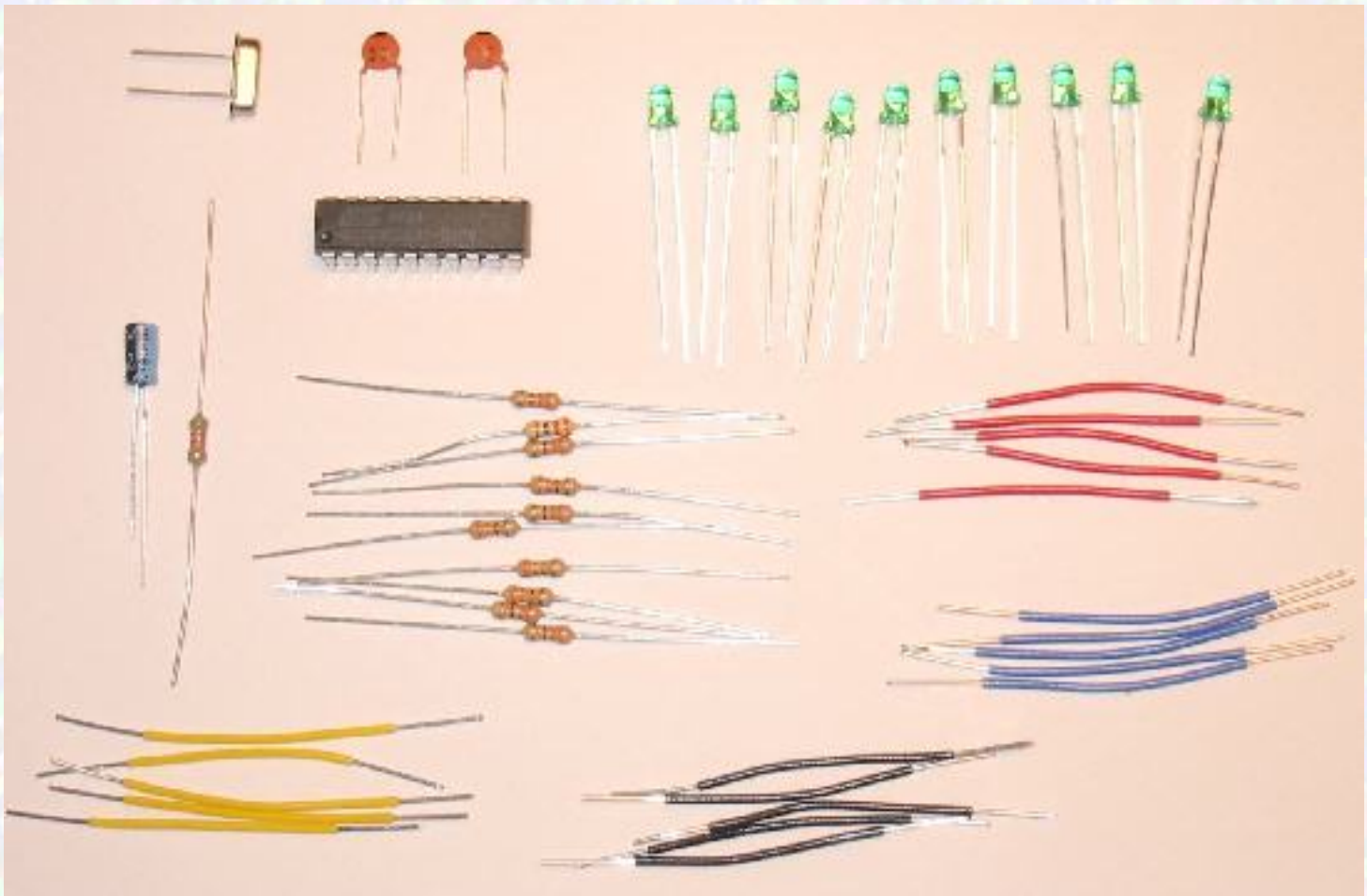
$$V_{cc} = 5V \text{ and } Gnd = 0V$$



The basic process of compiling an assembly language program and loading it into the microcontroller was covered in the [first microcontroller project](#). In this project the program is similar. Instead of just making an LED blink, we are using all of Port 1 on the 2051 to make an 8 bit counter. The program for this is [ledproj2.asm](#). Once you have compiled the program, download it to the 2051. Make sure the power is off for the circuit you have built. Then move the microcontroller to the circuit.

Turn on the power to the circuit. The LEDs should all come on. Then they should start blinking on and off as the counter counts up from 0 to 255 (all on to all off).

You can order the parts for this project. It includes:



1 -

AT89C2051-24PC Microcontroller

1 - 11.0592 MHz Crystal

2 - 33pF Capacitors

1 - 10 uF Capacitor

1 - 8.2k Resistor

10 - 330 ohm Resistors

10 - LEDs

Jumper Wires

For Price and Ordering Information, look at 2051Kit on [Order Form](#). If you want a printed copy of this and the other tutorials, remember to mark that on the order

form.

Send Questions to support@iguanalabs.com

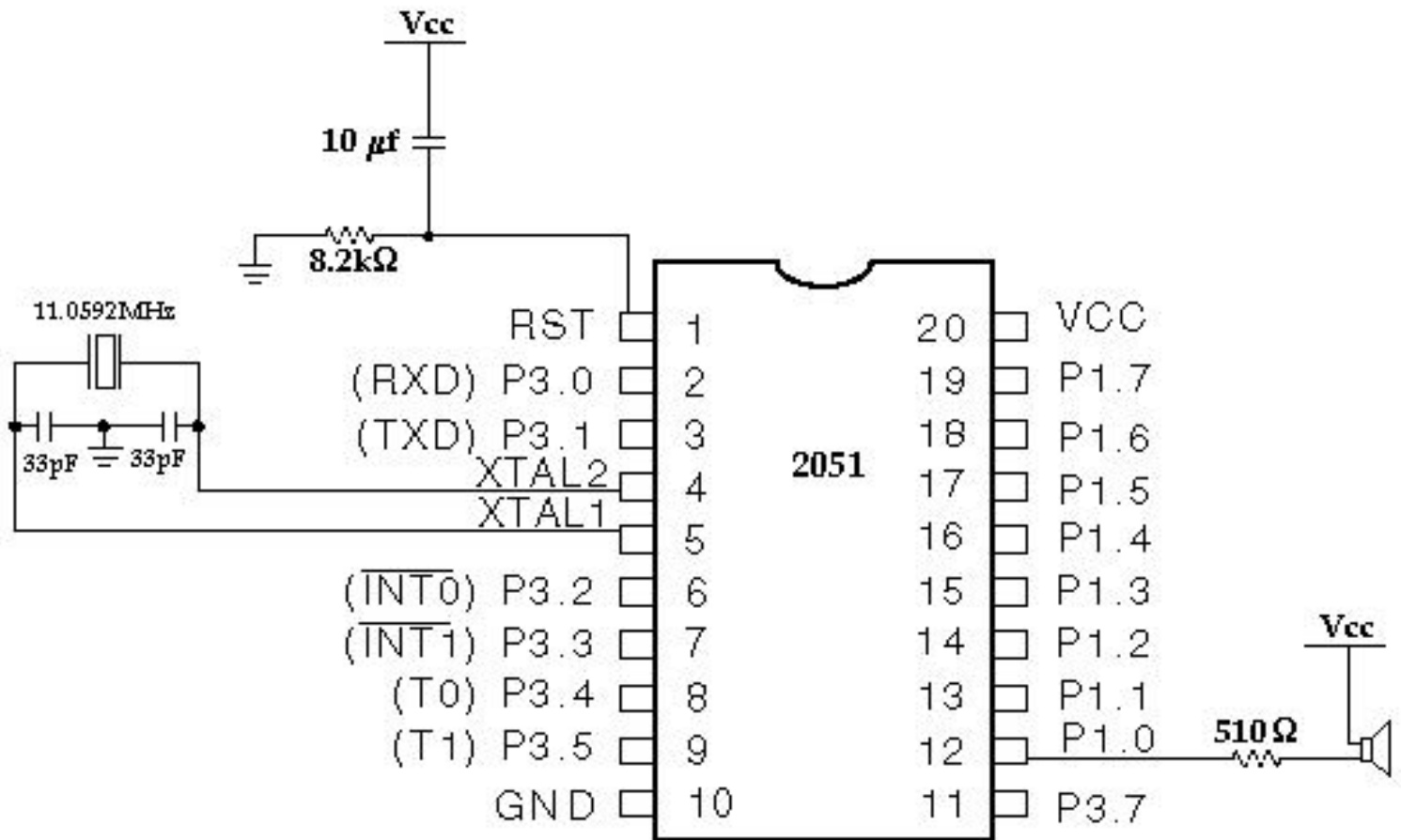
Iguana Labs

This page last updated on June 27, 2002.

Making sound with the 2051

The first step is to build the circuit. This design is intended for use with an Atmel 2051 programmable microcontroller (a 20 pin version of the 8051 microcontroller). An 8051 or 8052 can be substituted for the 2051 if the connections are moved to the appropriate pins on those chips.

Vcc = 5V and Gnd = 0V



⊥ Ground

⊥ Capacitor

⊞ Resistor

⊡ LED

⊡ Crystal (XTAL)



LED



Crystal (XTAL)

The basic process of compiling a program written in assembly language and then programming the resulting file into the microcontroller was covered in the [first microcontroller tutorial](#). This project uses the same code as the first program. To start, we just make a minor change to the original assembly language program, ledtest.asm. Replace the line at the bottom that says ACALL DELAYHS with ACALL DELAYMS. This changes the delay from half a second to one millisecond. Save it as a new file called [sounds.asm](#). That change makes it so that rather than delaying a half a second, we are only delaying one millisecond. Compile the sounds.asm file, load sounds.hex into the 2051 and put the chip back in the circuit. Connect the power. You should hear a tone coming from the speaker. If the LED was hooked up, it would be going on and off so fast that it would look like it was on all the time. But with the speaker, we can hear the voltage going on and off. You can only see the LED going on and off up to about 25 blinks per second. You can start to hear the speaker making noise at about 50 blinks per second. But instead of calling it blinks, we call it cycles. Right now the microcontroller is putting out about 500 cycles per second. This is also called the frequency. It is a frequency of 500 Hertz (where Hertz means cycles per second). If you want to make the frequency go down, you can add more delay. Try adding another millisecond delay by inserting another ACALL DELAYMS (see [sounds2.asm](#)). This will make the frequency 250 Hertz. You can hear that the sound is lower in frequency.

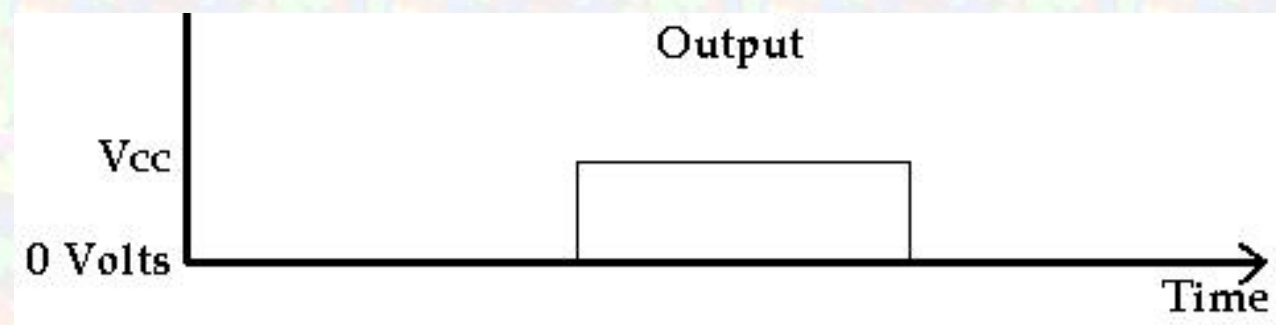
The information below is fairly advanced so don't worry if you want to skip it.

[Click here for the next tutorial - Using a 7 Segment Display with the 2051](#)

[Click here for more sound examples.](#)

How to get exactly the frequency you want:

To figure out how to get an exact frequency, you have to carefully look at how long it takes for the microcontroller to switch the output from 0 volts to 5 volts and back to 0 volts again. This is one cycle.



Lets start by determining what frequency the sounds.asm program is creating. First lets look at the DELAYMS routine. We need to figure out how long this routine lasts. By just looking at the code, we can figure out that it goes through a loop 255 times. Each time through the loop it does 3 commands. The first two commands each take 12 clock cycles and the 3rd command (CJNE) takes 24 clock cycles. (You can find this out by looking for information on the commands in [2051arch.pdf](#)). So each loop takes 48 clock

cycles. To translate this into time, we need to look at the clock speed of the microcontroller. We are using an 11.0592 MHz crystal. This means the clock is running at a frequency of 11,059,200 cycles per second. (MHz is MegaHertz which is million cycles per second). Each cycle takes $1/11,059,200$ seconds = 0.00000009 seconds. So each loop takes $48 * 0.00000009 = 0.00000434$ seconds. And 255 loops takes $255 * 0.00000434 = 0.001106771$ seconds which is slightly longer than 1 millisecond (1 millisecond = 0.001 seconds). If we wanted to get closer to exactly 1 ms we could change the loop so that it only repeats 230 times rather than 255.

So, the program makes the output go from 0 to 5 volts, then waits 1 ms, then goes from 5 volts to 0 volts, then waits 1 ms and that makes one cycle. So one cycle takes about 2 ms (This is called the period). To convert that to frequency, divide 1 cycle by 2 ms ($1/0.002 = 500$). Then you get 500 cycles per second. Or, to be exact, using the numbers above, one cycle takes 2.213542 ms for a frequency of 451.76 Hz.

So, to get an exact frequency, you can start with the frequency and work backwards. Say you want to make 440 Hz, which is a musical A note. To find the period, divide 1 by 440. This gives you the period equal to 0.002272727 seconds. Then divide this by 2 to find out how long each delay must be (there will be 2 delays per cycle). Each delay should be 0.001136364. Then find out how many microcontroller clock cycles this is by dividing by 0.00000009. This equals 12626 cycles (after dropping the decimal part). Using our loop that takes 48 cycles, this would be about 263 loops ($12626 / 48 = 263$). We can only go up to 255 loops so then we can either make our loop take more time, or add in an extra DELAY routine that adds in the extra 8 loops. The easiest solution is to make our loop longer. We can add in an extra 12 cycles per loop by putting in a NOP (no operation) command. Then each loop is 60 cycles and we need about 210 loops ($12626 / 60 = 210$). The resulting code is shown in [sound440.asm](#). That will not be exactly 440 Hz because we had to round off in some places (you can't do 210.43 loops but 210 is close. To figure out exactly what frequency that we made, we can do the same as we did above with the DELAYMS routine. Each loop is $0.00000009 * 60 = 0.0000054$ seconds. Each DELAYMS takes $210 * 0.0000054 = 0.001134$ seconds. With 2 delays per cycle this is a period of 0.002268. In terms of frequency, $1/0.002268 = 440.9$ cycles per second which is close to 440.

Note: To really be exactly right on the frequency you are making, you need to include the time in each cycle for the other commands, CPL, ACALL and RET, and the commands in the DELAYMS loop, MOV and RET. These add an extra 96 clock cycles each time through. Since it takes 2 times through to make a cycle on the output, That is an extra 192 cycles. This equals $192 * 0.00000009 = 0.00001728$ seconds. So the period is actually $0.002268 + 0.00001728 = 0.00228528$ and the frequency is actually 437.6. So this extra time must be considered if you are trying to get a very precise frequency.

Note: You are limited in how close you can get to an exact frequency by the microcontroller clock speed. The faster the clock is, the more accurate you can be. For example, with a 11.0592 MHz clock where each cycle is 0.00000009 seconds, the closest you can get to 440 Hz is 440.0788621 Hz. This is found by $1/440 = 0.002272727$ seconds and $0.002272727 / 0.00000009 = 25253$ cycles (must round to closest whole number because you can't have part of a cycle). Since the shortest commands take 12 clock cycles, then you won't be able to write a routine that takes exactly 25253 cycles. It has to be some multiple of 12. The closest multiple of 12 is 25248. Then $25248 * 0.00000009 = 0.00227232$ seconds and $1 / 0.00227232 =$

440.0788621 Hz. If you have a faster microcontroller clock speed you can be more accurate. For example, with a 24 MHz clock (The fastest you can use with a 2051 microcontroller) then you can get 440.0440044 Hz. Also, if you use a clock that gives you a different period you may be able to get exactly 440 Hz. For example, if you have a microcontroller clock that is 22,440,002.69 MHz then you can get much closer to 440 Hz, but you have to find a crystal that runs at that exact speed, and there probably is not one.

The parts for this kit are included in Microcontroller Beginner Kit. See the [MB Kit page](#) for parts list and see [Order Form](#) for price and ordering information.

Send Questions to tech@iguanalabs.com

[Click here to go to the Main Tutorial Page](#)

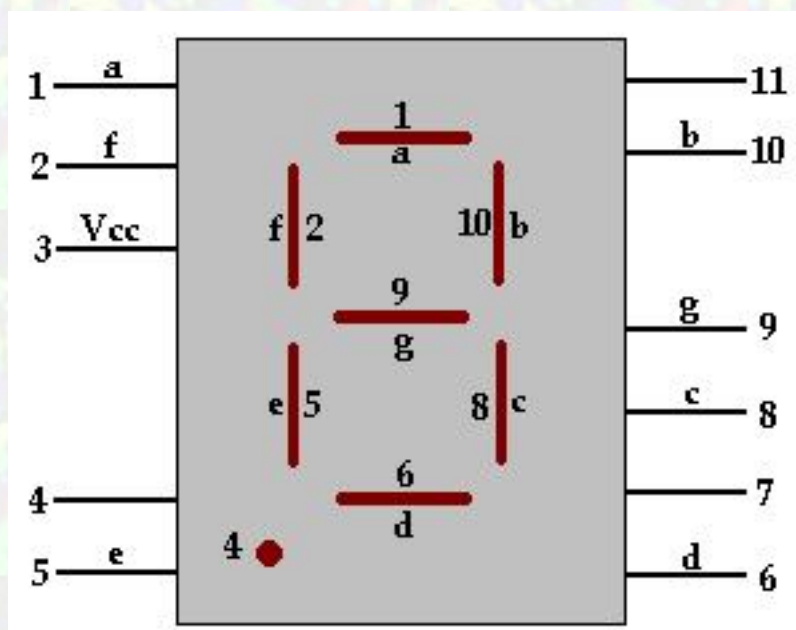
[Click here for more sound examples.](#)

Iguana Labs

This page last updated on February 8, 2002.

The 7 Segment Display

The 7 segment display is found in many displays such as microwaves or fancy toaster ovens and occasionally in non cooking devices. It is just 7 LEDs that have been combined into one case to make a convenient device for displaying numbers and some letters. The display is shown on the left. The pinout of the display is on the right.



This version is a common anode version. That means that the positive leg of each LED is connected to a common point which is pin 3 in this case. Each LED has a negative leg that is connected to one of the pins of the device. To make it work you need to connect pin 3 to 5 volts. Then to make each segment light up, connect the ground pin for that led to ground. A resistor is required to limit the current. Rather than using a resistor from each LED to ground, you can just use one resistor from Vcc to pin 3 to limit the current.

The following table shows how to form the numbers 0 to 9 and the letters A, b, C, d, E, and F. '0' means that pin is connected to ground. '1' means that pin is connected to Vcc.

	a (Pin 1)	b (Pin 10)	c (Pin 8)	d (Pin 6)	e (Pin 5)	f (Pin 2)	g (Pin 9)
0	0	0	0	0	0	0	1
1	1	0	0	1	1	1	1
2	0	0	1	0	0	1	0
3	0	0	0	0	1	1	0
4	1	0	0	1	1	0	0
5	0	1	0	0	1	0	0
6	0	1	0	0	0	0	0
7	0	0	0	1	1	1	1
8	0	0	0	0	0	0	0
9	0	0	0	1	1	0	0
A	0	0	0	1	0	0	0
b	1	1	0	0	0	0	0
C	0	1	1	0	0	0	1
d	1	0	0	0	0	1	0
E	0	1	1	0	0	0	0
F	0	1	1	1	0	0	0

Now, we want to run the display with the 2051 microcontroller. We will use Port 1 to run the display. Use the same configuration as in the [first 2051 tutorial](#). Connect the 2051 to the 7 segment display as follows.

- 2051 pin 12 to display pin 9 (P1.0 will control segment g)
- 2051 pin 13 to display pin 2 (P1.1 will control segment f)
- 2051 pin 14 to display pin 5 (P1.2 will control segment e)
- 2051 pin 15 to display pin 6 (P1.3 will control segment d)
- 2051 pin 16 to display pin 8 (P1.4 will control segment c)
- 2051 pin 17 to display pin 10 (P1.5 will control segment b)
- 2051 pin 18 to display pin 1 (P1.6 will control segment a)

Connect the display pin 3 to 5 volts using a 100 ohm resistor.

Here is a program that cycles through the numbers and letters above. [7seg.asm](#)

Compile the program using [TASM](#) and load the hex file into the 2051. Put the 2051 into the circuit and connect the power.

The parts for this project are included in the [Microcontroller Beginner Kit](#). See the [ordering information page](#) for pricing.

[Back To Tutorials Menu](#)

[Previous - Making Sound with the 2051 Microcontroller](#)

[Next - Using a Switch as an Input to the Microcontroller](#)

Send Questions to support@iguanalabs.com

Iguana Labs

This page last updated on June 25, 2002.

Using a Switch as an Input to a Microcontroller

Up until now, all the microcontroller projects have just done what they were programmed to do without any input from us. You have probably already thought that it would be nice to have the microcontroller do something based on inputs it receives from the real world. This tutorial shows how to use a switch as an input so that you can control what the microcontroller is doing.

In this tutorial, we will use the 7 segment display as we connected it in the previous tutorial ([Using a 7 segment display](#)). The switch will be used to change what is shown on the display.

The type of switch we will be using is called a normally open switch. This is often abbreviated as NO switch where NO stands for Normally Open. The switch has 2 legs (connectors). Normally Open means that the connectors are not connected (forming an open circuit) unless you press the switch. When you press the switch, a connection is made between the two connectors, creating a short circuit (or closed circuit). You can also get Normally Closed switches (NC) which means that they are closed, forming a short circuit, unless you are pressing the button.

Our switch is really a button rather than a switch because you have to hold it down to maintain the connection between the two points. These types are more useful with microcontrollers than a true switch. A true switch lets you flip between open and closed without having to hold down the button.

Inputs to the Microcontroller

To use one of the input/output pins of the 8051 microcontroller as an input, you must write a '1' to that pin.

Then in the software, you can check to see if the input is a '0' or a '1' and do different operations based on that input.

Use the same hardware configuration that you have after the 7 segment display tutorial. Add in the switch by connecting one wire to Pin 2 of the 2051 (that is Port 3, Bit 0) and connecting the other wire to ground. Now when the button is not pressed, pin 2 will be left unconnected to anything and when the software checks (reads) pin 2, it will see the '1' that we wrote there to make it an input pin. When the button is pressed, pin 2 will be connected to ground and when the software checks pin 2, it will see a '0'.

Software

Our first program will constantly check the input to see if the input (P3.0) is a '0' or a '1'. This is called polling.

We will use the command JB (Jump if Bit is set) to check if pin 2 is a '1' or a '0'. (If pin 2 is '1' then we say that the bit is set. In this case the bit is P3.0 so we would say P3.0 is set.) Below is the little bit of assembly code we will use to monitor the input and change the display. It is colorized as it is in the [AY Pad software](#).

```
SETB P3.0           ; This is required to use P3.0 as an input
LOOP:
JB P3.0, NOT_PRESSED ; If the button is not pressed, skip the next line
ACALL DISPLAY_0      ; Display '0' on the 7 segment display
AJMP LOOP            ; Jump back up to LOOP:
NOT_PRESSED:
ACALL DISPLAY_1      ; Display '1' on the 7 segment display
AJMP LOOP            ; go to LOOP(always jump back to point labeled LOOP)
```

The first line sets up pin 2 as an input by writing a '1' to P3.0. Then "JB P3.0, NOT_PRESSED" checks to see if P3.0 is a '1' which means the button is not pressed. If the button is not pressed, it jumps to that label (NOT_PRESSED) and calls the routine to make a '1' on the display. If the button is pressed, it will not jump but instead will go to the next line which calls the routine to make a '0' on the display. So the display will show the value of P3.0.

Note: Another command you may want to use is JNB (Jump if Not Bit set) which is the opposite of JB.

The program is [switch.asm](#).

Compile the program using [TASM](#) and load the hex file into the 2051. Put the 2051 into the circuit and connect the power. The display should say '1' until you press the button and then should say '0' while the button is being pressed.

Send Questions to support@iguanalabs.com

[Back To Tutorials Page](#)

[Previous - Using a 7 Segment Display](#)

[Next - Using a Light Sensor](#)

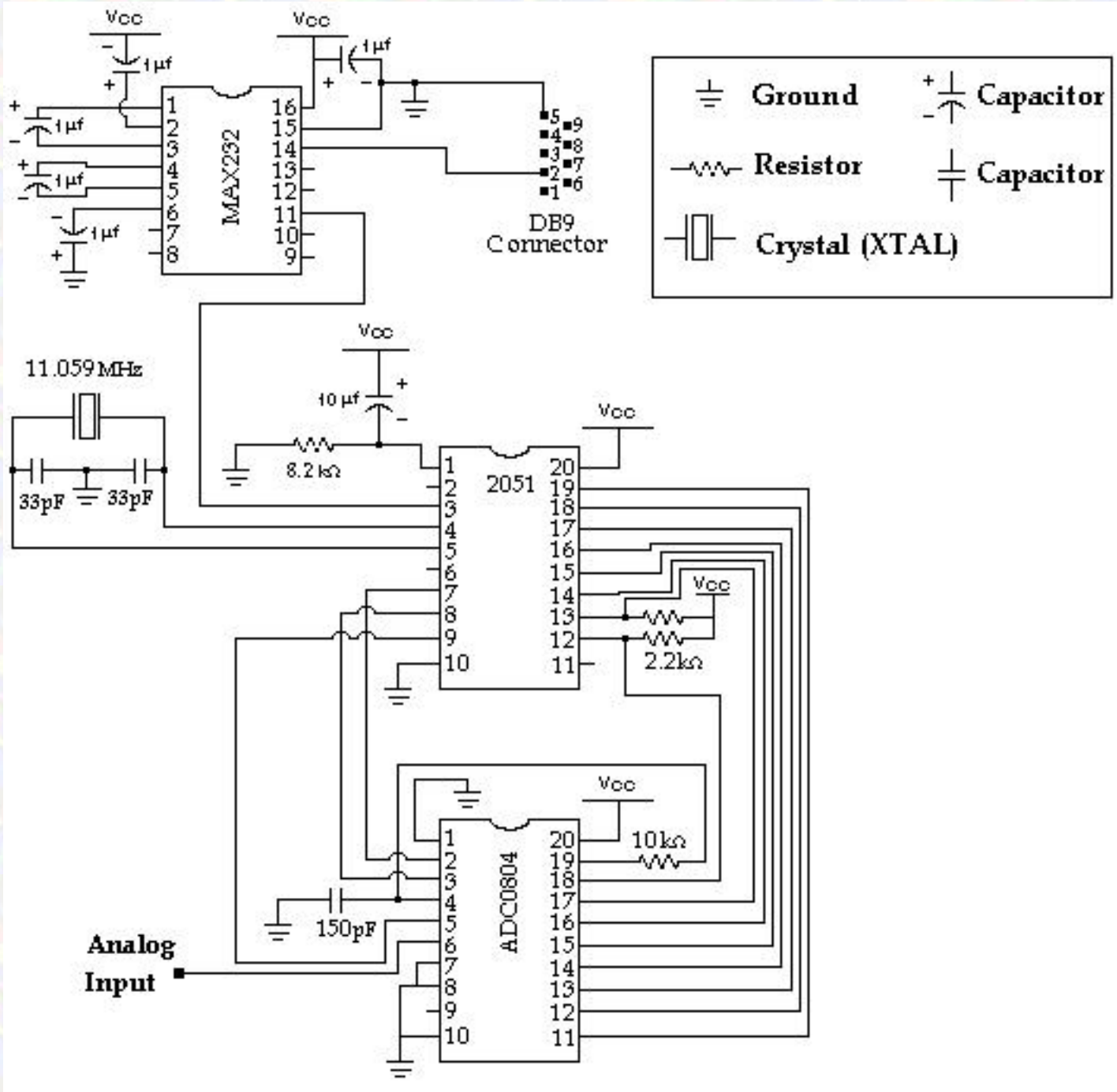
Iguana Labs

This page last updated on March 26, 2002.

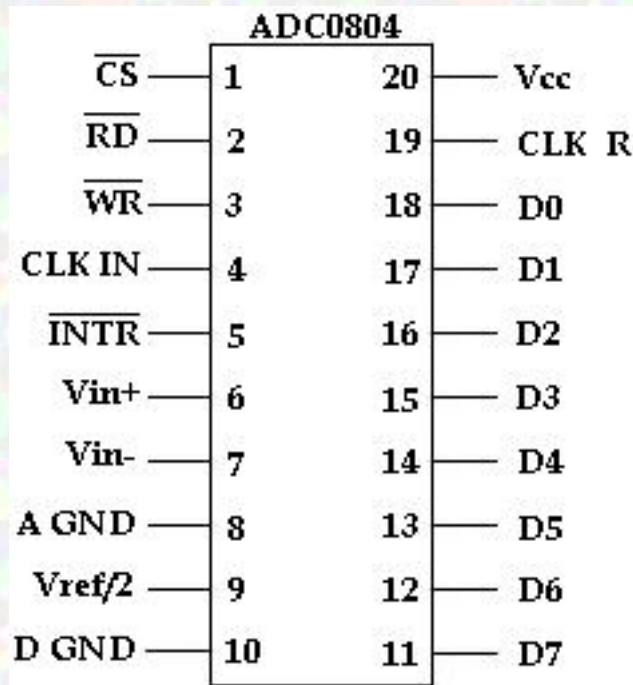
Data Collection - Analog to Digital Conversion and Communicating with a PC

This tutorial shows how to set up a system for collecting data through a computers serial port. It uses an ADC0804 chip to convert from analog to digital, an [AT89C2051](#) microcontroller to control the ADC0804 and send data to the PC, and a MAX232CPE chip to convert the signals from and to RS232 levels for sending and receiving from the PC.

Refer to the diagram below as you go through the individual steps in building the circuit.



Step 1.) Analog to Digital Conversion - The ADC0804 IC



The easiest way to do analog to digital conversion is to use an IC such as the ADC0804 that does the work for you. The analog voltage is applied to pin 6 and the result is available at pins 11 through 18. **We will connect pin 1 (Chip Select) to ground so that the chip is always enabled.** (If you wanted to use more than one ADC you could use this pin to control which chip is currently enabled).

Connect pin 7 (Vin -) to ground.

The ADC0804 includes an internal oscillator which requires an external capacitor and resistor to operate. **Connect the 150 pF capacitor from pin 4 to ground and the 10k ohm resistor from pin 4 to pin 19.**

Also for power,

Connect pin 20 to 5 volts.

Connect Pin 8 to ground.

Connect pin 10 to ground.

[For more information on the ADC0804 click here to get the data sheet for it.](#)

Step 2.) Interfacing the ADC0804 to the [2051](#)

The [AT89C2051](#) is a general purpose microcontroller. It is a 20 pin version of the 8051 and uses the same language. See below for more information about programming the chip.

To control the ADC0804, we will use 3 lines from the 2051.

Connect pin 2 (Read) from the ADC0804 to pin 7 (P3.3) of the 2051.

Connect pin 3 (Write) to pin 8 (P3.4).

Connect pin 5 (Interrupt) to pin 9 (P3.5).

The 8 bit Output Data from the ADC0804 will be connected to Port 1 of the 2051.

Connect pin 18 (D0) of the ADC0804 to pin 12 of the 2051 (P1.0).

Connect pin 17 (D1) to pin 13 (P1.1).

Connect pin 16 (D2) to pin 14 (P1.2).

Connect pin 15 (D3) to pin 15 (P1.3).

Connect pin 14 (D4) to pin 16 (P1.4).

Connect pin 13 (D5) to pin 17 (P1.5).

Connect pin 12 (D6) to pin 18 (P1.6).

Connect pin 11 (D7) to pin 19 (P1.7).

The 2051 pins 12 and 13 do not have internal pull up resistors so external pull up resistors are required.

Connect a 2.2k ohm resistor from pin 12 of the 2051 to 5 volts.

Connect a 2.2k ohm resistor from pin 13 of the 2051 to 5 volts.

To power the 2051,

Connect pin 20 of the 2051 to 5 volts.

Connect pin 10 of the 2051 to ground.

For the 2051 oscillator,

Connect the 11 MHz Crystal from pin 4 of the 2051 to pin 5 of the 2051.

Connect one 33 pF capacitor from pin 4 of the 2051 to ground.

Connect the other 33 pF capacitor from pin 5 of the 2051 to ground.

For the 2051 reset circuit,

Connect the 8.2k ohm resistor from pin 1 of the 2051 to ground.

Connect the 10 uF capacitor from pin 1 of the 2051 to 5 volts.

The 2051 controls the analog to digital conversion process. The conversion process has several stages.

Stage 1) To trigger a new conversion, we must make pin 3 (Write) low and then return it to the high state. The conversion process starts when Write goes high (rising edge triggered).

Stage 2) When the conversion process is complete, pin 5 (Interrupt) will go low.

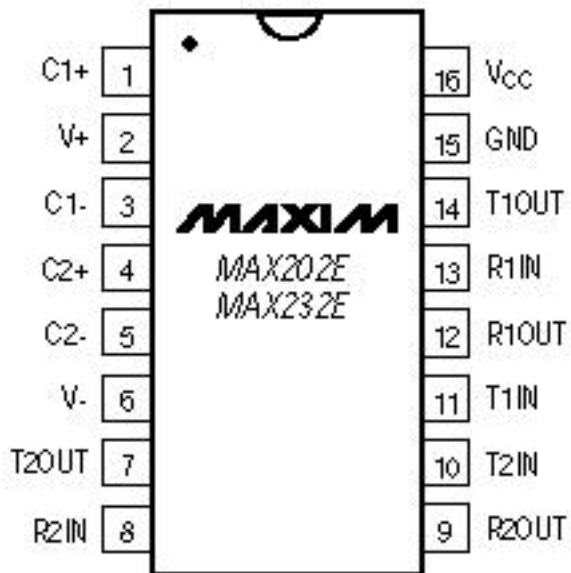
Stage 3) When we see pin 5 (Interrupt) go low, we must make pin 2 (Read) low to load the new value into the outputs D0 - D7.

Stage 4) Next we read the values into the 2051 Port 1.

Stage 5) Finally, we return pin 2 (Read) to the high state. The next conversion can be started immediately.

Step 3.) Communicating with the PC - The MAX232 IC

Now that we have the 8 bit value in the 2051, we want to send that value to the PC. The 2051 has a built in serial port that makes it very easy to communicate with the PC's serial port but the 2051 outputs are 0 and 5 volts and we need +10 and -10 volts to meet the RS232 serial port standard. The easiest way to get these values is to use the MAX232. The MAX232 acts as a buffer driver for the processor. It accepts the standard digital logic values of 0 and 5 volts and converts them to the RS232 standard of +10 and -10 volts. It also helps protect the processor from possible damage from static that may come from people handling the serial port connectors.



The MAX232 requires 5 external 1uF capacitors. These are used by the internal charge pump to create +10 volts and -10 volts.

For the first capacitor, the negative leg goes to ground and the positive leg goes to pin 16.

For the second capacitor, the negative leg goes to 5 volts and the positive leg goes to pin 2.

For the third capacitor, the negative leg goes to pin 3 and the positive leg goes to pin 1.

For the fourth capacitor, the negative leg goes to pin 5 and the positive leg goes to pin 4.

For the fifth capacitor, the negative leg goes to pin 6 and the positive leg goes to ground.

The MAX232 includes 2 receivers and 2 transmitters so two serial ports can be used with a single chip. We will only use one transmitter for this project. **The only**

connection that must be made to the 2051 is one jumper from pin 3 of the 2051 to pin 11 of the MAX232.

To power the MAX232,
Connect pin 16 to 5 volts.
Connect pin 15 to ground.

[For more information on the MAX232 chip click here to get the data sheet for it.](#)

The only thing left is that we need some sort of connector to connect to the serial port. The sample code below is written for Comm1 and most computers use a 9 pin DB9 male connector for Comm1 so a 9 pin female connector is included for this project. You may also want to buy a DB9 extension cable (Shown on order form as DB9 to DB9 cable) to make the connection easier. There should be 3 wires soldered to the DB9 connector pins 2, 3 and 5. **Connect the wire from pin 5 of the connector to ground on the breadboard. Connect the wire from pin 2 of the connector to pin 14 of the MAX232.** (The other wire is for receiving and is not used in this project.)

The Software

The basic process of compiling an assembly language program and loading it into the microcontroller was covered in the [first microcontroller project](#). The 2051 assembly language program for this project is [adcproj.asm](#). The 2051 in this kit comes preprogrammed with the adcproj program so you can build this kit without having a device programmer. If you need to alter the 2051 software you will need a device programmer such as the [PG302](#) to reprogram the 2051.

Make sure the power is off to the circuit you have built. Connect the circuit to the PC's serial port, Comm1. Turn on the power to the breadboard. The circuit should send a continuous stream of values to the PC. To see the values on the PC, try this [sample program](#). This program takes the received value and

divides it by 51 to find the corresponding voltage level. (The minimum value is 0 which is 0 volts and the maximum value is 255 which is 5 volts.) The source code for the sample program (written in VB 3.0 and 5.0) is included on the CD included with the kit. Two files your computer may need to run the sample program are [Vbrun300.dll](#) and [Mscomm.vbx](#)

Testing the Circuit

To test your circuit, connect various voltages to pin 6 of the ADC0804. If you connect a jumper from pin 6 to 5 volts, the voltage on the sample program should say 5 volts. Remove the jumper.

Try connecting a 2.2k resistor from pin 6 to ground and another 2.2k resistor from pin 6 to 5 volts. The result should be around 2.5 volts. Remove the resistors.

Try playing with the 220 uF capacitor and the 15k Ohm resistor. Connect the negative leg of the capacitor to ground and the positive leg to pin 6 of the ADC0804. Connect the resistor from pin 6 to 5 volts. The voltage should rise quickly and then slower as it approaches 5 volts. Now remove the resistor. The voltage should stay at the same voltage and slowly decay as the capacitor loses it's charge. Connect the resistor from pin 6 to ground to quickly discharge the capacitor.

You can order the parts for this project. It includes:

[1 - AT89C2051-24PC](#)

[Microcontroller](#) (programmed with the adcproj software)

1 - 11.0592 MHz Crystal

2 - 33pF Capacitors

1 - 10 uF Capacitor

1 - 220 uF Capacitor

5 - 2.2k Resistors

1 - 8.2k Resistor

1 - 10k Resistor

1 - 15k Resistor

1 - ADC0804

1 - 150 pF Capacitor

1 - MAX232

5 - 1 uF Capacitors

1 - DB9 Connector

Jumper Wires

1 - CD with source code ([click here to see full contents of CD](#))



For Price and Ordering Information, look at ADC2051Kit on [Order Information Page](#) or call 800-297-1633. If you want a printed copy of this and the other tutorials, remember to mark that on the order form.

You may also be interested in the Microcontroller Beginner Kit. It includes a power supply, a breadboard, and the PG302 so you can reprogram the 2051 microcontroller with your own custom programs. [Click here for more information.](#)

Send Questions to tech@iguanalabs.com

Iguana Labs

This page last updated on May 1, 2002.

Iguana Labs

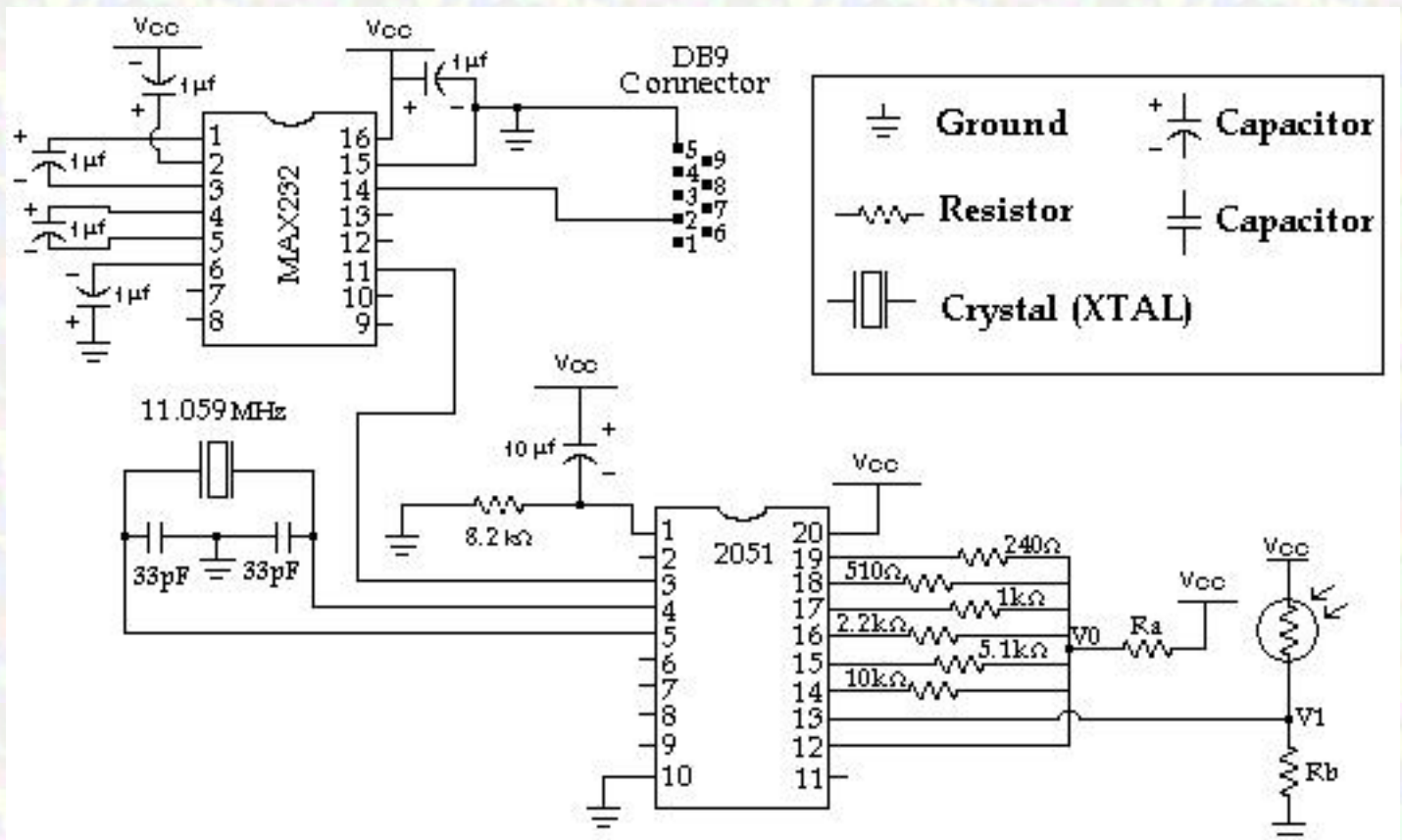
800-297-1633 (US)

Light Sensor - Collecting Light Data and Sending it to a PC

This tutorial shows how to set up a microcontroller based system that converts a signal from a light sensor to a 6 bit digital value. This value can be used by the microcontroller, perhaps for a robotic controller, or as in this tutorial, sent to the PC. It uses the [AT89C2051](#) microcontroller to collect data and send it to the PC. A MAX232CPE chip is used to convert the signals from and to RS232 levels for sending and receiving through the serial port. The 2051 microcontroller has a built in analog comparator that is used to make a simple analog to digital converter to convert the light sensor output to a digital value.

This tutorial is similar to the [Data Collection tutorial](#).

Refer to the diagram below to build the circuit. The [Data Collection Tutorial](#) has more detailed instructions on building the circuit (except for the resistor network).



The Light Sensor

Light sensors are one of the most common types of sensors. They are used in night lights, street lights, alarms, toys, cameras, etc. We are using a CDS (Cadmium Sulfide) Photocell to detect light. The resistance of the sensor varies based on the amount of light that hits it. The resistance can vary from 300K in the dark to 1K in the light. Our goal is to convert this into a digital value. We have to convert the variable resistance into a voltage and then the voltage into a digital value. To convert the resistance into a voltage, we use a second resistor R_b . Then assuming that no current goes into pin 13, you can find V_1 . To find V_1 you can use Ohms Law. With $V_{cc} = 5$, Ohms Law gives you

$$(5 - V_1) / R_{pc} = V_1 / R_b$$

where R_{pc} is the resistance of the sensor (photocell).

Then you can solve for $V_1 = 5 * R_b / (R_{pc} + R_b)$

Then the maximum voltage is when R_{pc} is at its minimum, 1K. Then $V_{1max} = 5 * R_b / (1k + R_b)$

The minimum voltage is when $R_{pc} = 300K$. $V_{1min} = 5 * R_b / (300k + R_b)$.

Using the equations for V_1 max and min you can determine that 5.1K is a good value for R_b . 5.1K gives you a wide voltage range from minimum to maximum. 5.1K works well for general light to dark situations. If you are only interested in bright environments (are you making an outside robot?) then use a larger value of R_b to shift the light sensitivity range towards bright lights (Perhaps 50K). Or if you are interested in dark environments (are you making a robotic vampire dog that barks at the moon and hides from bright lights?) then use a smaller value of R_b (perhaps 510).

Now we have a sensor voltage, V_1 , that varies from about 0.1 volts to 4.2 volts. If you want to take the easy route you can use the hardware set up in the [Data Collection tutorial](#) to convert this voltage to an eight bit digital value.

Analog to Digital Conversion using the 2051

The [Data Collection Tutorial](#) shows how to use a ready made Analog to Digital converter chip to get data from an analog source. This tutorial shows a different method. This method is not as accurate as the ADC0804 but it is less expensive, uses less power, and is easily modified to suit specific needs.

This method uses the built in analog comparator on the 2051 (not a normal feature of 8051 based chips). The voltage generated by the sensor circuit is connected to the negative input of the comparator (P1.1) and we will generate a voltage to connect to the positive input of the comparator (P1.0). The output of the comparator goes to P3.6. P3.6 is not an external pin on the 2051. It can only be accessed by the

internal software. If the voltage at P1.0 is higher than P1.1 then P3.6 will be a 1. If the voltage at P1.0 is lower than P1.1 then P3.6 will be a 0.

By using the other 6 Port 1 pins (P1.2 through P1.7) we can generate a voltage using a resistor network connected to those pins. By changing the values of the Port 1 pins we will get as close as possible to matching the voltage from the sensor circuit. Then we will have a 6 bit digital value that is a reflection of the sensor voltage at P1.1.

Each of the 6 Port 1 pins is connected to V0 through a resistor. Setting a pin to 0 or 1 subtracts from the voltage at V0 or doesn't. The value of the resistor determines how much voltage is subtracted. If all 6 pins are set to 1 then no current is flowing through the resistor network and $V0 = 5V$. The small resistor on P1.7 (240 ohms) can subtract the most voltage. When we set it to 0 current flows through R_a and the voltage at V0 goes down. The exact amount depends on the value of R_a . The resistors are chosen so they are roughly twice the value of the resistor connected to the next higher pin. (Ideally they would be exactly double the other value but it is difficult to get resistors that have exactly the right values.) By doubling the resistance, the pin can subtract half as much voltage. When you get to P1.2 with the 10K resistor, it only has a small effect on the voltage at V0 when you set P1.2 to 0 or 1.

Now we can control the voltage at V0 fairly accurately with P1.2 through P1.6. To make a small change in voltage, change the lower pins and to make a large change in voltage change the higher pins. By starting with P1.6 through P1.2 set to 000000 (P1.6 is on the left and P1.2 is on the right) and counting up to 111111 you can get 64 different voltages!

To find the right voltage to match the voltage at P1.1, we start at 000000 and count up until the comparator output at P3.6 switches to 1 to tell us our generated voltage is higher than the sensor voltage. Then we can "track" the voltage by adjusting the value up and down depending on the output of the comparator. Since the comparator only tells us high or low (it can not tell you if you have an exact match) then one possibly annoying aspect of this approach is that the P1.2 bit is constantly switching from 0 to 1 to 0 to 1... as the comparator output tells us we are low, then high, then low. To avoid having to watch the 6 bit value oscillate we just use the top 5 digits as our answer. Look at the documentation in the software for the 2051 in [light.asm](#) for more details on the tracking routine.

The details of communicating with the PC are covered in the [Data Collection Tutorial](#). For this project we are sending the upper 5 digit value (P1.3 through P1.6) to the PC. This can be displayed on the screen using the [sample light program](#).

The Software

The basic process of compiling an assembly language program and loading it into the microcontroller was covered in the [first microcontroller project](#). The 2051 assembly language program for this project is [light.asm](#). The 2051 in this kit comes preprogrammed with the light program so you can build this kit

without having a device programmer. If you need to alter the 2051 software you will need a device programmer such as the [PG302](#) to reprogram the 2051.

Make sure the power is off to the circuit you have built. Connect the circuit to the PC's serial port, Comm1. Connect the power to the breadboard. The circuit should send a continuous stream of values to the PC. To see the values on the PC, try this [sample light program](#). The source code for the sample program (written in VB 5.0) is on the CD included with the kit. Two files your computer may need to run the sample program are [Vbrun300.dll](#) and [Mscomm.vbx](#)

You can order the parts for this project. It includes:

[1 - AT89C2051-24PC Microcontroller](#) (programmed with the light software)

1 - 11.0592 MHz Crystal

2 - 33pF Capacitors

1 - 10 uF Capacitor

1 - 8.2k Resistor

5 - 240 Ohm Resistors

5 - 510 Ohm Resistors

5 - 1k Resistors

5 - 2.2k Resistors

5 - 5.1k Resistors

5 - 10k Resistors

1 - MAX232

5 - 1 uF capacitors

1 - DB9 connector

1 - CDS Photocell Light Sensor

Jumper Wires

1 - CD with source code ([click here to see full contents of CD](#))

For Price and Ordering Information, look at LightKit on the [Order Information Page](#) or call 800-297-1633.

You may also be interested in the Microcontroller Beginner Kit. It includes a power supply, a breadboard, and the PG302 so you can reprogram the 2051 microcontroller with your own custom programs. [Click here for more information.](#)

Send Questions to tech@iguanalabs.com

Iguana Labs

This page last updated on August 1, 2002.

LCD Instruction Set

Instruction	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Description
Display Clear	0	0	0	0	0	0	0	0	0	1	Clears all Display and sets DD RAM address to 0 in the address counter.
Cursor Home	0	0	0	0	0	0	0	0	1	*	Sets DD RAM address to 0. Contents remain unchanged.
Set Entry Mode	0	0	0	0	0	0	0	1	I/D	S	Sets cursor direction and specifies shift. (These operations are performed during writing/reading data.)
Display ON/OFF control	0	0	0	0	0	0	1	D	C	B	Sets Display ON/OFF (D), cursor ON/OFF (C), cursor blink (B).
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	*	*	Shifts cursor, keeping DD RAM contents.
Function Set	0	0	0	0	1	IF	*	*	*	*	Sets Data Length (IF)
Brightness Control (VFD Only)	1	0	*	*	*	*	*	*	BR1	BR0	Accepts 1 byte data of just after "Function Set" as brightness control data.
CG RAM address setting	0	0	0	1	A5	A4	A3	A2	A1	A0	Sets the CG RAM address.
DD RAM address setting	0	0	1	A6	A5	A4	A3	A2	A1	A0	Sets the DD RAM address.
Busy Flag & address reading	0	1	BF	A6	A5	A4	A3	A2	A1	A0	Reads Busy Flag (BF) and address counter.
Data Writing to CG or DD RAM	1	0	A7	A6	A5	A4	A3	A2	A1	A0	Writes data into CG RAM or DD RAM
Data Reading from CG or DD RAM	1	1	A7	A6	A5	A4	A3	A2	A1	A0	Reads data from CG RAM or DD RAM

* - Don't Care

I/D: 1 = Increment, 0 = Decrement
 S: 1 = Display Shift Enabled, 0 = Cursor Shift Enabled
 S/C: 1 = Display Shift, 0 = Cursor Move
 R/L: 1 = Shift to the Right, 0 = Shift to the Left
 IF: 1 = 8 bits, 0 = 4 bits
 BF: 1 = Busy, 0 = Not Busy
 BR1, BR0: 00 = 100%, 01 = 75%, 10 = 50%, 11 = 25%
 DD RAM = Display Data RAM
 CG RAM = Character Generator RAM

Links

[ExpBoard with LCD Module](#)

[8051 Device Programmer](#)

[Analog to Digital Converter Kit](#)

[Microcontroller Beginner Kit](#)

[Order Page](#)

[Back to Tutorials Menu](#)

This page last updated on March 21, 2002.

Microcontroller Beginner Kit (MBKit)

The Microcontroller Beginner Kit includes everything you need to get started in electronics. It starts with basic electricity. After explaining some electrical concepts, the basic electronic parts are presented. These parts are used to build simple circuits to demonstrate how the basic parts work. Then an Integrated Circuit, the 555 timer, is introduced and is used to make an LED blink. Control of the blinking speed is explained. Next microcontrollers are introduced (8051 microcontrollers). The manual works through some simple projects that show how to program the microcontroller and use it in some simple circuits with LEDs, a speaker, a 7 segment display, and input switches. After working through the manual, you will be able to program the microcontroller with your own custom programs. To get a more detailed look at what is in the manual, take a look at the web version of the instruction manual at <http://www.iguanalabs.com/mbktut> . This kit does not include the parts or instructions for the [Light Sensor Kit](#) or the [ADC2051Kit](#).

This kit includes all the parts required to do the following projects:

Transistors and LEDs

555 Timer IC (Pulses, Oscillators, Clocks, Etc))

5 Volt Power Supply

First Microcontroller Project - 2051 Kit

Simple Embedded System

Making Sound with the 2051

Using a 7 Segment Display

Using Switches as Inputs

The MBKit comes with:



(Click Pictures for Large Version)

Resistors:

- 10 - 100 ohm resistors
- 10 - 330 ohm resistors
- 5 - 510 ohm resistors
- 5 - 2,200 ohm resistors
- 5 - 8,200 ohm resistor
- 5 - 10,000 ohm resistors
- 5 - 15,000 ohm resistor
- 5 - 100,000 ohm resistors

Capacitors:

- 2 - 33pf capacitors
- 2 - 10uf capacitors

2 - 220uf capacitors

Other Components:

10 - LEDs (Green)

1 - 7 Segment Display

5 - NPN Transistors

1 - 11.0592 MegaHertz Crystal

1 - LM7805, 5 Volt Regulator

2 - 555 Timer ICs

2 - [AT89C2051 Microcontrollers](#) (20 pin versions of the popular 8051 microcontroller)

1 - Speaker

2 - Normally Open Push Button Switches

1 - Power Supply Adapter (To use the power supply with the breadboard)

IC Extractor

Small Breadboard

[PG302 Device Programmer \(click here for more information on PG302\)](#)

Power Supply (US and Canada Orders Only)

6 Ft DB9 Cable

CD with [PG302 Software for Windows \(Free Updates From Web Site\)](#) and MBKit Software (includes compiler, assembly language editor, and more) [Click here to see list of CD contents.](#)

45 page Printed Color Manual

Free Shipping! (USPS Priority Mail (2-4 Days). US Orders Only. Faster shipping methods are also available.)

Technical Support (During business hours, call 800-297-1633 or send an email to support@iguanalabs.com)

For price and ordering information [click here for the Order Information Page](#) or call 800-297-1633

Send any questions to support@iguanalabs.com

Iguana Labs

800-297-1633 (US)

Iguana Labs CD Contents

PG302 Software

Latest Version 5.20 (8/8/02)

Linux Version - For 2051 only

PG302 Auto - For the AT90S2313, AT90S8515, and AT90S1200. Programs Flash, EEPROM, and sets the lock bits by only pressing one button. Look at the included file Instruct.txt for detailed information.

Windows 3.1 - For that old computer

Microcontroller Beginner Kit Software

AY Pad - Customizable Text Editor featuring color syntax highlighting for 8051 assembly language, AVR assembly language and others.

Data Sheets in PDF format - Hardware Description, Architecture Description, and Memory Description

Assembly Language Files for Kit

Tasm - Turbo Assembler used for compiling assembly language programs for kit

ExpBoard Software

AY Pad - Customizable Text Editor featuring color syntax highlighting for 8051 assembly language, AVR assembly language and others.

Assembly Language Files for ExpBoard

Tasm - Turbo Assembler used for compiling 8051 assembly language programs

avrasm.exe - Command Line Compiler for AVR assembly language programs

ADC2051 Kit Software

Assembly Language Files

Data Sheets in PDF format - ADC0804, MAX232

Visual Basic Source Code Files for versions 3.0 and 5.0

Tools

binhex.zip - Binary to Hex converter - converts files in Binary format to Hex format so the PG302 can use them

bas051.zip - Basic Compiler for the 8051

pacific.exe - C compiler for the 8051
emily52.zip - Dos based simulator
sim51eng.zip - Dos based simulator
disasm.zip - Two programs that can convert hex files to assembly language (disassemblers)
astudio3.exe - AVR assembler and simulator
avrasm.exe - Command Line Compiler for AVR assembly language programs
pkz204g.exe - PK Zip tools for creating and extracting zip files
Adobe Acrobat pdf Reader

Data Sheets (in pdf format)

AT90S1200
AT90S2313
AT90S8515
AT89C2051
AT89C4051
AT89C51
AT89C52
AT89S53
AT89S8252
AVR Instruction Set
8051 Programmers Guide and Instruction Set
MAX232CPE
ADC0804
DAC0830

This disk is included with the [PG302](#), [ExpBoard](#), [ADC2051Kit](#), and the [Microcontroller Beginner Kit](#). It can also be ordered separately. See [Order Information Page](#) for pricing or call 800-297-1633.

This page last updated on August 2, 2002.

Iguana Labs

800-297-1633 (US)

PG302 Programmer Software

[Features](#) - [Latest Software](#) - [Ordering Information](#) - [PG302 Page](#)



Features:

Works on Windows 3.1, 95, 98, Me, 2000, NT, XP

Programs Lock Bits

Chip Checksum

Optional Auto Erase

Optional Auto Verify

Optional Auto Difference Check

Status Bar

Easy access to any of the latest files you've programmed.

Select Comm1, 2, 3 or 4

On Start Up, restores last settings (Comm Port, Device, etc.)

Very easy to use

Download Latest Version - v5.20 (8/8/02)

[Click here for v5_20.exe \(self extracting exe\)](#)

[Click here for v5_20.zip \(zipped version\)](#)

Upgrade for 5.0 versions and up

[pg302.exe](#) - This is the uncompressed executable file for version 5.20. Download it and replace the existing pg302.exe with this new pg302.exe. (To fix disappearing help file problem, get the full version above.)

[files.exe](#) - Includes all the support files that may be missing. Can be used with pg302.exe above to bypass the normal installation process (sometimes necessary for NT systems). Put this self extracting file in the same directory as pg302.exe and run it.

Other Versions (all compressed as self extracting exes)

[pg302auto.exe](#) - For the AT90S2313, programs Flash, EEPROM, and sets the lock bits by only pressing one button. Look at the file [Instructions.txt](#) for detailed information.

[setup301.exe](#) - Version 3.01 can be used on older MS Windows such as Windows 3.1.

Software Hints and Tips

You can install multiple copies of the software and configure each one differently.

-Each copy must be in it's own directory, but the hex files can be anywhere and can be shared by the different copies. This is handy if you often program different types of chips or work on different projects at once. You can install a copy for each chip type and project. Then each copy can be configured for a specific chip with just the files needed for that chip or project. For example, you can have one configured for programming a 2051 that just lists the files for the 2051. Then you can have another copy that is configured for programming a 2313 that just lists the files used for the 2313.

-Also, the various copies of the software can all be running at the same time. This is nice if you need to constantly reprogram both the Flash and the EEPROM. You can have one copy running that programs the Flash and another copy running that programs the

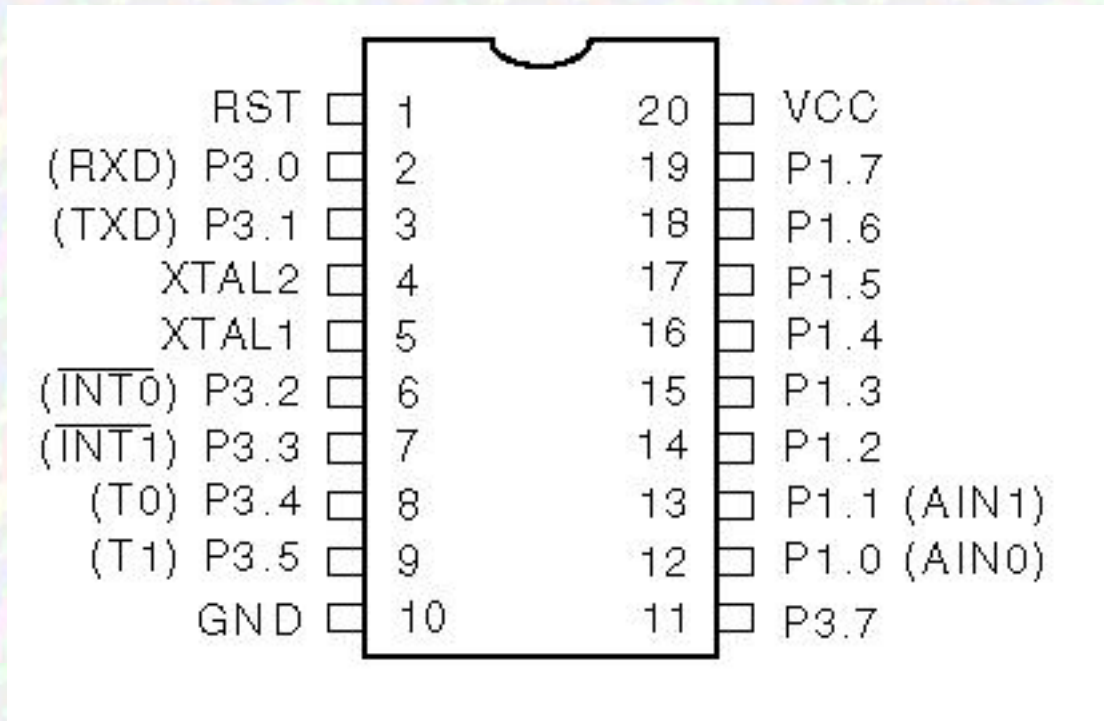
EEPROM so you don't have to keep switching the settings and file names.

-Make a shortcut on the desktop for each copy with a different name so you can easily start up the version you currently need.

-Install a new copy each time you start a new project so you will have a version that only lists the files you need for that project.

**[Back to PG302
home page](#)**

ATMEL AT89C2051 Pinout and Description



The 2051 is a low voltage (2.7V - 6V), high performance CMOS 8-bit microcontroller with 2 Kbytes of Flash programmable and erasable read only memory (PEROM). This device is compatible with the industry standard 8051 instruction set and pinout. The 2051 is a powerful microcomputer which provides a highly flexible and cost effective solution to many embedded control applications. The 2051 provides the following features:

- ~ 2 Kbytes of Flash
- ~ 128 bytes of RAM
- ~ 15 I/O lines
- ~ two 16-bit timer/counters
- ~ five vector, two-level interrupt architecture
- ~ full duplex serial port
- ~ precision analog comparator
- ~ on chip oscillator and clock circuitry

In addition, the 2051 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM,

timer/counters, serial port and interrupt system to continue functioning. The Power Down Mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

[To learn how to use the 2051 microcontroller, get the Microcontroller Beginner Kit \(click here\)](#)

2051 - 24MHz DIP Available

See [Order Form](#) For Pricing or call 800-297-1633

The 2051 can be programmed using a device programmer such as the [PG302](#) ([Click Here](#)).

We also have Projects that use the 2051 Microcontroller.
See the [Intermediate Tutorial Section](#).

This page last updated on May 1, 2002.

Basic Definitions and Concepts

Introduction

Welcome to the exciting world of electronics. Before we can build anything we need to look at a couple of things. Anytime you have an electrical circuit, you have voltage and electricity. We build circuits to control voltage and current.

Current

Current is what flows through a wire. Think of it as water flowing in a river. The current flows from one point to another point just like water in a river. Current flows from points of high voltage to points of low voltage. Current can be shown in circuit diagrams by using arrows as in Figure 1. The arrow shows which way the current is flowing. An I is usually included beside the arrow to indicate current.



Figure 1

The unit of measurement for current is the Ampere, or Amp for short, and abbreviated as A. (The name Ampere comes from Mr. Ampere who played with electricity as a small boy in Vermont.) Common currents are 0.001 Amps (0.001A) to 0.5 Amps (0.5A). Since currents are usually small, they are usually given in the form of milliAmps (abbreviated mA.) The milli means divided by 1000, so 0.001 Amps equals 1 milliAmp (1 mA) since $1 / 1000 = 0.001$. Also, 0.5 Amps equals 500 milliAmps (500mA) since $500 / 1000 = 0.5$.

Voltage

Voltage indicates the power level of a point. Voltage is measured in volts. If we continue the river

comparison, a point at the top of a hill would be at a high voltage level and a point at the bottom of a hill would be at a low voltage level. Then, just as water flows from a high point to a low point, current flows from a point of high voltage to a point of low voltage. If one point is at 5 volts and another point is at 0 volts then when a wire is connected between them, current will flow from the point at 5 volts to the point at 0 volts.

A measurement of voltage is much like a measurement of height. It gives you the difference in voltage between those two points. If point A is at 10 volts and point B is at 2 volts then the voltage measured between A and B is 8 volts ($10 - 2$). This is similar to measuring height. We measure the height of hills the same way. We say the sea level is at zero feet and then compare other points to that level. On top of Mary's Peak you are 4000 ft high (compared to sea level). In the same way we call the lowest voltage in a circuit zero volts and give it the name ground. Then all other points in the circuit are compared to that ground point. Rivers always flow towards sea level and currents always flow towards ground.

A battery is similar to a dam. On one side is a lot of stored up energy. When a path is formed from that side to the other side then current flows. If there is no path then current does not flow and the energy just stays there waiting for a path to form to the other side. The path can be a big path with lots of current flowing or a small path with just a little bit of current flowing. With a dam, a little bit of water flow could go on for a long time, but flow through a big path that lets all the water go at once would only last a short while. A battery is the same. If there is big path from the high voltage side to the low voltage side then the battery will not last long.

There are two special cases that we give names. One is when the current is zero (open circuit) and the other is when the voltage is zero (short circuit).

Open Circuit

An open circuit is when two points are not connected by anything. No current flows and nothing happens. If a wire in your vacuum cleaner breaks it can cause an open circuit and no current can flow so it does not do anything. There may be a voltage between those two points but the current can not flow without a connection.

Short Circuit

A short circuit (or short) is when two points with different voltage levels are connected with no resistance (see resistors) between two points. This can cause a large amount of current to flow. If a short circuit happens in your house, it will usually cause a circuit breaker to break or a fuse to blow. If there is no device to limit the current, the wires may melt and cause a fire. This situation is something like a dam breaking. There is a large amount of energy suddenly free to flow from a high point to a low point with nothing to limit the current.

Series Connection

A series connection is when two components are joined together by a common leg and nothing else is connected to that point as shown in Figure 2.



Figure 2

Parallel Connection

A parallel connection is when two components are joined together by both legs as shown in Figure 3.

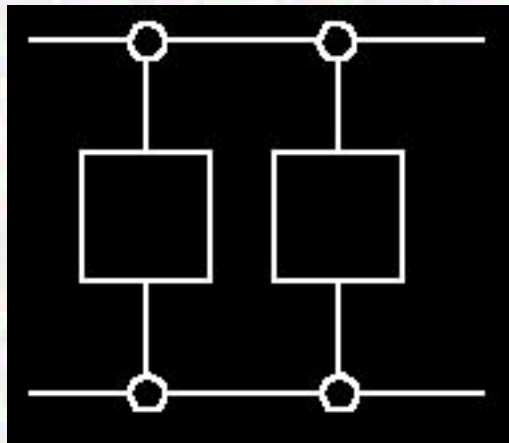


Figure 3

[Back to Tutorials Menu](#)

[Next - Basic Components](#)

Iguana Labs

This page last updated March 19, 2002

Basic Electrical Components

Resistors

Resistors are components that have a predetermined resistance. Resistance determines how much current will flow through a component. Resistors are used to control voltages and currents. A very high resistance allows very little current to flow. Air has very high resistance. Current almost never flows through air. (Sparks and lightning are brief displays of current flow through air. The light is created as the current burns parts of the air.) A low resistance allows a large amount of current to flow. Metals have very low resistance. That is why wires are made of metal. They allow current to flow from one point to another point without any resistance. Wires are usually covered with rubber or plastic. This keeps the wires from coming in contact with other wires and creating short circuits. High voltage power lines are covered with thick layers of plastic to make them safe, but they become very dangerous when the line breaks and the wire is exposed and is no longer separated from other things by insulation.

Resistance is given in units of ohms. (Ohms are named after Mho Ohms who played with electricity as a young boy in Germany.) Common resistor values are from 100 ohms to 100,000 ohms. Each resistor is marked with colored stripes to indicate its resistance. To learn how to calculate the value of a resistor by looking at the stripes on the resistor, go to [Resistor Values](#) which includes more information about resistors.

Variable Resistors

Variable resistors are also common components. They have a dial or a knob that allows you to change the resistance. This is very useful for many situations. Volume controls are variable resistors. When you change the volume you are changing the resistance which changes the current. Making the resistance higher will let less current flow so the volume goes down. Making the resistance lower will let more current flow so the volume goes up. The value of a variable resistor is given as its highest resistance value. For example, a 500 ohm variable resistor can have a resistance of anywhere between 0 ohms and 500 ohms. A variable resistor may also be called a potentiometer (pot for short).

Diodes

Diodes are components that allow current to flow in only one direction. They have a positive side (leg) and a negative side. When the voltage on the positive leg is higher than on the negative leg then current flows through the diode (the resistance is very low). When the voltage is lower on the positive leg than on the negative leg then the current does not flow (the resistance is very high). The negative leg of a diode is the one with the line closest to it. It is called the cathode. The positive end is called the anode.

Usually when current is flowing through a diode, the voltage on the positive leg is 0.65 volts higher than on the negative leg.

LED

Light Emitting Diodes are great for projects because they provide visual entertainment. LEDs use a special material which emits light when current flows through it. Unlike light bulbs, LEDs never burn out unless their current limit is passed. A current of 0.02 Amps (20 mA) to 0.04 Amps (40 mA) is a good range for LEDs. They have a positive leg and a negative leg just like regular diodes. To find the positive side of an LED, look for a line in the metal inside the LED. It may be difficult to see the line. This line is closest to the positive side of the LED. Another way of finding the positive side is to find a flat spot on the edge of the LED. This flat spot is on the negative side.

When current is flowing through an LED the voltage on the positive leg is about 1.4 volts higher than the voltage on the negative side. Remember that there is no resistance to limit the current so a resistor must be used in series with the LED to avoid destroying it.

To learn about LEDs through an interactive kit, look at [LED and Transistor Kit](#)

Switches

Switches are devices that create a short circuit or an open circuit depending on the position of the switch. For a light switch, ON means short circuit (current flows through the switch, lights light up and people dance.) When the switch is OFF, that means there is an open circuit (no current flows, lights go out and people settle down. This effect on people is used by some teachers to gain control of loud classes.)

When the switch is ON it looks and acts like a wire. When the switch is OFF there is no connection.

[Back to Tutorials Menu](#)

[Previous - Basic Definitions](#)

[Next - Finding The Value of a Resistor By Reading It's Color Codes](#)

Iguana Labs

This page last updated on March 19, 2002.

Finding the Value of a Resistor by Color Codes

To calculate the value of a resistor using the color coded stripes on the resistor, use the following procedure.

Step One: Turn the resistor so that the gold or silver stripe is at the right end of the resistor.

Step Two: Look at the color of the first two stripes on the left end. These correspond to the first two digits of the resistor value. Use the table given below to determine the first two digits.

Step Three: Look at the third stripe from the left. This corresponds to a multiplication value. Find the value using the table below.

Step Four: Multiply the two digit number from step two by the number from step three. This is the value of the resistor in ohms. The fourth stripe indicates the accuracy of the resistor. A gold stripe means the value of the resistor may vary by 5% from the value given by the stripes.

Resistor Color Codes (with gold or silver strip on right end)

Color	First Stripe	Second Stripe	Third Stripe	Fourth Stripe
Black	0	0	x1	
Brown	1	1	x10	
Red	2	2	x100	
Orange	3	3	x1,000	
Yellow	4	4	x10,000	
Green	5	5	x100,000	
Blue	6	6	x1,000,000	

Purple	7	7		
Gray	8	8		
White	9	9		
Gold				5%
Silver				10%

Follow the above procedure with the examples below and soon you will be able to quickly determine the value of a resistor by just a glance at the color coded stripes.

Examples

Example 1:

You are given a resistor whose stripes are colored from left to right as brown, black, orange, gold. Find the resistance value.

Step One: The gold stripe is on the right so go to Step Two.

Step Two: The first stripe is brown which has a value of 1. The second stripe is black which has a value of 0. Therefore the first two digits of the resistance value are 10.

Step Three: The third stripe is orange which means $\times 1,000$.

Step Four: The value of the resistance is found as $10 \times 1000 = 10,000$ ohms (10 kilohms = 10 kohms).

The gold stripe means the actual value of the resistor may vary by 5% meaning the actual value will be somewhere between 9,500 ohms and 10,500 ohms. (Since 5% of $10,000 = 0.05 \times 10,000 = 500$)

Example 2:

You are given a resistor whose stripes are colored from left to right as orange, orange, brown, silver. Find the resistance value.

Step One: The silver stripe is on the right so go to Step Two.

Step Two: The first stripe is orange which has a value of 3. The second stripe is orange which has a value of 3. Therefore the first two digits of the resistance value are 33.

Step Three: The third stripe is brown which means $\times 10$.

Step Four: The value of the resistance is found as $33 \times 10 = 330$ ohms.

The silver stripe means the actual value of the resistor may vary by 10% meaning the actual value will be between 297 ohms and 363 ohms. (Since 10% of $330 = 0.10 \times 330 = 33$)

Example 3:

You are given a resistor whose stripes are colored from left to right as blue, gray, red, gold. Find the resistance value.

Step One: The gold stripe is on the right so go to Step Two.

Step Two: The first stripe is blue which has a value of 6. The second stripe is gray which has a value of 8. Therefore the first two digits of the resistance value are 68.

Step Three: The third stripe is red which means $\times 100$.

Step Four: The value of the resistance is found as $68 \times 100 = 6800$ ohms (6.8 kilohms = 6.8 kohms).

The gold stripe means the actual value of the resistor may vary by 5% meaning the actual value will be somewhere between 6,460 ohms and 7,140 ohms. (Since 5% of $6,800 = 0.05 \times 6,800 = 340$)

Example 4:

You are given a resistor whose stripes are colored from left to right as green, brown, black, gold. Find the resistance value.

Step One: The gold stripe is on the right so go to Step Two.

Step Two: The first stripe is green which has a value of 5. The second stripe is brown which has a value of 1. Therefore the first two digits of the resistance value are 51.

Step Three: The third stripe is black which means $\times 1$.

Step Four: The value of the resistance is found as $51 \times 1 = 51$ ohms.

The gold stripe means the actual value of the resistor may vary by 5% meaning the actual value will be somewhere between 48.45 ohms and 53.55 ohms. (Since 5% of $51 = 0.05 \times 51 = 2.55$)

Other Resistor Information

There are some more rules that may be useful when working with resistors. You do not need to know them but if you need a resistor with a value that you do not have, you may be able to use the following information to create the value of resistor you need.

First Rule for Resistors : Series Connection

When two resistors are connected in series, as shown in Figure 1, the new resistance between points A and B is $R_1 + R_2$.



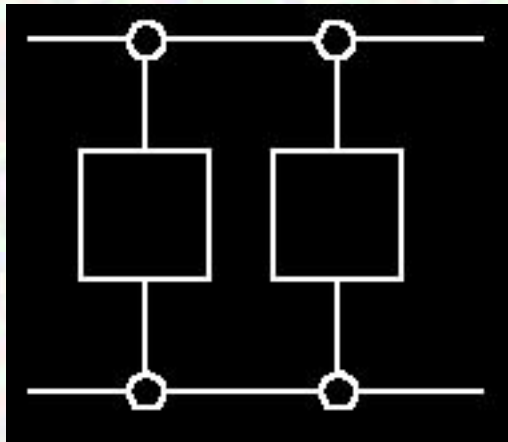
Figure 1

The resistors add together. For example if $R_1 = 500$ ohms and $R_2 = 250$ ohms then the resistance between points A and B would be $R_1 + R_2 = 500 + 250 = 750$ ohms.

Second Rule for Resistors : Parallel Connection

When two resistors are connected in parallel, as shown in Figure 2, the new resistance is smaller than either R_1 or R_2 . The new resistance between points A and B is $(R_1 \times R_2) / (R_1 + R_2)$.

A



B

Figure 2

For example, if $R_1 = 500$ and $R_2 = 250$ then the resistance between points A and B = $(500 \times 250) / (500 + 250) = (125,000) / (750) = 167$ ohms. If $R_1 = R_2$ then the new resistance is just $R_1 / 2$.

Using these two rules, resistors can be combined to form new resistance values.

[Back to Tutorials Menu](#)

[Previous - Basic Components](#)

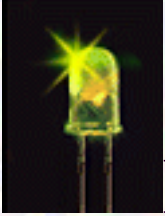
[Next - Using Ohm's Law](#)

Iguana Labs

This page last updated on March 19, 2002..

Iguana Labs

800-297-1633 (US)



Learning About Transistors and LEDs

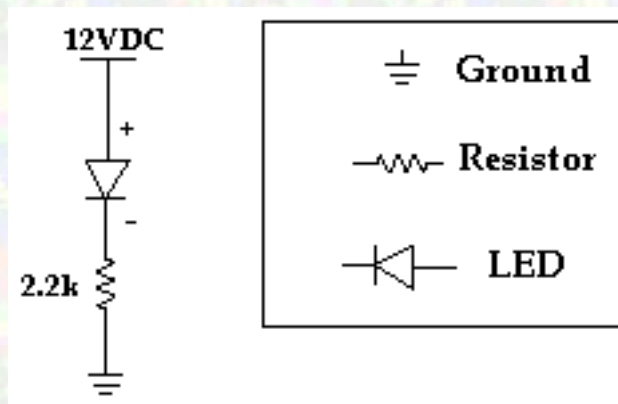
The LED



An LED is the device shown above. Besides red, they can also be yellow, green and blue. The letters LED stand for Light Emitting Diode. If you are unfamiliar with diodes, take a moment to review the components in the [Basic Components Tutorial](#). The important thing to remember about diodes (including LEDs) is that current can only flow in one direction.

To make an LED work, you need a voltage supply and a resistor. If you try to use an LED without a resistor, you will probably burn out the LED. The LED has very little resistance so large amounts of current will try to flow through it unless you limit the current with a resistor. If you try to use an LED without a power supply, you will be highly disappointed.

So first of all we will make our LED light up by setting up the circuit below.



Step 1.) First you have to find the positive leg of the LED. The easiest way to do this is to look for the leg that is longer.

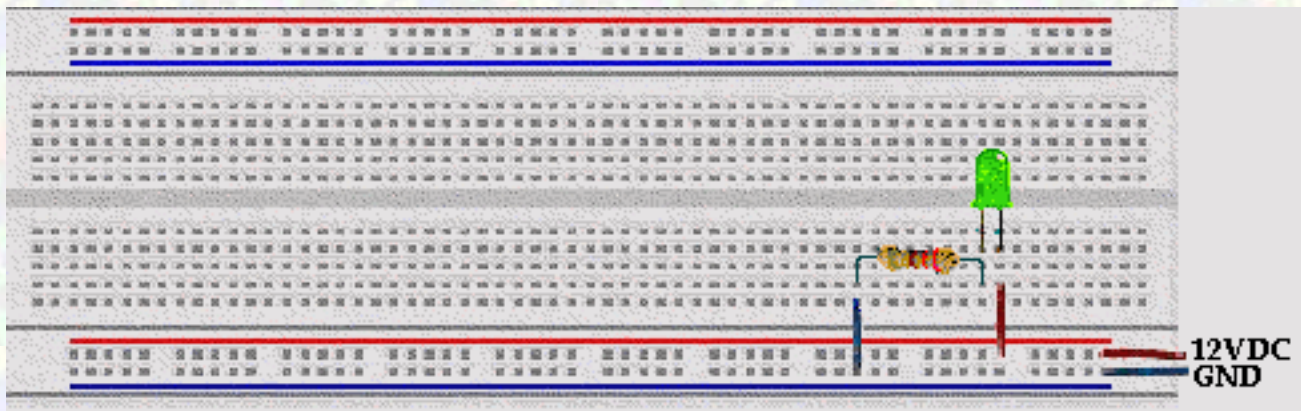
Step 2.) Once you know which side is positive, put the LED on your [breadboard](#) so the positive leg is in one row and the negative leg is in another row. (In the picture below the rows are vertical.)

Step 3.) Place one leg of a 2.2k ohm resistor (does not matter which leg) in the same row as the negative leg of the LED. Then place the other leg of the resistor in an empty row.

Step 4.) Unplug the power supply adapter from the power supply. Next, put the ground (black wire) end of the power supply adapter in the sideways row with the blue stripe beside it. Then put the positive (red wire) end of the power supply adapter in the sideways row with the red stripe beside it.

Step 5.) Use a short jumper wire (use red since it will be connected to the positive voltage) to go from the positive power row (the one with the red stripe beside it) to the positive leg of the LED (not in the same hole, but in the same row). Use another short jumper wire (use black) to go from the ground row to the resistor (the leg that is not connected to the LED). Refer to the picture below if necessary.

The breadboard should look like the picture shown below.



Now plug the power supply into the wall and then plug the other end into the power supply adapter and the LED should light up. Current is flowing from the positive leg of the LED through the LED to the negative leg. Try turning the LED around. It should not light up. No current can flow from the negative leg of the LED to the positive leg.

People often think that the resistor must come first in the path from positive to negative, to limit the amount of current flowing through the LED. But, the current is limited by the resistor no matter where the resistor is. Even when you first turn on the power, the current will be limited to a certain amount, and can be found using ohm's law.

Revisiting Ohm's Law

Ohm's Law can be used with resistors to find the current flowing through a circuit. The law is $I = VD/R$ (where I = current, VD = voltage across resistor, and R = resistance). For the circuit above we can only use Ohm's law for the resistor so we must use the fact that when the LED is on, there is a 1.4 voltage drop across it. This means that if the positive leg is connected to 12 volts, the negative leg will be at 10.6 volts. Now we know the voltage on both sides of the resistor and can use Ohm's law to calculate the current. The current is $(10.6 - 0) / 2200 = 0.0048$ Amperes = 4.8 mA

This is the current flowing through the path from 12V to GND. This means that 4.8 mA is flowing through the LED and the resistor. If we want to change the current flowing through the LED (changing the brightness) we can change the resistor. A smaller resistor will let more current flow and a larger resistor will let less current flow. Be careful when using smaller resistors because they will get hot.

Next, we want to be able to turn the LED on and off without changing the circuit. To do this we will learn to use another electronic component, the transistor.

1.6.1 The Transistor

Transistors are basic components in all of today's electronics. They are just simple switches that we can use to turn things on and off. Even though they are simple, they are the most important electrical component. For example, transistors are almost the only components used to build a Pentium processor. A single Pentium chip has about 3.5 million transistors.

The ones in the Pentium are smaller than the ones we will use but they work the same way.

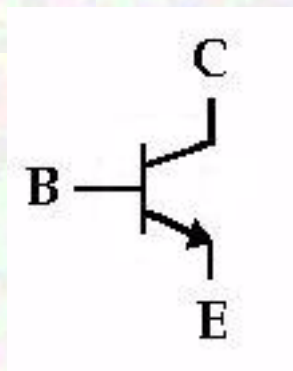
Transistors that we will use in projects look like this:



The transistor has three legs, the Collector (C), Base (B), and Emitter (E). Sometimes they are labeled on the flat side of the transistor. Transistors always have one round side and one flat side. If the round side is facing you, the Collector leg is on the left, the Base leg is in the middle, and the Emitter leg is on the right.

Transistor Symbol

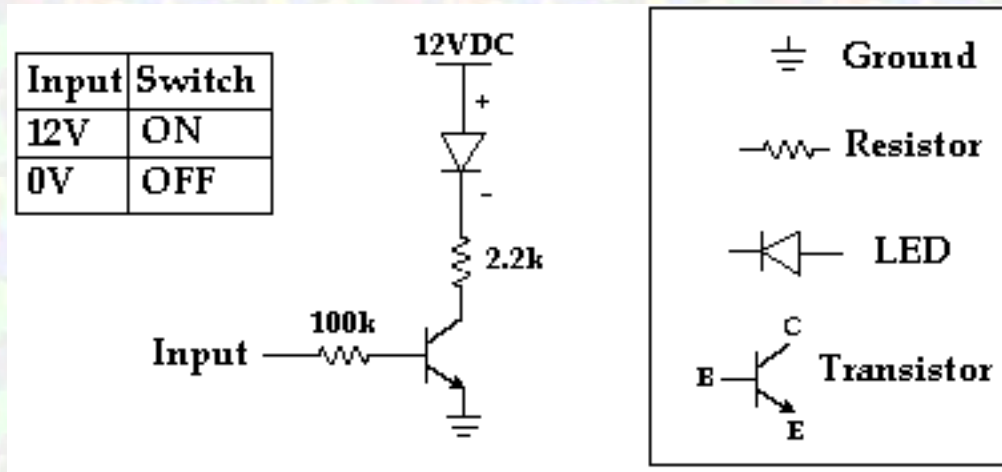
The following symbol is used in circuit drawings (schematics) to represent a transistor.



Basic Circuit

The Base (B) is the On/Off switch for the transistor. If a current is flowing to the Base, there will be a path from the Collector (C) to the Emitter (E) where current can flow (The Switch is On.) If there is no current flowing to the Base, then no current can flow from the Collector to the Emitter. (The Switch is Off.)

Below is the basic circuit we will use for all of our transistors.

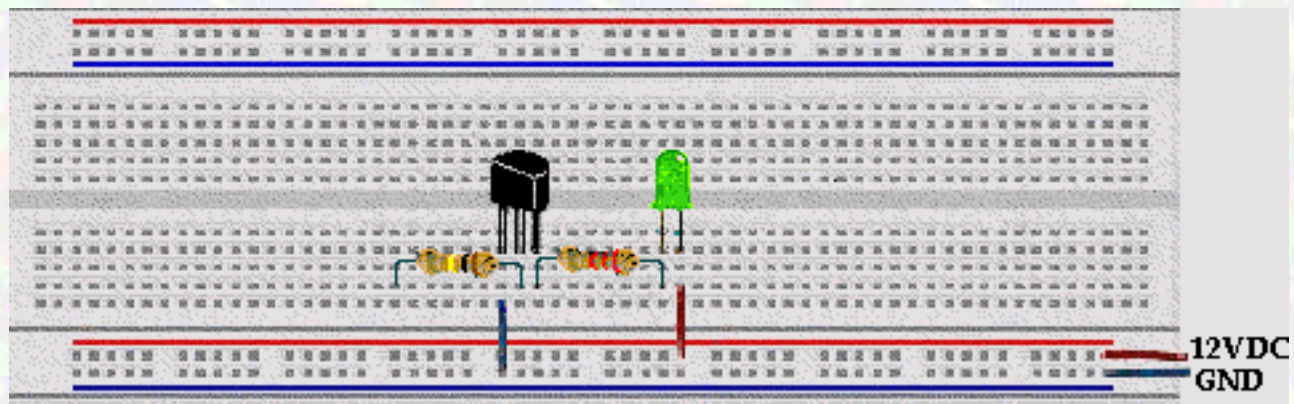


To build this circuit we only need to add the transistor and another resistor to the circuit we built above for the LED.

Unplug the power supply from the power supply adapter before making any changes on the breadboard. To put the transistor in the breadboard, separate the legs slightly and place it on the breadboard so each leg is in a different row. The collector leg should be in the same row as the leg of the resistor that is connected to ground (with the black jumper wire).

Next move the jumper wire going from ground to the 2.2k ohm resistor to the Emitter of the transistor.

Next place one leg of the 100k ohm resistor in the row with the Base of the transistor and the other leg in an empty row and your breadboard should look like the picture below.



Now put one end of a yellow jumper wire in the positive row (beside the red line) and the other end in the row with the leg of the 100k ohm resistor (the end not connected to the Base). Reconnect the power supply and the transistor will come on and the LED will light up. Now move the one end of the yellow jumper wire from the positive row to the ground row (beside the blue line). As soon as you remove the yellow jumper wire from the positive power supply, there is no current flowing to the base. This makes the transistor turn off and current can not flow through the LED. As we will see later, there is very little current flowing through the 100k resistor. This is very important because it means we can control a large

current in one part of the circuit (the current flowing through the LED) with only a small current from the input.

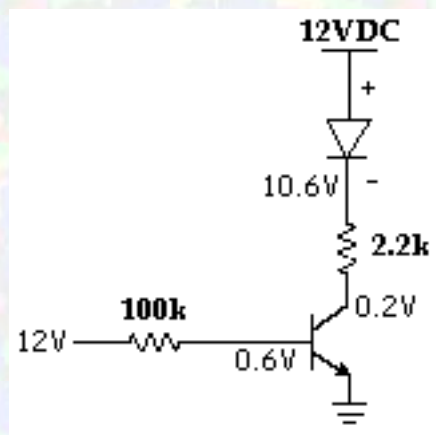
Back to Ohm's Law

We want to use Ohm's law to find the current in the path from the Input to the Base of the transistor and the current flowing through the LED. To do this we need to use two basic facts about the transistor.

1.) If the transistor is on, then the Base voltage is 0.6 volts higher than the Emitter voltage.

2.) If the transistor is on, the Collector voltage is 0.2 volts higher than the Emitter voltage.

So when the 100k resistor is connected to 12VDC, the circuit will look like this:



So the current flowing through the 100k resistor is $(12 - 0.6) / 100000 = 0.000114 \text{ A} = 0.114 \text{ mA}$.

The current flowing through the 2.2k ohm resistor is $(10.6 - 0.2) / 2200 = 0.0047 \text{ A} = 4.7 \text{ mA}$.

If we want more current flowing through the LED, we can use a smaller resistor (instead of 2200) and we will get more current through the LED without changing the amount of current that comes from the Input line. This means we can control things that use a lot of power (like electric motors) with cheap, low power circuits. Soon you will learn how to use a microcontroller (a simple computer). Even though the microcontroller can not supply enough current to turn lights and motors on and off, the microcontroller can turn transistors on and off and the transistors can control lots of current for lights and motors.

For Ohm's law, also remember that when the transistor is off, no current flows through the transistor.

1.6.2 Introduction to Digital Devices - The Inverter

In digital devices there are only two values, usually referred to as 0 and 1. 1 means there is a voltage (usually 5 volts) and 0 means the voltage is 0 volts.

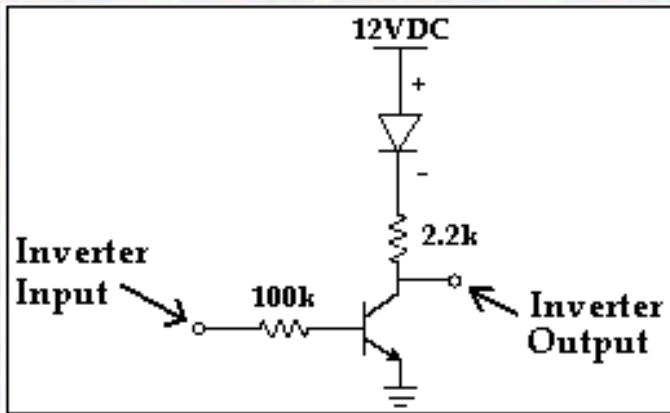
An inverter (also called a NOT gate) is a basic digital device found in all modern electronics. So for an inverter, as the

name suggests, its output is the opposite of the input (Output is NOT the Input). If the input is 0 then the output is 1 and if the input is 1 then the output is 0. We can summarize the operation of this device in a table.

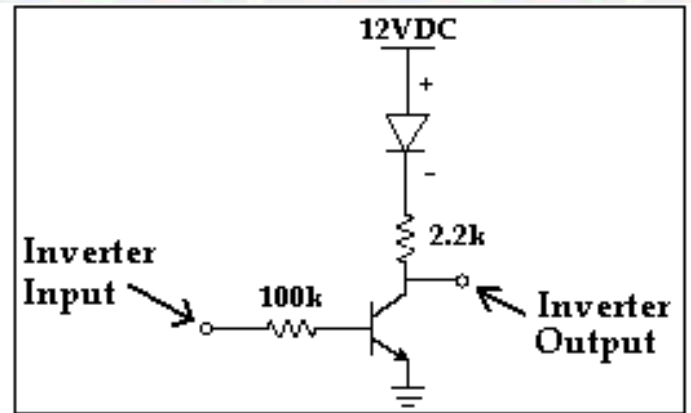
Input	Output
1	0
0	1

To help us practice with transistors we will build an inverter. Actually we have already built an inverter. The transistor circuit we just built is an inverter circuit. To help see the inverter working, we will build a circuit with two inverters. The circuit we will use is shown below.

First Inverter (already built)



Second Inverter



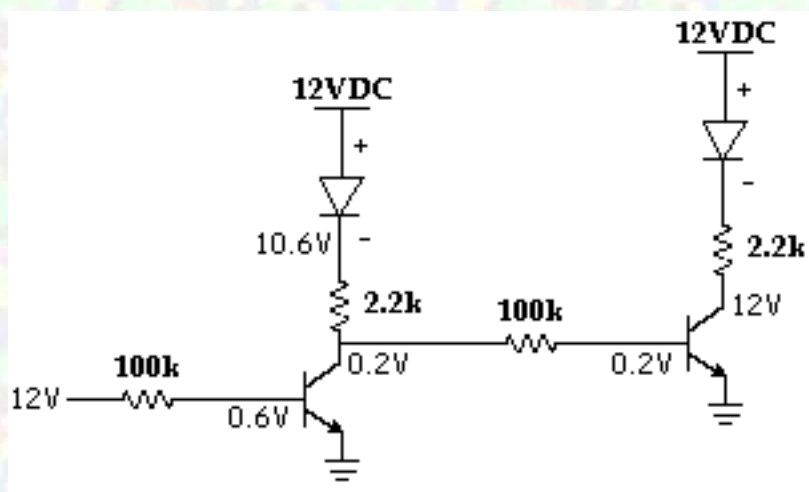
To build the circuit, use the transistor circuit we just built as the first inverter. The first inverter input is the end of the 100k ohm resistor connected to the yellow jumper wire. Build another circuit identical to the first (the basic transistor circuit from Section 1.6.1) except leave out the yellow jumper wire connected to the 100k ohm resistor (the inverter input). This circuit is the second inverter.

Connect the output of the first inverter to the input of the second inverter by putting one end of a jumper wire in the same row of holes as the 2.2k ohm resistor and the Collector of the transistor (the output of the first inverter) and putting the other end in the same row of holes as the leg of the 100k ohm resistor of the second inverter (the input to the second inverter).

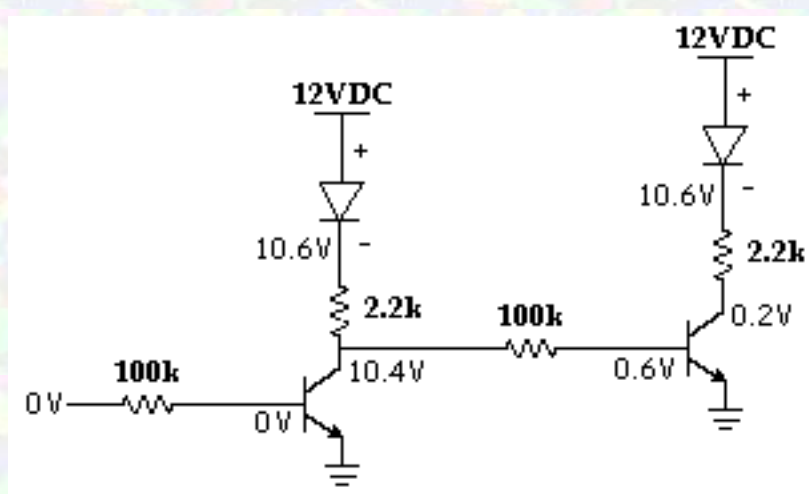
Here is how to check if you built it correctly. Connect the first inverter input (the yellow jumper wire) to 12V (the positive row). The LED in the first inverter should come on and the LED in the second inverter should stay off. Then connect the first inverter input to 0V (the ground row). (You are turning off the switch of the first inverter.) The first LED should go off and the second LED should come on. If this does not happen, check to make sure no metal parts are touching. Check to make sure all the parts are connected correctly.

The input can either be connected to 12V or 0V. When the Inverter Input is 12V, the transistor in the first inverter will turn

on and the LED will come on and the Inverter Output voltage will be 0.2V. The first Inverter Output is connected to the input of the second inverter. The 0.2V at the input of the second inverter is small enough that the second transistor is turned off. The circuit voltages are shown in the diagram below.



When the Inverter Input is connected to 0V, the transistor in the first inverter is turned off and the LED will get very dim. There is a small amount of current still flowing through the LED to the second inverter. The voltage at the first Inverter Output will go up, forcing the second inverter transistor to come on. When the second inverter transistor comes on, the second inverter LED will come on. To find the voltage at the output of the first inverter (10.4V), use Ohm's law. There is no current flowing through the transistor in the first inverter so the path of the current is through the first LED, through the 2.2k resistor, through the 100k resistor, through the second transistor to ground. The voltage at the negative side of the first LED is fixed at 10.6V by the LED. The voltage at the second transistor base is fixed at 0.6V by the transistor. Then given those two voltages, you should be able to find the voltage at the point in the middle (10.4V) using Ohm's law. (Hint: First find the current and then work through Form 1 of ohm's law to find the voltage at the point between the 2.2k resistor and the 100k resistor.)



Switch the input back and forth from 0V to 12V and you can see that when the first stage is on, the second stage is off. This demonstrates the inverting action of the Inverter.

The next project in the series is called Pulses, Oscillators, Clocks...

It introduces capacitors and the LM555 timer. With these you can make circuits with LEDs that will continually flash!

[Click here to go to the Pulses, Oscillators, Clocks... page.](#)

For more details on digital logic, go to [Truth Table Page](#).

You can order the parts to build this circuit. The kit includes the parts for this project and the parts needed for the [555 project](#).

Beginner's Kit, \$19.00 (includes shipping):

- 5 - 330 ohm Resistors
- 5 - 510 ohm Resistors
- 5 - 2.2K ohm Resistors
- 5 - 10K ohm Resistors
- 5 - 100K ohm Resistors
- 1 - 220 uF Capacitor
- 5 - LEDs
- 5 - NPN Transistors
- 2 - 555 Timer ICs
- 1 - [Small Breadboard](#)

1 - 12 Volt DC Power Supply for Breadboard
Jumper Wires

To Order, go to the [Order Information Page](#) or call 800-297-1633. If you would like a printed copy of this tutorial (as well as our other other tutorials), remember to mark that on the order form. You may also be interested in the Microcontroller Beginner Kit which includes all the parts from this kit plus several additional projects. [Click here for more information](#).

Volume discounts and variations on the kit are also available (for example, if you already have a breadboard or power supply and don't want another). Write to support@iguanalabs.com for more information.

[Home Page](#)

Send Questions to support@iguanalabs.com

This page last updated on May 1, 2002.

Finding Voltage and Current Using Ohm's Law

There is a simple relationship between current, voltage and resistance. This relationship is called Ohm's Law. The formula is the following.

Difference in Voltage = Current * Resistance

or $DV = I * R$

This is Form 1 of Ohm's Law.

To find current and resistance the following forms can be used. They are the same as the above formula but in a different form.

Form 2: Current = Difference in Voltage / Resistance

or $I = DV / R$

Form 3: Resistance = Difference in Voltage / Current

or $R = DV / I$

These formulas are always used for situations where there are two points with a resistor between them. DV is the difference in voltage between the two points and current is what flows between the two points. These simple relationships allow us to calculate many things. Given any two of the three values (Current, Resistance, and Difference in Voltage) the third can be found. The most common calculation is for current. Voltage is easy to measure and the resistance can be found from the resistor (see [color codes](#)). Once these values are known, current can be calculated using Form 2 of Ohm's law, $I = DV / R$. For example, consider the problem shown in Figure 1. One side is at 0 volts (ground) and the other side is at 5 volts (with a multimeter, black probe on right side, red probe on left side).



Figure 1

The voltage difference between Point A and Point B is $5 - 0 = 5$ volts ($DV=5$). There is a resistor between the two points which has a value of 500 ohms ($R=500$). We know that current flows from a point of high voltage to a point of low voltage so we can draw an arrow from the higher voltage to the lower voltage.



Figure 2

Now we can find the current flowing through the resistor using Form 2 of Ohm's Law.

$$I = DV / R$$

$$DV / R = 5 / 500$$

$$5 / 500 = 0.01 \text{ Amps}$$

$$0.01 \text{ Amps} = 10 \text{ milliAmps}$$

10 milliamps can be abbreviated as 10 mA

This means the current is 10 mA. ($I = 10\text{mA}$)

Now to check our answer we can use Form 1 and Form 3 of Ohm's law. We have to use the value of current in Amps for these formulas. So if we have $I = 0.01$ Amps and Resistance = 500 ohms then by using Form 1 of Ohm's law we can find:

Difference in Voltage = DV

$$DV = I * R$$

$$I * R = 0.01 * 500$$

$$0.01 * 500 = 5 \text{ volts}$$

This is the voltage we started with so the value we found for the current must be right.

We can also check the answer with Form 3 by using $I = 0.01$ Amps and $DV = 5$ volts.

$$\text{Resistance} = R$$

$$R = DV / I$$

$$DV / I = 5 / 0.01$$

$$5 / 0.01 = 500 \text{ ohms.}$$

$$\text{So } R = 500 \text{ ohms}$$

Now consider the problem shown in Figure 3. The voltage on one side is 10 volts and the voltage on the other side is 3 volts. Therefore the voltage difference between the two points is $10 - 3 = 7$ volts ($DV = 7$ V). The resistor is 400 ohms ($R = 400$).



Figure 3

Then the current flowing from left to right is

$$I = DV / R$$

$$DV / R = 7 / 400$$

$$7 / 400 = 0.0175 \text{ Amps}$$

$$0.0175 \text{ Amps} = 17.5 \text{ milliAmps}$$

$$17.5 \text{ milliAmps} = 17.5 \text{ mA}$$

This means the current flowing from the left to the right is 17.5 mA.

Now suppose we have two points with a voltage difference of 5 volts. Point A is at 5 volts and Point B is at 0 volts (ground). (Notice that the voltage difference is the important part. If Point A is at 7 volts and Point B is at 2 volts then the voltage difference is the same, $7 - 2 = 5$ volts.) Now suppose we want a current to flow between Points A and B and we want the current to be 0.02 Amps ($I = 0.02$ Amps = 20 mA).

Now we need to find the value of the resistor so we use Form 3 of Ohm's Law.

Resistance = Difference in Voltage / Current or $R = DV / I$

$DV / I = 5 / 0.02 = 250$ ohms

This means that putting a resistor with a value of 250 ohms between Points A and B will make a current flow from Point A to Point B and the current will be 0.02 Amps (20 mA). Now using the values of voltage and resistance, check the value of the current using Form 2 of Ohm's law.

[Back to Tutorials Menu](#)

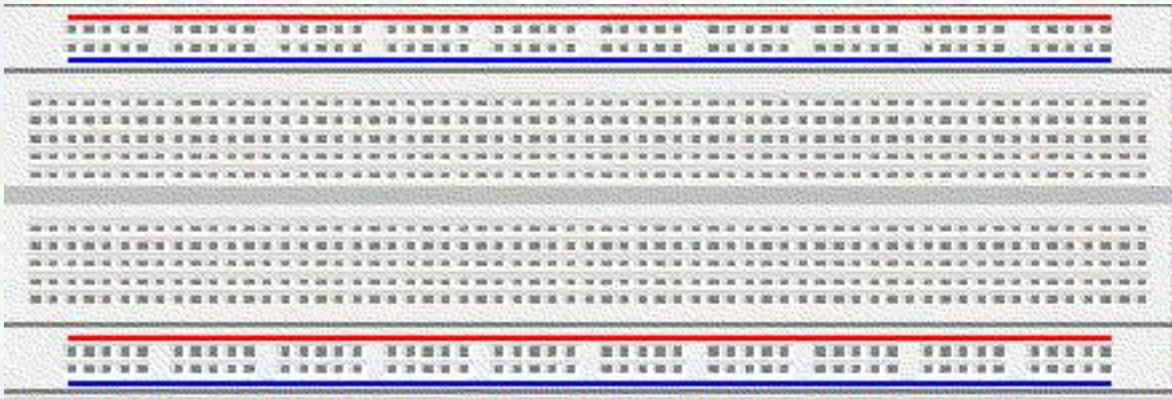
[Previous - Finding The Value of a Resistor By Reading It's Color Codes](#)

[Next - Using a Breadboard](#)

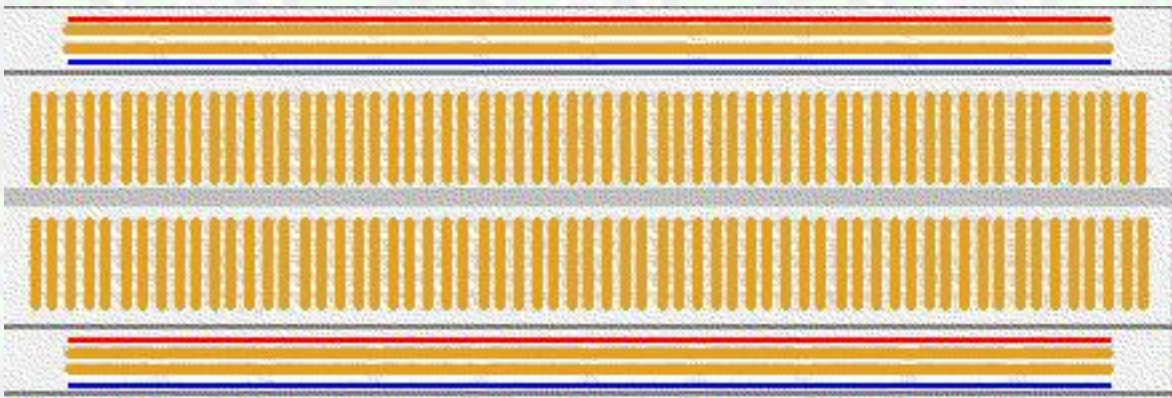
Iguana Labs

This page last updated on March 19, 2002.

Using the bread board (Socket Board)



The bread board has many strips of metal (copper usually) which run underneath the board. The metal strips are laid out as shown below.



These strips connect the holes on the top of the board. This makes it easy to connect components together to build circuits. To use the bread board, the legs of components are placed in the holes (the sockets). The holes are made so that they will hold the component in place. Each hole is connected to one of the metal strips running underneath the board.

Each wire forms a **node**. A node is a point in a circuit where two components are connected. Connections between different components are formed by putting their legs in a common node. On the

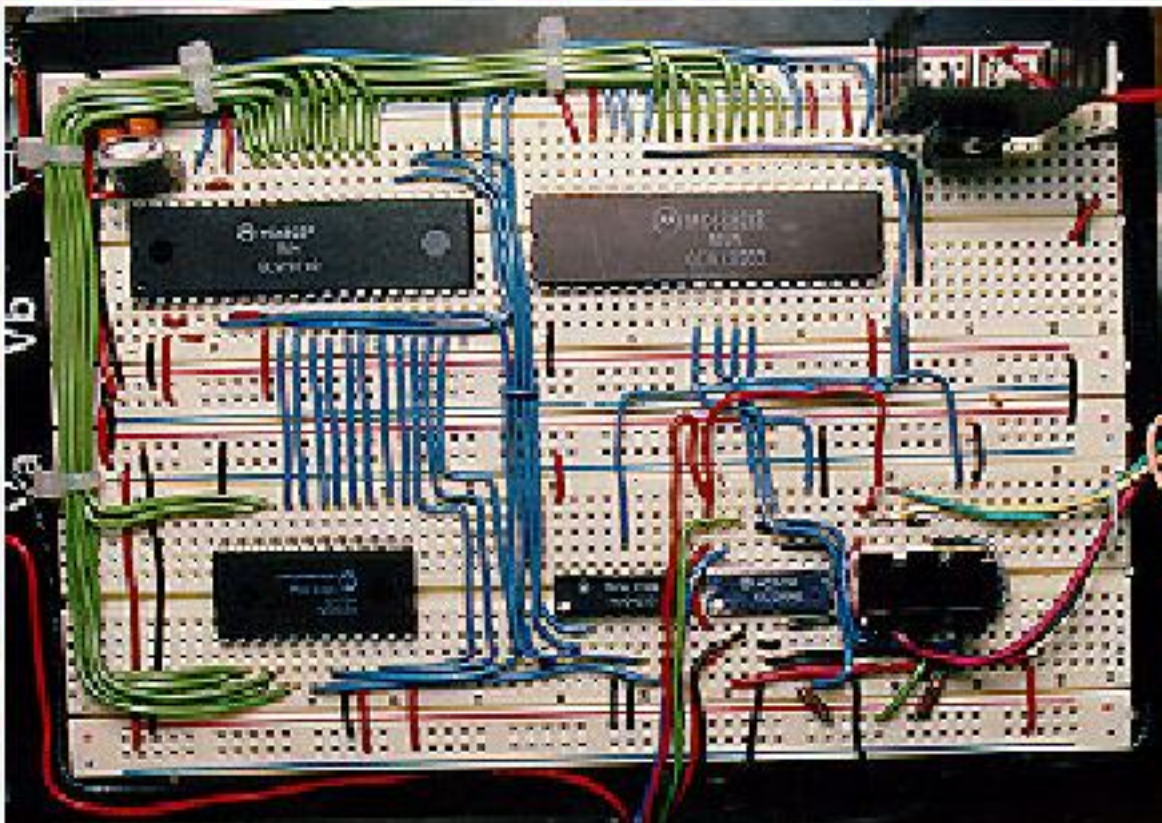
bread board, a node is the row of holes that are connected by the strip of metal underneath.

The long top and bottom row of holes are usually used for power supply connections.

The rest of the circuit is built by placing components and connecting them together with jumper wires. Then when a path is formed by wires and components from the positive supply node to the negative supply node, we can turn on the power and current flows through the path and the circuit comes alive.

For chips with many legs (ICs), place them in the middle of the board so that half of the legs are on one side of the middle line and half are on the other side.

A completed circuit might look like the following. This circuit uses two small breadboards.



We have two sizes of breadboards for sale. See [Order Form](#) for more information

[Back to Tutorials Menu](#)

[Previous - Using Ohm's Law](#)

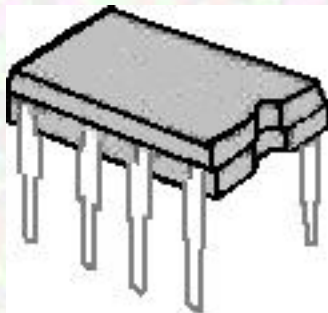
[Next - Using Transistors and LEDs](#)

Send Questions to support@iguanalabs.com

Iguana Labs

This page last updated on March 19, 2002.

Oscillators, Pulse Generators, Clocks... - Capacitors and the 555 Timer IC

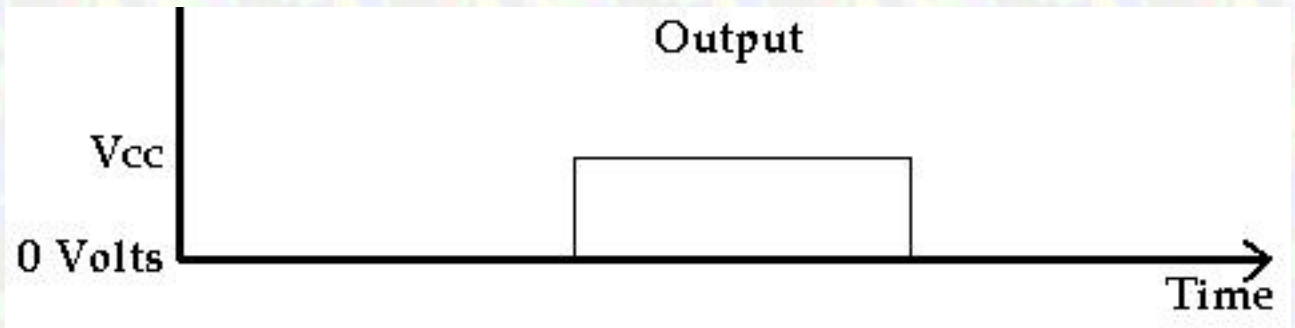


Introduction

As electronic designs get bigger, it becomes difficult to build the complete circuit. So we will use prebuilt circuits that come in packages like the one shown above. This prebuilt circuit is called an IC. IC stands for Integrated Circuit. An IC has many transistors inside it that are connected together to form a circuit. Metal pins are connected to the circuit and the circuit is stuck into a piece of plastic or ceramic so that the metal pins are sticking out of the side. These pins allow you to connect other devices to the circuit inside. We can buy simple ICs that have several inverter circuits like the one we built in the [LED and Transistor tutorial](#) or we can buy complex ICs like a Pentium Processor.

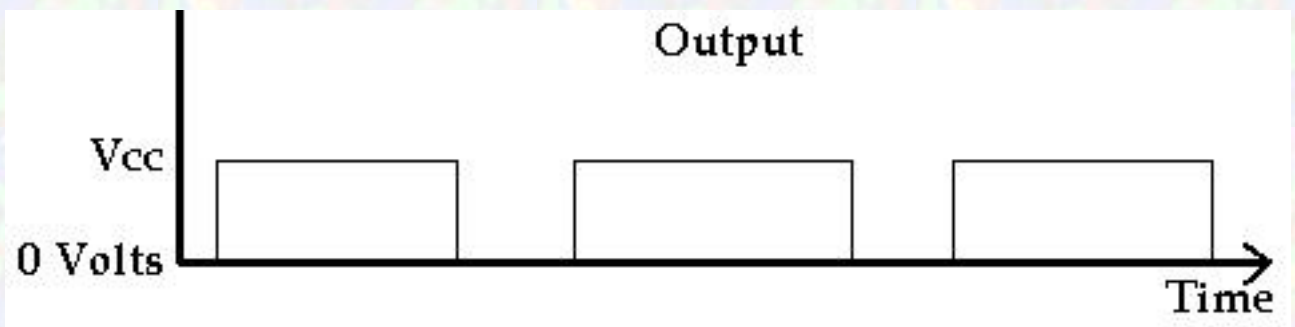
The Pulse - More than just an on/off switch

So far the circuits we have built have been stable, meaning that the output voltage stays the same. If you change the input voltage, the output voltage changes and once it changes it will stay at the same voltage level. The 555 integrated circuit (IC) is designed so that when the input changes, the output goes from 0 volts to V_{cc} (where V_{cc} is the voltage of the power supply). Then the output stays at V_{cc} for a certain length of time and then it goes back to 0 volts. This is a pulse. A graph of the output voltage is shown below.



The Oscillator (A Clock) - More than just a Pulse

The pulse is nice but it only happens one time. If you want something that does something interesting forever rather than just once, you need an oscillator. An oscillator puts out an endless series of pulses. The output constantly goes from 0 volts to V_{cc} and back to 0 volts again. Almost all digital circuits have some type of oscillator. This stream of output pulses is often called a clock. You can count the number of pulses to tell how much time has gone by. We will see how the 555 timer can be used to generate this clock. A graph of a clock signal is shown below.



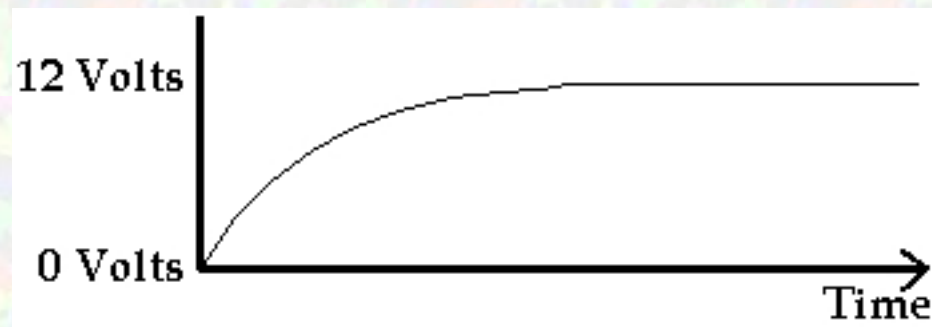
The Capacitor

If you already understand capacitors you can skip this part.



The picture above on the left shows two typical capacitors. Capacitors usually have two legs. One leg is the positive leg and the other is the negative leg. The positive leg is the one that is longer. The picture on the right is the symbol used for capacitors in circuit drawings (schematics). When you put one in a circuit, you must make sure the positive leg and the negative leg go in the right place. Capacitors do not always have a positive leg and a negative leg. The smallest capacitors in this kit do not. It does not matter which way you put them in a circuit.

A capacitor is similar to a rechargeable battery in the way it works. The difference is that a capacitor can only hold a small fraction of the energy that a battery can. (Except for really big capacitors like the ones found in old TVs. These can hold a lot of charge. Even if a TV has been disconnected from the wall for a long time, these capacitors can still make lots of sparks and hurt people.) As with a rechargeable battery, it takes a while for the capacitor to charge. So if we have a 12 volt supply and start charging the capacitor, it will start with 0 volts and go from 0 volts to 12 volts. Below is a graph of the voltage in the capacitor while it is charging.



The same idea is true when the capacitor is discharging. If the capacitor has been charged to 12 volts and then we connect both legs to ground, the capacitor will start discharging but it will take some time for the voltage to go to 0 volts. Below is a graph of what the voltage is in the capacitor while it is discharging.



We can control the speed of the capacitor's charging and discharging using resistors.

Capacitors are given values based on how much electricity they can store. Larger capacitors can store more energy and take more time to charge and discharge. The values are given in Farads but a Farad is a really large unit of measure for common capacitors. In this kit we have 2

33pf capacitors, 2 10uf capacitors and 2 220uF capacitors. Pf means picofarad and uf means microfarad. A picofarad is 0.000000000001 Farads. So the 33pf capacitor has a value of 33 picofarads or 0.000000000033 Farads. A microfarad is 0.000001 Farads. So the 10uf capacitor is 0.00001 Farads and the 220uF capacitor is 0.000220 Farads. If you do any calculations using the value of the capacitor you have to use the Farad value rather than the picofarad or microfarad value.

Capacitors are also rated by the maximum voltage they can take. This value is always written on the larger can shaped capacitors. For example, the 220uF capacitors in this kit have a maximum voltage rating of 25 volts. If you apply more than 25 volts to them they will die. We don't have to worry about that with this kit because our power supply can only put out 12 volts.

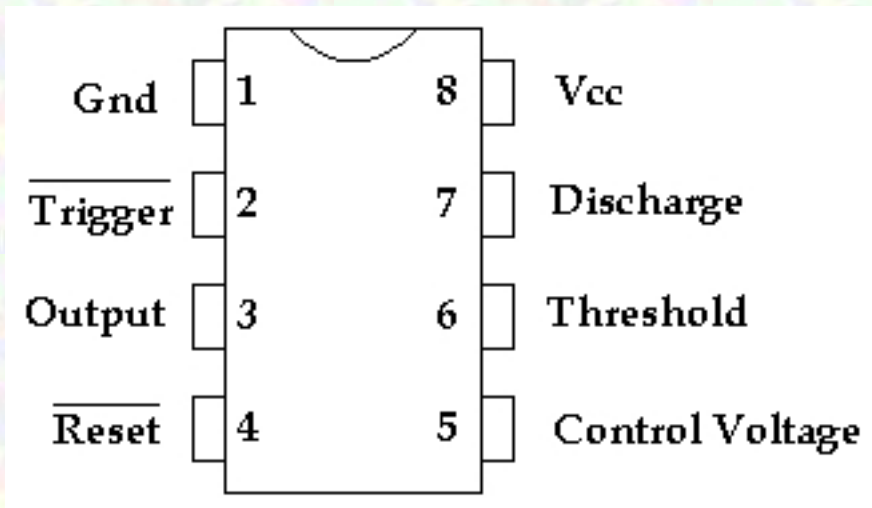
The 555 Timer

Creating a Pulse

The 555 is made out of simple transistors that are about the same as on / off switches. They do not have any sense of time. When you apply a voltage they turn on and when you take away the voltage they turn off. So by itself, the 555 can not create a pulse. The way the pulse is created is by using some components in a circuit attached to the 555 (see the circuit on the next page). This circuit is made of a capacitor and a resistor. We can flip a switch and start charging the capacitor. The resistor is used to control how fast the capacitor charges. The bigger the resistance, the longer it takes to charge the capacitor. The voltage in the capacitor can then be used as an input to another switch. Since the voltage starts at 0, nothing happens to the second switch. But eventually the capacitor will charge up to some point where the second switch comes on.

The way the 555 timer works is that when you flip the first switch, the Output pin goes to Vcc (the positive power supply voltage) and starts charging the capacitor. When the capacitor voltage gets to $\frac{2}{3} V_{cc}$ (that is $V_{cc} * \frac{2}{3}$) the second switch turns on which makes the output go to 0 volts.

The pinout for the 555 timer is shown below



Deep Details

Pin 2 (Trigger) is the 'on' switch for the pulse. The line over the word Trigger tells us that the voltage levels are the opposite of what you would normally expect. To turn the switch on you apply 0 volts to pin 2. The technical term for this opposite behavior is 'Active Low'. It is common to see this 'Active Low' behavior for IC inputs because of the inverting nature of transistor circuits like we saw in the LED and Transistor Tutorial.

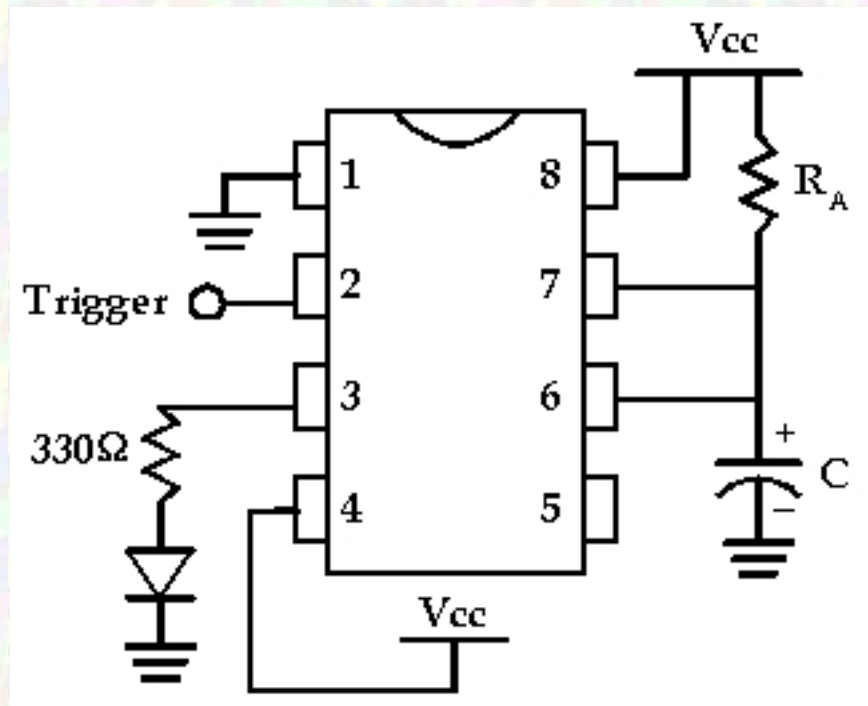
Pin 6 is the off switch for the pulse. We connect the positive side of the capacitor to this pin and the negative side of the capacitor to ground. When Pin 2 (Trigger) is at Vcc, the 555 holds Pin 7 at 0 volts (Note the inverted voltage). When Pin 2 goes to 0 volts, the 555 stops holding Pin 7 at 0 volts. Then the capacitor starts charging. The capacitor is charged through a resistor connected to Vcc. The current starts flowing into the capacitor, and the voltage in the capacitor starts to increase.

Pin 3 is the output (where the actual pulse comes out). The voltage on this pin starts at 0 volts. When 0 volts is applied to the trigger (Pin 2), the 555 puts out Vcc on Pin 3 and holds it at Vcc until Pin 6 reaches $\frac{2}{3}$ of Vcc (that is $V_{cc} * \frac{2}{3}$). Then the 555 pulls the voltage at Pin 3 to ground and you have created a pulse. (Again notice the inverting action.) The voltage on Pin 7 is also pulled to ground, connecting the capacitor to ground and discharging it.

Seeing the pulse

To see the pulse we will use an LED connected to the 555 output, Pin 3. When the output is 0 volts the LED will be off. When the output is Vcc the LED will be on.

Building the Circuit



Place the 555 across the middle line of the breadboard so that 4 pins are on one side and 4 pins are on the other side. (You may need to bend the pins in a little so they will go in the holes.) Leave the power disconnected until you finish building the circuit. The diagram above shows how the pins on the 555 are numbered. You can find pin 1 by looking for the half circle in the end of the chip. Sometimes instead of a half circle, there will be a dot or shallow hole by pin 1.

Before you start building the circuit, use jumper wires to connect the red and blue power rows to the red and blue power rows on the other side of the board. Then you will be able to easily reach Vcc and Ground lines from both sides of the board. (If the wires are too short, use two wires joined together in a row of holes for the positive power (Vcc) and two wires joined together in a different row of holes for the ground.)

Connect Pin 1 to ground.

Connect Pin 8 to Vcc.

Connect Pin 4 to Vcc.

Connect the positive leg of the LED to a 330 ohm resistor and connect the negative end of the LED to ground. Connect the other leg of the 330 ohm resistor to the output, Pin 3.

Connect Pin 7 to Vcc with a 10k resistor ($R_A = 10K$).

Connect Pin 7 to Pin 6 with a jumper wire.

Connect Pin 6 to the positive leg of the 220uF Capacitor ($C = 220\mu F$). (You will need to bend the positive (long leg) up and out some so that the negative leg can go in the breadboard.)

Connect the negative leg of the capacitor to ground.

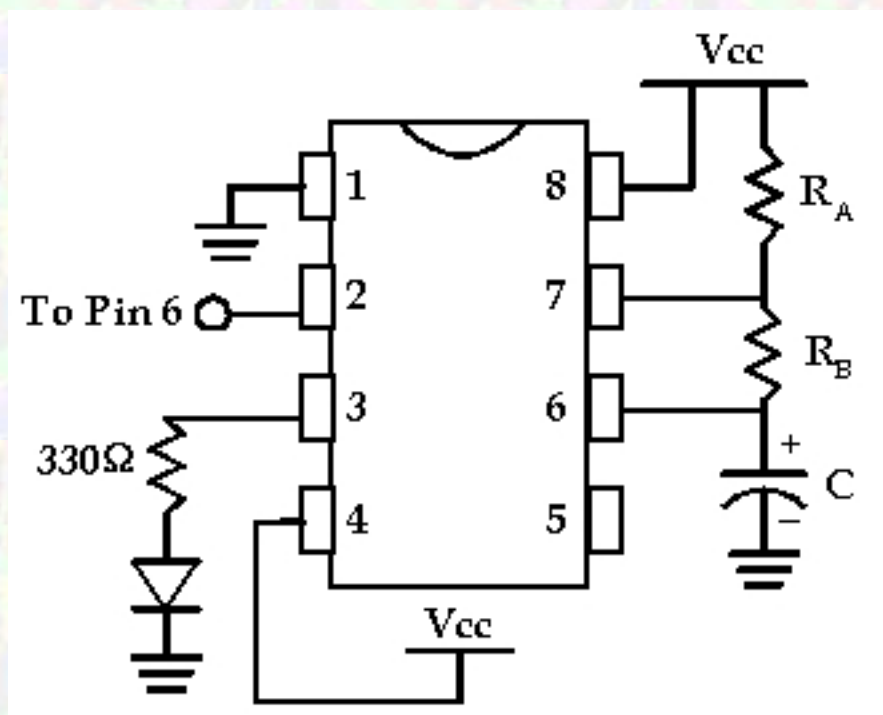
Connect a wire to Pin 2 to use as the trigger. Start with Pin 2 connected to Vcc.

Now connect the power. The LED will come on and stay on for about 2 seconds. Remove the wire connected to Pin 2 from Vcc. You should be able to trigger the 555 again by touching the wire connected to pin 2 with your finger or by connecting it to ground and removing it. (It should be about a 2 second pulse.)

Making it Oscillate

Next we will make the LED flash continually without having to trigger it. We will hook up the 555 so that it triggers itself. The way this works is that we add in a resistor between the capacitor and the discharge pin, Pin 7. Now, the capacitor will charge up (through RA and RB) and when it reaches $2/3 V_{cc}$, Pin 3 and Pin 7 will go to ground. But the capacitor can not discharge immediately because of RB. It takes some time for the charge to drain through RB. The more resistance RB has, the longer it takes to discharge. The time it takes to discharge the capacitor will be the time the LED is off.

To trigger the 555 again, we connect Pin 6 to the trigger (Pin 2). As the capacitor is discharging, the voltage in the capacitor gets lower and lower. When it gets down to $1/3 V_{cc}$ this triggers Pin 2 causing Pin 3 to go to Vcc and the LED to come on. The 555 disconnects Pin 7 from ground, and the capacitor starts to charge up again through RA and RB.



To build this circuit from the previous circuit, do the following.

Disconnect the power.

Take out the jumper wire between Pin 6 and Pin 7 and replace it with a 2.2k resistor ($R_B = 2.2K$).

Use the jumper wire at pin 2 to connect Pin 2 to Pin 6.

Now reconnect the power and the LED should flash forever (as long as you pay your electricity bill).

Experiment with different resistor values of R_A and R_B to see how it changes the length of time that the LED flashes. (You are changing the amount of time that it takes for the Capacitor to charge and discharge.)

Formulas

These are the formulas we use for the 555 to control the length of the pulses.

$t_1 = \text{charge time (how long the LED is on)} = 0.693 * (R_A + R_B) * C$

$t_2 = \text{discharge time (how long the LED is off)} = 0.693 * R_B * C$

$T = \text{period} = t_1 + t_2 = 0.693 * (R_A + 2 * R_B) * C$

$\text{Frequency} = 1 / T = 1.44 / ((R_A + 2 * R_B) * C)$

t_1 and t_2 are the time in seconds. C is the capacitor value in Farads. $220\mu\text{F} = 0.000220 \text{ F}$. So for our circuit we have:

$t_1 = 0.693 * (10000 + 2200) * 0.000220 = 1.86 \text{ seconds}$

$t_2 = 0.693 * 2200 * 0.000220 = 0.335 \text{ seconds}$

$T = 1.86 + 0.335 = 2.195 \text{ seconds}$

$\text{Frequency} = 0.456 \text{ (cycles per second)}$

All the parts in this kit are included with the [Beginners Kit](#) and the [Microcontroller Beginner Kit](#).

If you already have a breadboard and a power supply, and you just want the parts for this kit, order the 555Kit.

The 555Kit includes:



2 - 555 ICs
5 - 10K ohm Resistors
5 - 2.2K ohm Resistors
5 - 510 ohm Resistors
5 - 330 ohm Resistors
1 - 220 uF Capacitor
5 - LEDs
Jumper Wires

For Pricing, look for the 555Kit on the [Ordering Information Page](#) or call 800-297-1633.

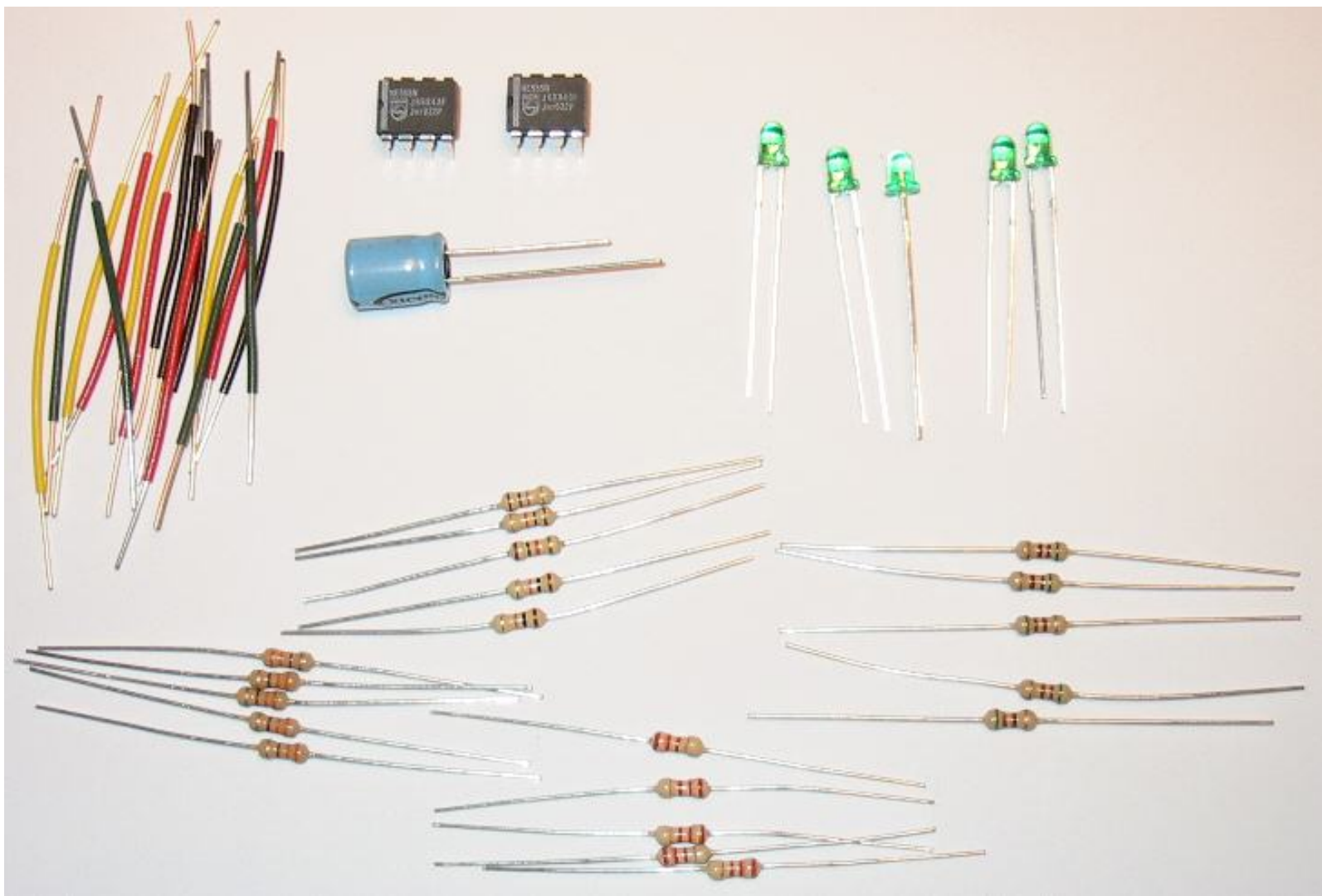
Volume discounts and variations on the kit are also available. Write to

support@iguanalabs.com for more information.

[Home Page](#)

Send Questions to tech@iguanalabs.com

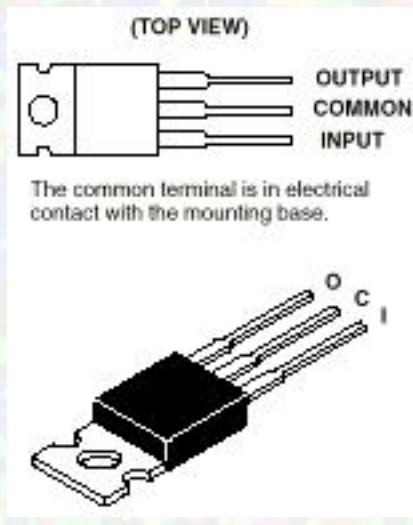
This page last updated on May 1, 2002.



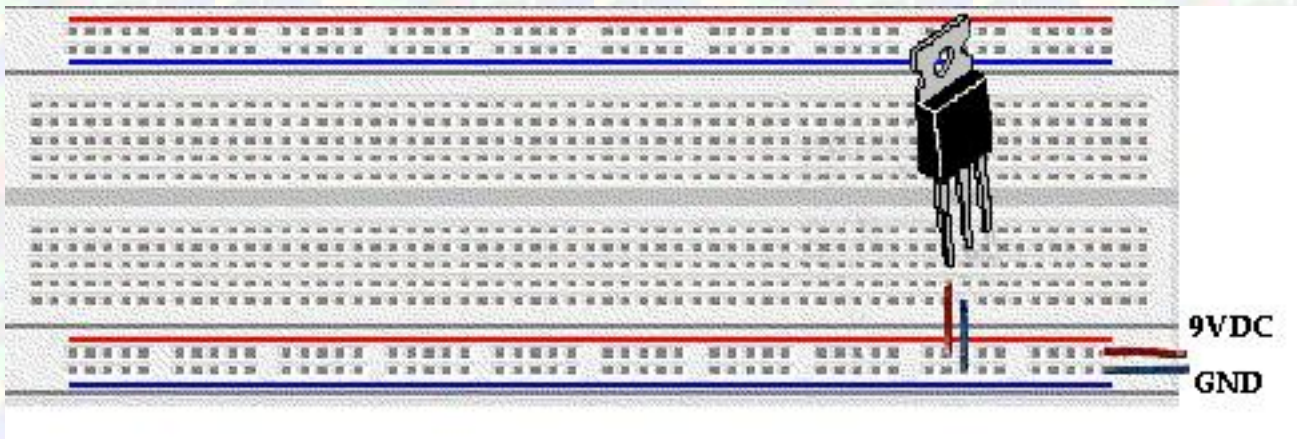
Building a 5 volt power supply

Most digital logic circuits and processors need a 5 volt power supply. To use these parts we need to build a regulated 5 volt source. Usually you start with an unregulated power supply ranging from 9 volts to 24 volts DC (A 9 volt power supply is included with the [first kit](#)). To make a 5 volt power supply, we use a LM7805 voltage regulator IC (Integrated Circuit).

The IC is shown below.

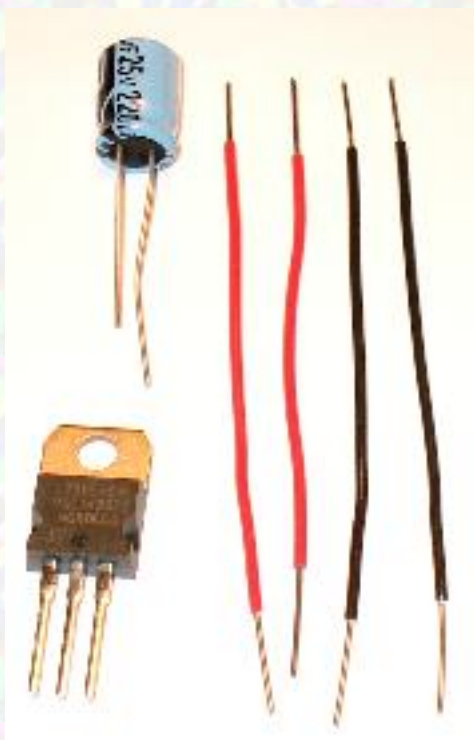


The LM7805 is simple to use. You simply connect the positive lead of your unregulated DC power supply (anything from 9VDC to 24VDC) to the Input pin, connect the negative lead to the Common pin and then when you turn on the power, you get a 5 volt supply from the Output pin. The breadboarded circuit is shown below.



Sometimes the input supply line (the 9VDC above) may be noisy. To help smooth out this noise and get a better 5 volt output, a capacitor is usually added to the circuit, going between 9VDC and ground (GND). We use a 220 uF capacitor.

You can order the parts for this kit. It includes:



- 1 - LM7805
- 1 - 220 uF Capacitor
- Jumper Wires

For Price and Ordering Information, look at 5voltKit on [Order Form](#). If you want a printed copy of this and the other tutorials, remember to mark that on the order

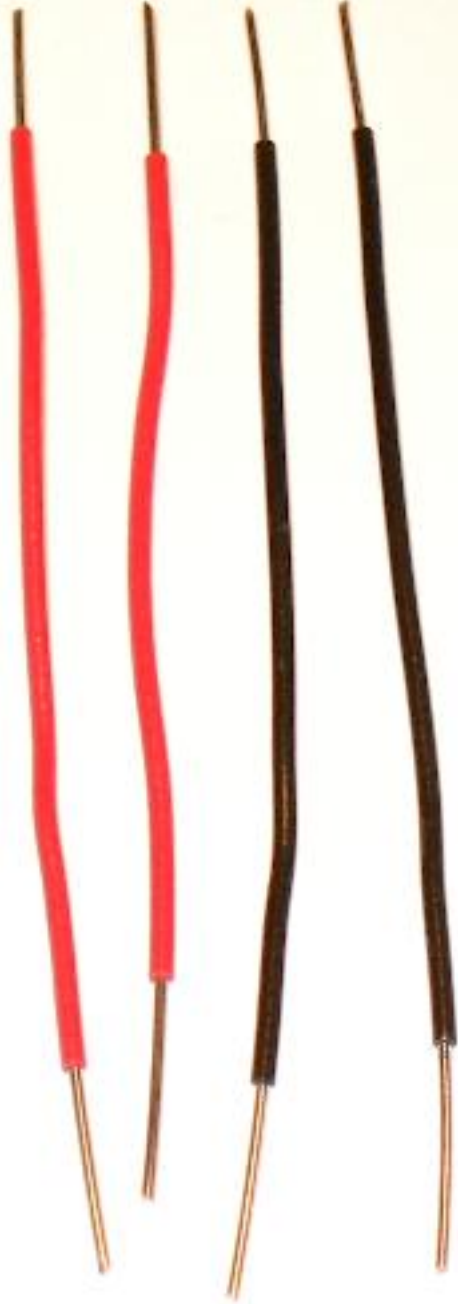
form.

[Back to Tutorials Menu](#)

Send Questions to tech@iguanalabs.com

Iguana Labs

This page last updated on April 16, 2002



Iguana Labs

800-297-1633 (US)

[PG302 Microcontroller Programmer](#)

[Experimenter/Controller Board](#)

[Tutorials for Electronics](#)

[8051 Assembly Language Library](#)

[Chip Pinouts](#)

[Assemblers and other free tools](#)

[Other 8051 web sites](#)

[Prototyping Supplies](#)

[Catalog](#)

[Online Secure Order Form](#)

[Support](#)

[Information Privacy Policy](#)

[Click here to send email to support@iguanalabs.com](mailto:support@iguanalabs.com)

Iguana Labs

This page last updated on June 18, 2002.

Using the Multimeter to Measure Voltage and Resistance

Multimeters are commonly used to measure voltage and resistance between two points. Current is rarely measured because you must alter the circuit to measure the current. Since we don't really want to alter the circuit, we measure voltage and resistance and calculate the current using Ohm's law ($I = DV / R$).

Voltage and resistance are always measured between two points.

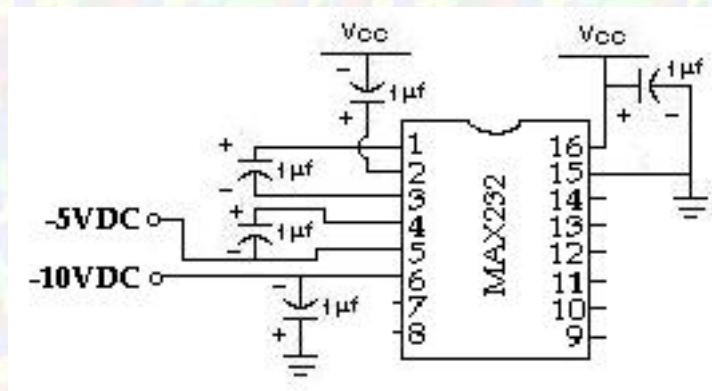
Voltage is the easiest and most common measurement. To measure voltage you set the multimeter to either DC or AC voltage and choose the range based on what you estimate the voltage to be. Then you touch the black (negative) probe to ground (the most negative point in the circuit) and then touch the red (positive) probe to a point in the circuit where you want to know the voltage (while keeping the black probe touching ground.) If you want to know the voltage difference (DV) between one side of a resistor and the other side you can simply put the black probe on one side and the red probe on the other. If the meter has an analog scale (a needle) and the needle goes the wrong way, reverse the red and black probes so the needle will go the other way and give you the voltage difference between those two points.

To measure resistance in a circuit, first turn off (or disconnect) the power supply. You may damage the multimeter if the circuit is still powered.) Next, select a range on the multimeter and touch two metal points in the circuit. If the needle doesn't move or goes all the way to the end of the scale, select another range. You can not use this method to measure the resistance of a resistor in the circuit because there may be other paths between the nodes of a resistor. One leg of a resistor must be disconnected from the circuit to make sure that the only path between the two probes is through that resistor. To measure the resistance of a resistor, select the range on the meter that you think is closest to the right value and use the probes to touch either side of the resistor. If you have selected the right range, the needle will be somewhere between the left and the right end of the scale. To find the value of the resistor, read the number from the scale that matches the range you are using.

Iguana Labs

-5 & -10 Volt DC Kit

Parts needed to build a simple negative power supply



Vcc = 5 Volts DC

The schematic above shows how to build a simple -5 Volt DC and -10 Volt DC power supply using a MAX232CPE chip and a 5 volt supply. A 5 volt supply is easy to build using a common DC wall adapter (9 to 20 volts DC or so) and a 7805 chip (as in our [5 Volt Kit](#)). The MAX232 chip is intended to be used for communicating with a PC through a serial port. (See the [Data Collection Kit](#) for more information on communicating with a PC.) A serial port requires a negative 10 volt signal to work properly so the most important thing that the MAX232 chip does is generate a -10 Volt power source for those signals. But negative voltages are also needed for other applications such as operational amplifiers (op-amps). The MAX232 chip can be used for this as long as the current requirements are not too high.

With no load, the outputs are about -5 Volts and -9.5 Volts. (The -10 Volt Source does not quite reach -10 Volts but it is still referred to as the -10 Volt Source.)

For the -10 Volt Source:

At 4 mA the voltage drops to about -8.5 volts.

At 10 mA, the voltage drops to about -6 volts.

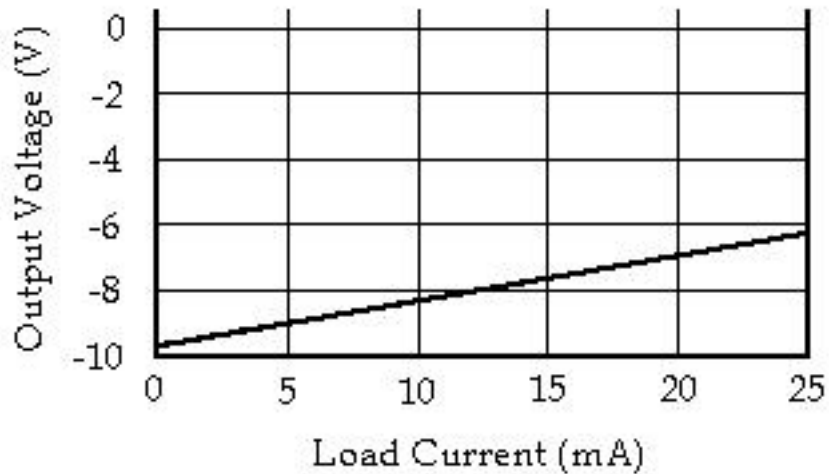
For the -5 Volt Source:

At 3.3 mA, the voltage drops to about -4.6 volts.

At 9 mA the voltage drops to about -3.5 volts.

The graph below shows how the -10 Volt Source drops as current increases.

OUTPUT VOLTAGE vs. LOAD CURRENT



Note: The -5 Volt output and the -10 Volt output are not independent. If you are using both outputs, the voltages will drop off faster.

Since the chip can not provide much current, one common way it is used is in a double opamp configuration. The first opamp acts as a buffer, inverting the signal to a negative signal then a second opamp reinverts the signal to positive and supplies the power, drawing current from the positive voltage source, V_{cc} .

For more information on the MAX232 chip [click here for the Data Sheet](#).

This kit includes:

- 1 - MAX232CPE
- 5 - 1 uF capacitors
- Jumper Wires

For Price and Ordering Information, look at -5&10Kit on the [Order Information Page \(click here\)](#).

Links

[Main Tutorial Page](#)

[5 Volt Kit](#)

[Data Collection Kit](#)

Microcontroller Beginner Kit

Send Questions to support@iguanalabs.com

Iguana Labs

This page last updated on June 20, 2001.



+5V-Powered, Multichannel RS-232 Drivers/Receivers

General Description

The MAX220–MAX249 family of line drivers/receivers is intended for all EIA/TIA-232E and V.28/V.24 communications interfaces, particularly applications where $\pm 12V$ is not available.

These parts are especially useful in battery-powered systems, since their low-power shutdown mode reduces power dissipation to less than $5\mu W$. The MAX225, MAX233, MAX235, and MAX245/MAX246/MAX247 use no external components and are recommended for applications where printed circuit board space is critical.

Applications

Portable Computers
Low-Power Modems
Interface Translation
Battery-Powered RS-232 Systems
Multidrop RS-232 Networks

Features

Superior to Bipolar

- ◆ Operate from Single +5V Power Supply (+5V and +12V—MAX231/MAX239)
- ◆ Low-Power Receive Mode in Shutdown (MAX223/MAX242)
- ◆ Meet All EIA/TIA-232E and V.28 Specifications
- ◆ Multiple Drivers and Receivers
- ◆ 3-State Driver and Receiver Outputs
- ◆ Open-Line Detection (MAX243)

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX220CPE	0°C to +70°C	16 Plastic DIP
MAX220CSE	0°C to +70°C	16 Narrow SO
MAX220CWE	0°C to +70°C	16 Wide SO
MAX220C/D	0°C to +70°C	Dice*
MAX220EPE	-40°C to +85°C	16 Plastic DIP
MAX220ESE	-40°C to +85°C	16 Narrow SO
MAX220EWE	-40°C to +85°C	16 Wide SO
MAX220EJE	-40°C to +85°C	16 CERDIP
MAX220MJE	-55°C to +125°C	16 CERDIP

Ordering Information continued at end of data sheet.

*Contact factory for dice specifications.

Selection Table

Part Number	Power Supply (V)	No. of RS-232 Drivers/Rx	No. of Ext. Caps	Nominal Cap. Value (μF)	SHDN & Three-State	Rx Active in SHDN	Data Rate (kbps)	Features
MAX220	+5	2/2	4	0.1	No	—	120	Ultra-low-power, industry-standard pinout
MAX222	+5	2/2	4	0.1	Yes	—	200	Low-power shutdown
MAX223 (MAX213)	+5	4/5	4	1.0 (0.1)	Yes	✓	120	MAX241 and receivers active in shutdown
MAX225	+5	5/5	0	—	Yes	✓	120	Available in SO
MAX230 (MAX200)	+5	5/0	4	1.0 (0.1)	Yes	—	120	5 drivers with shutdown
MAX231 (MAX201)	+5 and +7.5 to +13.2	2/2	2	1.0 (0.1)	No	—	120	Standard +5/+12V or battery supplies; same functions as MAX232
MAX232 (MAX202)	+5	2/2	4	1.0 (0.1)	No	—	120 (64)	Industry standard
MAX232A	+5	2/2	4	0.1	No	—	200	Higher slew rate, small caps
MAX233 (MAX203)	+5	2/2	0	—	No	—	120	No external caps
MAX233A	+5	2/2	0	—	No	—	200	No external caps, high slew rate
MAX234 (MAX204)	+5	4/0	4	1.0 (0.1)	No	—	120	Replaces 1488
MAX235 (MAX205)	+5	5/5	0	—	Yes	—	120	No external caps
MAX236 (MAX206)	+5	4/3	4	1.0 (0.1)	Yes	—	120	Shutdown, three state
MAX237 (MAX207)	+5	5/3	4	1.0 (0.1)	No	—	120	Complements IBM PC serial port
MAX238 (MAX208)	+5	4/4	4	1.0 (0.1)	No	—	120	Replaces 1488 and 1489
MAX239 (MAX209)	+5 and +7.5 to +13.2	3/5	2	1.0 (0.1)	No	—	120	Standard +5/+12V or battery supplies; single-package solution for IBM PC serial port
MAX240	+5	5/5	4	1.0	Yes	—	120	DIP or flatpack package
MAX241 (MAX211)	+5	4/5	4	1.0 (0.1)	Yes	—	120	Complete IBM PC serial port
MAX242	+5	2/2	4	0.1	Yes	✓	200	Separate shutdown and enable
MAX243	+5	2/2	4	0.1	No	—	200	Open-line detection simplifies cabling
MAX244	+5	8/10	4	1.0	No	—	120	High slew rate
MAX245	+5	8/10	0	—	Yes	✓	120	High slew rate, int. caps, two shutdown modes
MAX246	+5	8/10	0	—	Yes	✓	120	High slew rate, int. caps, three shutdown modes
MAX247	+5	8/9	0	—	Yes	✓	120	High slew rate, int. caps, nine operating modes
MAX248	+5	8/8	4	1.0	Yes	✓	120	High slew rate, selective half-chip enables
MAX249	+5	6/10	4	1.0	Yes	✓	120	Available in quad flatpack package



+5V-Powered, Multichannel RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS—MAX220/222/232A/233A/242/243

Supply Voltage (V _{CC})	-0.3V to +6V	20-Pin Plastic DIP (derate 8.00mW/°C above +70°C)	..440mW
Input Voltages		16-Pin Narrow SO (derate 8.70mW/°C above +70°C)	...696mW
T _{IN}	-0.3V to (V _{CC} - 0.3V)	16-Pin Wide SO (derate 9.52mW/°C above +70°C)762mW
R _{IN} (Except MAX220)	±30V	18-Pin Wide SO (derate 9.52mW/°C above +70°C)762mW
R _{IN} (MAX220)	±25V	20-Pin Wide SO (derate 10.00mW/°C above +70°C)	...800mW
T _{OUT} (Except MAX220) (Note 1)	±15V	20-Pin SSOP (derate 8.00mW/°C above +70°C)640mW
T _{OUT} (MAX220)	±13.2V	16-Pin CERDIP (derate 10.00mW/°C above +70°C)800mW
Output Voltages		18-Pin CERDIP (derate 10.53mW/°C above +70°C)842mW
T _{OUT}	±15V	Operating Temperature Ranges	
R _{OUT}	-0.3V to (V _{CC} + 0.3V)	MAX2_AC_, MAX2_C_0°C to +70°C
Driver/Receiver Output Short Circuited to GND	Continuous	MAX2_AE_, MAX2_E_-40°C to +85°C
Continuous Power Dissipation (T _A = +70°C)		MAX2_AM_, MAX2_M_-55°C to +125°C
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C)	...842mW	Storage Temperature Range-65°C to +160°C
18-Pin Plastic DIP (derate 11.11mW/°C above +70°C)	...889mW	Lead Temperature (soldering, 10sec)+300°C

Note 1: Input voltage measured with T_{OUT} in high-impedance state, $\overline{\text{SHDN}}$ or V_{CC} = 0V.

Note 2: For the MAX220, V+ and V- can have a maximum magnitude of 7V, but their absolute difference cannot exceed 13V.

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243

(V_{CC} = +5V ±10%, C1–C4 = 0.1µF, MAX220, C1 = 0.047µF, C2–C4 = 0.33µF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
RS-232 TRANSMITTERS						
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to GND		±5	±8		V
Input Logic Threshold Low				1.4	0.8	V
Input Logic Threshold High	All devices except MAX220		2	1.4		V
	MAX220: V _{CC} = 5.0V		2.4			
Logic Pull-Up/Input Current	All except MAX220, normal operation			5	40	µA
	$\overline{\text{SHDN}}$ = 0V, MAX222/242, shutdown, MAX220			±0.01	±1	
Output Leakage Current	V _{CC} = 5.5V, $\overline{\text{SHDN}}$ = 0V, V _{OUT} = ±15V, MAX222/242			±0.01	±10	µA
	V _{CC} = $\overline{\text{SHDN}}$ = 0V, V _{OUT} = ±15V			±0.01	±10	
Data Rate				200	116	kb/s
Transmitter Output Resistance	V _{CC} = V+ = V- = 0V, V _{OUT} = ±2V		300	10M		Ω
Output Short-Circuit Current	V _{OUT} = 0V		±7	±22		mA
RS-232 RECEIVERS						
RS-232 Input Voltage Operating Range					±30	V
RS-232 Input Threshold Low	V _{CC} = 5V	All except MAX243 R2 _{IN}	0.8	1.3		V
		MAX243 R2 _{IN} (Note 2)	-3			
RS-232 Input Threshold High	V _{CC} = 5V	All except MAX243 R2 _{IN}		1.8	2.4	V
		MAX243 R2 _{IN} (Note 2)		-0.5	-0.1	
RS-232 Input Hysteresis	All except MAX243, V _{CC} = 5V, no hysteresis in shdn.		0.2	0.5	1	V
	MAX243			1		
RS-232 Input Resistance			3	5	7	kΩ
TTL/CMOS Output Voltage Low	I _{OUT} = 3.2mA			0.2	0.4	V
TTL/CMOS Output Voltage High	I _{OUT} = -1.0mA		3.5	V _{CC} - 0.2		V
TTL/CMOS Output Short-Circuit Current	Sourcing V _{OUT} = GND		-2	-10		mA
	Shrinking V _{OUT} = V _{CC}		10	30		

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243 (continued)

(V_{CC} = +5V ±10%, C1–C4 = 0.1μF, MAX220, C1 = 0.047μF, C2–C4 = 0.33μF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

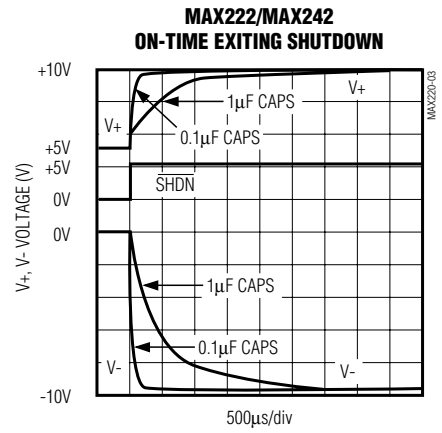
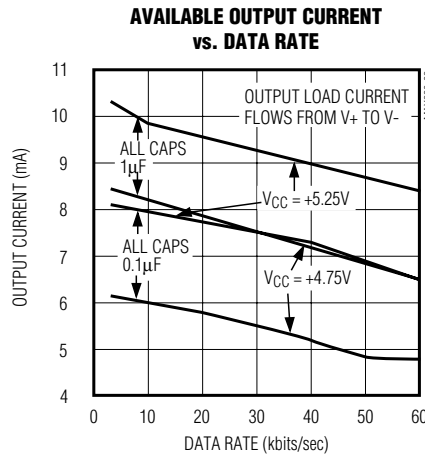
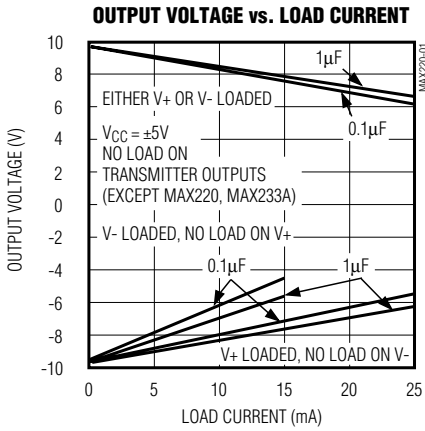
PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
TTL/CMOS Output Leakage Current	SHDN = V _{CC} or EN = V _{CC} (SHDN = 0V for MAX222), 0V ≤ V _{OUT} ≤ V _{CC}			±0.05	±10	μA
EN Input Threshold Low	MAX242			1.4	0.8	V
EN Input Threshold High	MAX242		2.0	1.4		V
Operating Supply Voltage			4.5		5.5	V
V _{CC} Supply Current (SHDN = V _{CC}), Figures 5, 6, 11, 19	No load	MAX220		0.5	2	mA
		MAX222/232A/233A/242/243		4	10	
	3kΩ load both inputs	MAX220		12		
		MAX222/232A/233A/242/243		15		
Shutdown Supply Current	MAX222/242	T _A = +25°C		0.1	10	μA
		T _A = 0°C to +70°C		2	50	
		T _A = -40°C to +85°C		2	50	
		T _A = -55°C to +125°C		35	100	
SHDN Input Leakage Current	MAX222/242				±1	μA
SHDN Threshold Low	MAX222/242			1.4	0.8	V
SHDN Threshold High	MAX222/242		2.0	1.4		V
Transition Slew Rate	C _L = 50pF to 2500pF, R _L = 3kΩ to 7kΩ, V _{CC} = 5V, T _A = +25°C, measured from +3V to -3V or -3V to +3V	MAX222/232A/233A/242/243	6	12	30	V/μs
		MAX220	1.5	3	30	
Transmitter Propagation Delay TLL to RS-232 (normal operation), Figure 1	t _{PHLT}	MAX222/232A/233A/242/243		1.3	3.5	μs
		MAX220		4	10	
	t _{PLHT}	MAX222/232A/233A/242/243		1.5	3.5	
		MAX220		5	10	
Receiver Propagation Delay RS-232 to TLL (normal operation), Figure 2	t _{PHLR}	MAX222/232A/233A/242/243		0.5	1	μs
		MAX220		0.6	3	
	t _{PLHR}	MAX222/232A/233A/242/243		0.6	1	
		MAX220		0.8	3	
Receiver Propagation Delay RS-232 to TLL (shutdown), Figure 2	t _{PHLS}	MAX242		0.5	10	μs
	t _{PLHS}	MAX242		2.5	10	
Receiver-Output Enable Time, Figure 3	t _{ER}	MAX242		125	500	ns
Receiver-Output Disable Time, Figure 3	t _{DR}	MAX242		160	500	ns
Transmitter-Output Enable Time (SHDN goes high), Figure 4	t _{ET}	MAX222/242, 0.1μF caps (includes charge-pump start-up)		250		μs
Transmitter-Output Disable Time (SHDN goes low), Figure 4	t _{DT}	MAX222/242, 0.1μF caps		600		ns
Transmitter + to - Propagation Delay Difference (normal operation)	t _{PHLT} - t _{PLHT}	MAX222/232A/233A/242/243		300		ns
		MAX220		2000		
Receiver + to - Propagation Delay Difference (normal operation)	t _{PHLR} - t _{PLHR}	MAX222/232A/233A/242/243		100		ns
		MAX220		225		

Note 3: MAX243 R_{2OUT} is guaranteed to be low when R_{2IN} is ≥ 0V or is floating.

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Typical Operating Characteristics

MAX220/MAX222/MAX232A/MAX233A/MAX242/MAX243



+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

ABSOLUTE MAXIMUM RATINGS—MAX223/MAX230–MAX241

V _{CC}	-0.3V to +6V	20-Pin Wide SO (derate 10.00mW/°C above +70°C)	800mW
V ₊	(V _{CC} - 0.3V) to +14V	24-Pin Wide SO (derate 11.76mW/°C above +70°C)	941mW
V ₋	+0.3V to -14V	28-Pin Wide SO (derate 12.50mW/°C above +70°C)	1W
Input Voltages		44-Pin Plastic FP (derate 11.11mW/°C above +70°C)	889mW
T _{IN}	-0.3V to (V _{CC} + 0.3V)	14-Pin CERDIP (derate 9.09mW/°C above +70°C)	727mW
R _{IN}	±30V	16-Pin CERDIP (derate 10.00mW/°C above +70°C)	800mW
Output Voltages		20-Pin CERDIP (derate 11.11mW/°C above +70°C)	889mW
T _{OUT}	(V ₊ + 0.3V) to (V ₋ - 0.3V)	24-Pin Narrow CERDIP	
R _{OUT}	-0.3V to (V _{CC} + 0.3V)	(derate 12.50mW/°C above +70°C)	1W
Short-Circuit Duration, T _{OUT}	Continuous	24-Pin Sidebrase (derate 20.0mW/°C above +70°C)	1.6W
Continuous Power Dissipation (T _A = +70°C)		28-Pin SSOP (derate 9.52mW/°C above +70°C)	762mW
14-Pin Plastic DIP (derate 10.00mW/°C above +70°C)		Operating Temperature Ranges	
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C)		MAX2 __ C	0°C to +70°C
20-Pin Plastic DIP (derate 11.11mW/°C above +70°C)		MAX2 __ E	-40°C to +85°C
24-Pin Narrow Plastic DIP		MAX2 __ M	-55°C to +125°C
(derate 13.33mW/°C above +70°C)		Storage Temperature Range	-65°C to +160°C
24-Pin Plastic DIP (derate 9.09mW/°C above +70°C)		Lead Temperature (soldering, 10sec)	+300°C
16-Pin Wide SO (derate 9.52mW/°C above +70°C)			

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX223/MAX230–MAX241

(MAX223/230/232/234/236/237/238/240/241, V_{CC} = +5V ±10%; MAX233/MAX235, V_{CC} = 5V ±5%, C1–C4 = 1.0μF; MAX231/MAX239, V_{CC} = 5V ±10%; V₊ = 7.5V to 13.2V; T_A = T_{MIN} to T_{MAX}; unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to ground		±5.0	±7.3		V
V _{CC} Power-Supply Current	No load, T _A = +25°C	MAX232/233		5	10	mA
		MAX223/230/234–238/240/241		7	15	
		MAX231/239		0.4	1	
V ₊ Power-Supply Current		MAX231		1.8	5	mA
		MAX239		5	15	
Shutdown Supply Current	T _A = +25°C	MAX223		15	50	μA
		MAX230/235/236/240/241		1	10	
Input Logic Threshold Low	T _{IN} ; EN, $\overline{\text{SHDN}}$ (MAX233); $\overline{\text{EN}}$, SHDN (MAX230/235–241)				0.8	V
Input Logic Threshold High	T _{IN}		2.0			V
	EN, $\overline{\text{SHDN}}$ (MAX223); $\overline{\text{EN}}$, SHDN (MAX230/235/236/240/241)		2.4			
Logic Pull-Up Current	T _{IN} = 0V			1.5	200	μA
Receiver Input Voltage Operating Range			-30		30	V

+5V-Powered, Multichannel RS-232 Drivers/Receivers

ELECTRICAL CHARACTERISTICS—MAX223/MAX230–MAX241 (continued)

(MAX223/230/232/234/236/237/238/240/241, $V_{CC} = +5V \pm 10\%$; MAX233/MAX235, $V_{CC} = 5V \pm 5\%$, $C_1-C_4 = 1.0\mu F$; MAX231/MAX239, $V_{CC} = 5V \pm 10\%$; $V_+ = 7.5V$ to $13.2V$; $T_A = T_{MIN}$ to T_{MAX} ; unless otherwise noted.)

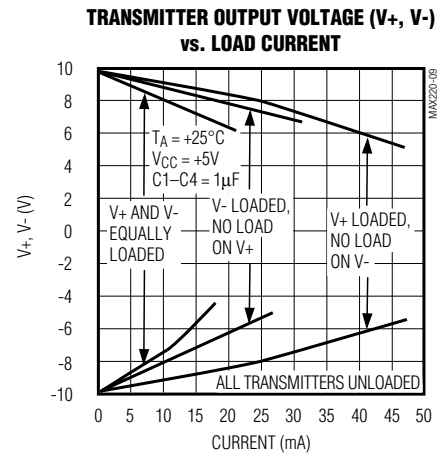
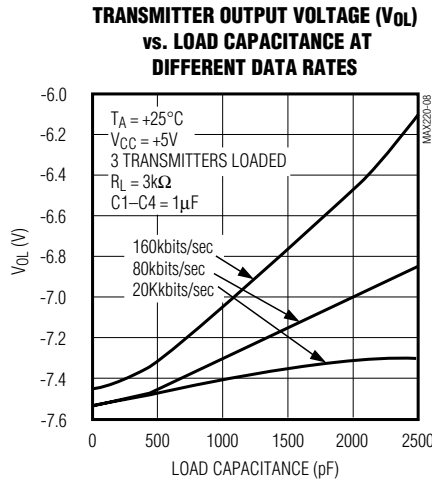
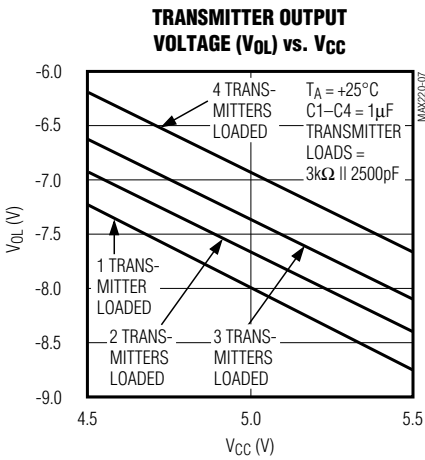
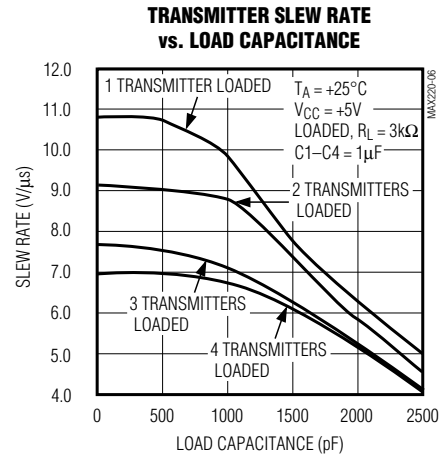
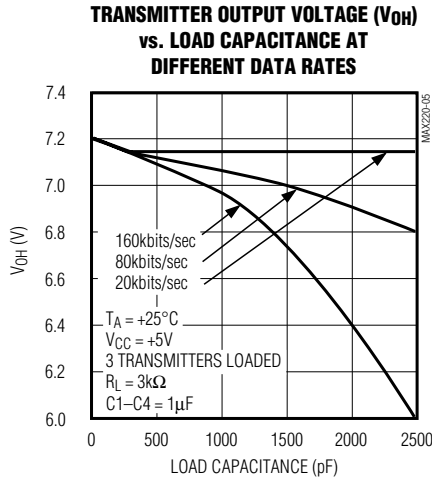
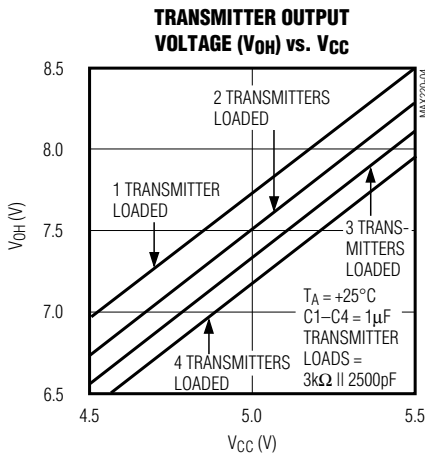
PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS	
RS-232 Input Threshold Low	$T_A = +25^\circ C$, $V_{CC} = 5V$	Normal operation $\overline{SHDN} = 5V$ (MAX223) $SHDN = 0V$ (MAX235/236/240/241)	0.8	1.2		V	
		Shutdown (MAX223) $\overline{SHDN} = 0V$, $EN = 5V$ (R_{4IN} , R_{5IN})	0.6	1.5			
RS-232 Input Threshold High	$T_A = +25^\circ C$, $V_{CC} = 5V$	Normal operation $\overline{SHDN} = 5V$ (MAX223) $SHDN = 0V$ (MAX235/236/240/241)		1.7	2.4	V	
		Shutdown (MAX223) $\overline{SHDN} = 0V$, $EN = 5V$ (R_{4IN} , R_{5IN})		1.5	2.4		
RS-232 Input Hysteresis	$V_{CC} = 5V$, no hysteresis in shutdown		0.2	0.5	1.0	V	
RS-232 Input Resistance	$T_A = +25^\circ C$, $V_{CC} = 5V$		3	5	7	$k\Omega$	
TTL/CMOS Output Voltage Low	$I_{OUT} = 1.6mA$ (MAX231/232/233, $I_{OUT} = 3.2mA$)				0.4	V	
TTL/CMOS Output Voltage High	$I_{OUT} = -1mA$		3.5	$V_{CC} - 0.4$		V	
TTL/CMOS Output Leakage Current	$0V \leq R_{OUT} \leq V_{CC}$; $EN = 0V$ (MAX223); $\overline{EN} = V_{CC}$ (MAX235–241)			0.05	± 10	μA	
Receiver Output Enable Time	Normal operation	MAX223		600		ns	
		MAX235/236/239/240/241		400			
Receiver Output Disable Time	Normal operation	MAX223		900		ns	
		MAX235/236/239/240/241		250			
Propagation Delay	RS-232 IN to TTL/CMOS OUT, $C_L = 150pF$	Normal operation		0.5	10	μs	
		$\overline{SHDN} = 0V$ (MAX223)	t_{PHLS}		4		40
			t_{PLHS}		6		40
Transition Region Slew Rate	MAX223/MAX230/MAX234–241, $T_A = +25^\circ C$, $V_{CC} = 5V$, $R_L = 3k\Omega$ to $7k\Omega$, $C_L = 50pF$ to $2500pF$, measured from $+3V$ to $-3V$ or $-3V$ to $+3V$		3	5.1	30	V/ μs	
	MAX231/MAX232/MAX233, $T_A = +25^\circ C$, $V_{CC} = 5V$, $R_L = 3k\Omega$ to $7k\Omega$, $C_L = 50pF$ to $2500pF$, measured from $+3V$ to $-3V$ or $-3V$ to $+3V$			4	30		
Transmitter Output Resistance	$V_{CC} = V_+ = V_- = 0V$, $V_{OUT} = \pm 2V$		300			Ω	
Transmitter Output Short-Circuit Current			± 10			mA	

+5V-Powered, Multichannel RS-232 Drivers/Receivers

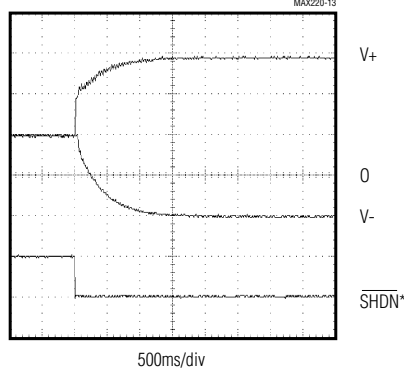
Typical Operating Characteristics

MAX220-MAX249

MAX223/MAX230-MAX241



V_+ , V_- WHEN EXITING SHUTDOWN (1µF CAPACITORS)



*SHUTDOWN POLARITY IS REVERSED FOR NON MAX241 PARTS

+5V-Powered, Multichannel RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS—MAX225/MAX244-MAX249

Supply Voltage (V_{CC})	-0.3V to +6V	Continuous Power Dissipation ($T_A = +70^\circ\text{C}$)	
Input Voltages		28-Pin Wide SO (derate 12.50mW/ $^\circ\text{C}$ above +70 $^\circ\text{C}$)	1W
T_{IN} , \overline{ENA} , \overline{ENB} , \overline{ENR} , \overline{ENT} , \overline{ENRA} , \overline{ENRB} , \overline{ENTA} , \overline{ENTB}	-0.3V to ($V_{CC} + 0.3\text{V}$)	40-Pin Plastic DIP (derate 11.11mW/ $^\circ\text{C}$ above +70 $^\circ\text{C}$)	0.611W
R_{IN}	$\pm 25\text{V}$	44-Pin PLCC (derate 13.33mW/ $^\circ\text{C}$ above +70 $^\circ\text{C}$)	1.07W
T_{OUT} (Note 3)	$\pm 15\text{V}$	Operating Temperature Ranges	
R_{OUT}	-0.3V to ($V_{CC} + 0.3\text{V}$)	MAX225C_-, MAX24_C_-	0 $^\circ\text{C}$ to +70 $^\circ\text{C}$
Short Circuit (one output at a time)		MAX225E_-, MAX24_E_-	-40 $^\circ\text{C}$ to +85 $^\circ\text{C}$
T_{OUT} to GND	Continuous	Storage Temperature Range	-65 $^\circ\text{C}$ to +160 $^\circ\text{C}$
R_{OUT} to GND	Continuous	Lead Temperature (soldering, 10sec)	+300 $^\circ\text{C}$

Note 4: Input voltage measured with transmitter output in a high-impedance state, shutdown, or $V_{CC} = 0\text{V}$.

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX225/MAX244-MAX249

(MAX225, $V_{CC} = 5.0\text{V} \pm 5\%$; MAX244-MAX249, $V_{CC} = +5.0\text{V} \pm 10\%$, external capacitors C1-C4 = 1 μF ; $T_A = T_{MIN}$ to T_{MAX} ; unless otherwise noted.)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	
RS-232 TRANSMITTERS						
Input Logic Threshold Low			1.4	0.8	V	
Input Logic Threshold High		2	1.4		V	
Logic Pull-Up/Input Current	Tables 1a-1d	Normal operation		10	50	μA
		Shutdown		± 0.01	± 1	
Data Rate	Tables 1a-1d, normal operation		120	64	kbits/sec	
Output Voltage Swing	All transmitter outputs loaded with 3k Ω to GND	± 5	± 7.5		V	
Output Leakage Current (shutdown)	Tables 1a-1d	\overline{ENA} , \overline{ENB} , \overline{ENT} , \overline{ENTA} , $\overline{ENTB} = V_{CC}$, $V_{OUT} = \pm 15\text{V}$		± 0.01	± 25	μA
		$V_{CC} = 0\text{V}$, $V_{OUT} = \pm 15\text{V}$		± 0.01	± 25	
Transmitter Output Resistance	$V_{CC} = V_+ = V_- = 0\text{V}$, $V_{OUT} = \pm 2\text{V}$ (Note 4)	300	10M		Ω	
Output Short-Circuit Current	$V_{OUT} = 0\text{V}$	± 7	± 30		mA	
RS-232 RECEIVERS						
RS-232 Input Voltage Operating Range				± 25	V	
RS-232 Input Threshold Low	$V_{CC} = 5\text{V}$	0.8	1.3		V	
RS-232 Input Threshold High	$V_{CC} = 5\text{V}$		1.8	2.4	V	
RS-232 Input Hysteresis	$V_{CC} = 5\text{V}$	0.2	0.5	1.0	V	
RS-232 Input Resistance		3	5	7	k Ω	
TTL/CMOS Output Voltage Low	$I_{OUT} = 3.2\text{mA}$		0.2	0.4	V	
TTL/CMOS Output Voltage High	$I_{OUT} = -1.0\text{mA}$	3.5	$V_{CC} - 0.2$		V	
TTL/CMOS Output Short-Circuit Current	Sourcing $V_{OUT} = \text{GND}$	-2	-10		mA	
	Shrinking $V_{OUT} = V_{CC}$	10	30			
TTL/CMOS Output Leakage Current	Normal operation, outputs disabled, Tables 1a-1d, $0\text{V} \leq V_{OUT} \leq V_{CC}$, $\overline{ENR}_- = V_{CC}$		± 0.05	± 0.10	μA	

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

ELECTRICAL CHARACTERISTICS—MAX225/MAX244–MAX249 (continued)

(MAX225, $V_{CC} = 5.0V \pm 5\%$; MAX244–MAX249, $V_{CC} = +5.0V \pm 10\%$, external capacitors C1–C4 = 1 μ F; $T_A = T_{MIN}$ to T_{MAX} ; unless otherwise noted.)

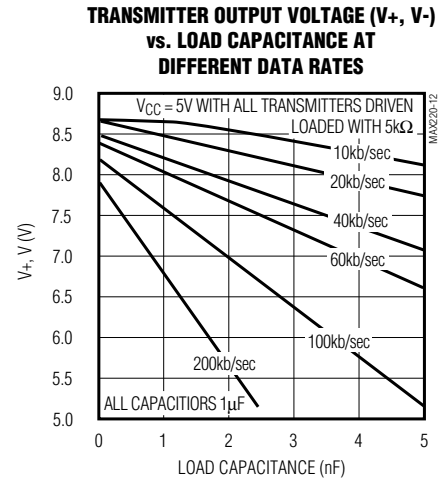
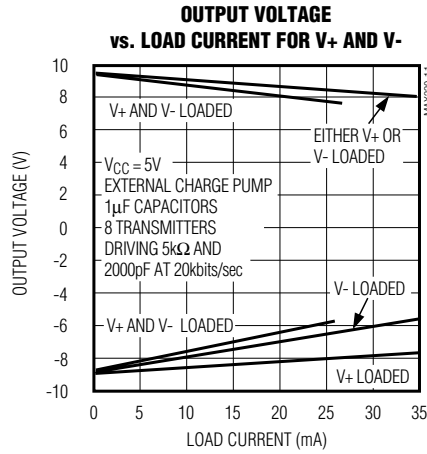
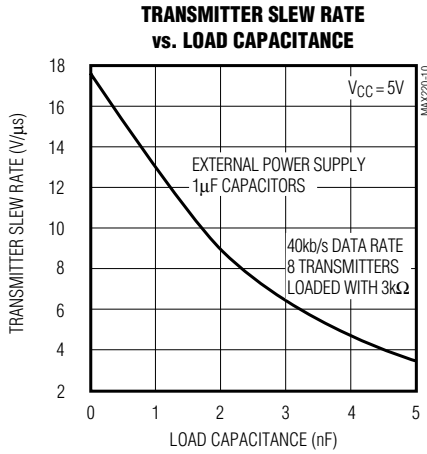
PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
POWER SUPPLY AND CONTROL LOGIC						
Operating Supply Voltage		MAX225	4.75		5.25	V
		MAX244–MAX249	4.5		5.5	
V_{CC} Supply Current (normal operation)	No load	MAX225		10	20	mA
		MAX244–MAX249		11	30	
	3k Ω loads on all outputs	MAX225		40		
		MAX244–MAX249		57		
Shutdown Supply Current	$T_A = +25^\circ\text{C}$			8	25	μ A
	$T_A = T_{MIN}$ to T_{MAX}				50	
Control Input	Leakage current				± 1	μ A
	Threshold low			1.4	0.8	V
	Threshold high		2.4	1.4		
AC CHARACTERISTICS						
Transition Slew Rate	$C_L = 50\text{pF}$ to 2500pF, $R_L = 3\text{k}\Omega$ to 7k Ω , $V_{CC} = 5V$, $T_A = +25^\circ\text{C}$, measured from +3V to -3V or -3V to +3V		5	10	30	V/ μ s
Transmitter Propagation Delay TLL to RS-232 (normal operation), Figure 1	t_{PHLT}			1.3	3.5	μ s
	t_{PLHT}			1.5	3.5	
Receiver Propagation Delay TLL to RS-232 (normal operation), Figure 2	t_{PHLR}			0.6	1.5	μ s
	t_{PLHR}			0.6	1.5	
Receiver Propagation Delay TLL to RS-232 (low-power mode), Figure 2	t_{PHLS}			0.6	10	μ s
	t_{PLHS}			3.0	10	
Transmitter + to - Propagation Delay Difference (normal operation)	$t_{PHLT} - t_{PLHT}$			350		ns
Receiver + to - Propagation Delay Difference (normal operation)	$t_{PHLR} - t_{PLHR}$			350		ns
Receiver-Output Enable Time, Figure 3	t_{ER}			100	500	ns
Receiver-Output Disable Time, Figure 3	t_{DR}			100	500	ns
Transmitter Enable Time	t_{ET}	MAX246–MAX249 (excludes charge-pump start-up)		5		μ s
		MAX225/MAX245–MAX249 (includes charge-pump start-up)		10		ms
Transmitter Disable Time, Figure 4	t_{DT}			100		ns

Note 5: The 300 Ω minimum specification complies with EIA/TIA-232E, but the actual resistance when in shutdown mode or $V_{CC} = 0V$ is 10M Ω as is implied by the leakage specification.

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Typical Operating Characteristics

MAX225/MAX244-MAX249



+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

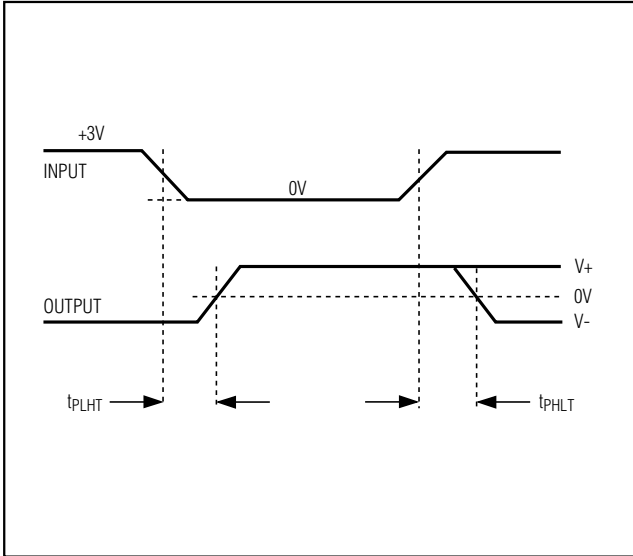


Figure 1. Transmitter Propagation-Delay Timing

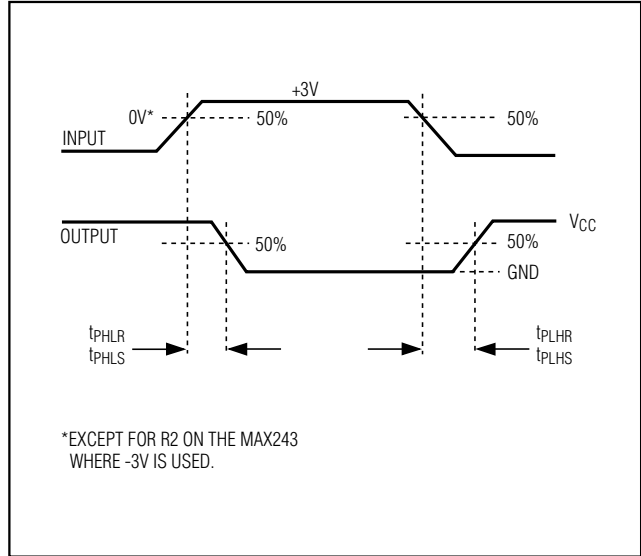


Figure 2. Receiver Propagation-Delay Timing

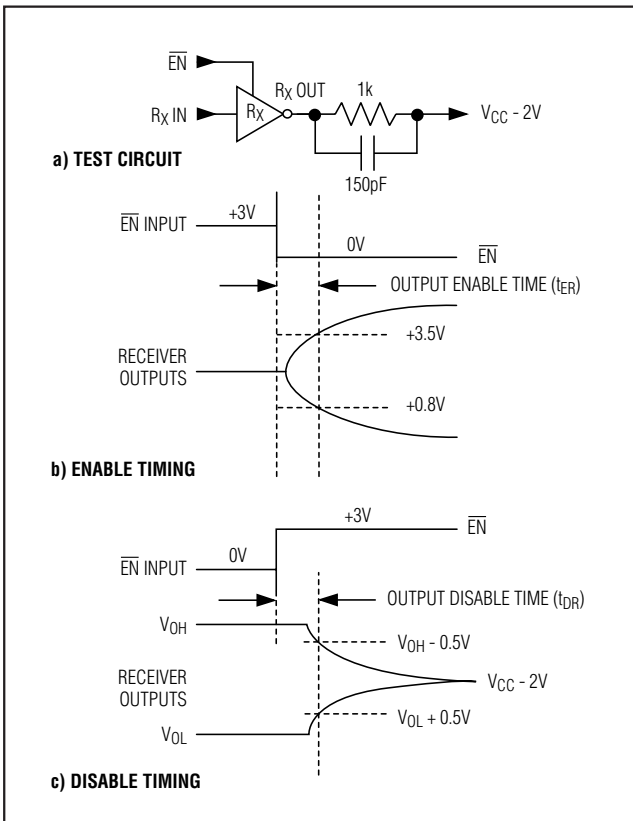


Figure 3. Receiver-Output Enable and Disable Timing

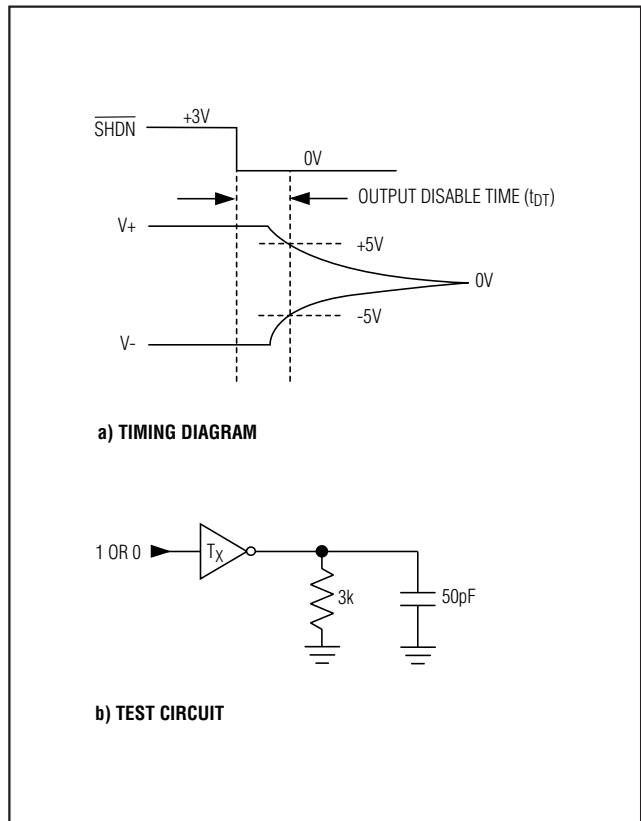


Figure 4. Transmitter-Output Disable Timing

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Table 1a. MAX245 Control Pin Configurations

$\overline{\text{ENT}}$	$\overline{\text{ENR}}$	OPERATION STATUS	TRANSMITTERS	RECEIVERS
0	0	Normal Operation	All Active	All Active
0	1	Normal Operation	All Active	All 3-State
1	0	Shutdown	All 3-State	All Low-Power Receive Mode
1	1	Shutdown	All 3-State	All 3-State

Table 1b. MAX245 Control Pin Configurations

$\overline{\text{ENT}}$	$\overline{\text{ENR}}$	OPERATION STATUS	TRANSMITTERS		RECEIVERS	
			TA1-TA4	TB1-TB4	RA1-RA5	RB1-RB5
0	0	Normal Operation	All Active	All Active	All Active	All Active
0	1	Normal Operation	All Active	All Active	RA1-RA4 3-State, RA5 Active	RB1-RB4 3-State, RB5 Active
1	0	Shutdown	All 3-State	All 3-State	All Low-Power Receive Mode	All Low-Power Receive Mode
1	1	Shutdown	All 3-State	All 3-State	RA1-RA4 3-State, RA5 Low-Power Receive Mode	RB1-RB4 3-State, RB5 Low-Power Receive Mode

Table 1c. MAX246 Control Pin Configurations

$\overline{\text{ENA}}$	$\overline{\text{ENB}}$	OPERATION STATUS	TRANSMITTERS		RECEIVERS	
			TA1-TA4	TB1-TB4	RA1-RA5	RB1-RB5
0	0	Normal Operation	All Active	All Active	All Active	All Active
0	1	Normal Operation	All Active	All 3-State	All Active	RB1-RB4 3-State, RB5 Active
1	0	Shutdown	All 3-State	All Active	RA1-RA4 3-State, RA5 Active	All Active
1	1	Shutdown	All 3-State	All 3-State	RA1-RA4 3-State, RA5 Low-Power Receive Mode	RB1-RB4 3-State, RA5 Low-Power Receive Mode

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

Table 1d. MAX247/MAX248/MAX249 Control Pin Configurations

$\overline{\text{ENTA}}$	$\overline{\text{ENTB}}$	$\overline{\text{ENRA}}$	$\overline{\text{ENRB}}$	OPERATION STATUS	TRANSMITTERS			RECEIVERS	
					MAX247	TA1-TA4	TB1-TB4	RA1-RA4	RB1-RB5
					MAX248	TA1-TA4	TB1-TB4	RA1-RA4	RB1-RB4
					MAX249	TA1-TA3	TB1-TB3	RA1-RA5	RB1-RB5
0	0	0	0	Normal Operation		All Active	All Active	All Active	All Active
0	0	0	1	Normal Operation		All Active	All Active	All Active	All 3-State, except RB5 stays active on MAX247
0	0	1	0	Normal Operation		All Active	All Active	All 3-State	All Active
0	0	1	1	Normal Operation		All Active	All Active	All 3-State	All 3-State, except RB5 stays active on MAX247
0	1	0	0	Normal Operation		All Active	All 3-State	All Active	All Active
0	1	0	1	Normal Operation		All Active	All 3-State	All Active	All 3-State, except RB5 stays active on MAX247
0	1	1	0	Normal Operation		All Active	All 3-State	All 3-State	All Active
0	1	1	1	Normal Operation		All Active	All 3-State	All 3-State	All 3-State, except RB5 stays active on MAX247
1	0	0	0	Normal Operation		All 3-State	All Active	All Active	All Active
1	0	0	1	Normal Operation		All 3-State	All Active	All Active	All 3-State, except RB5 stays active on MAX247
1	0	1	0	Normal Operation		All 3-State	All Active	All 3-State	All Active
1	0	1	1	Normal Operation		All 3-State	All Active	All 3-State	All 3-State, except RB5 stays active on MAX247
1	1	0	0	Shutdown		All 3-State	All 3-State	Low-Power Receive Mode	Low-Power Receive Mode
1	1	0	1	Shutdown		All 3-State	All 3-State	Low-Power Receive Mode	All 3-State, except RB5 stays active on MAX247
1	1	1	0	Shutdown		All 3-State	All 3-State	All 3-State	Low-Power Receive Mode
1	1	1	1	Shutdown		All 3-State	All 3-State	All 3-State	All 3-State, except RB5 stays active on MAX247

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Detailed Description

The MAX220–MAX249 contain four sections: dual charge-pump DC-DC voltage converters, RS-232 drivers, RS-232 receivers, and receiver and transmitter enable control inputs.

Dual Charge-Pump Voltage Converter

The MAX220–MAX249 have two internal charge-pumps that convert +5V to $\pm 10V$ (unloaded) for RS-232 driver operation. The first converter uses capacitor C1 to double the +5V input to +10V on C3 at the V+ output. The second converter uses capacitor C2 to invert +10V to -10V on C4 at the V- output.

A small amount of power may be drawn from the +10V (V+) and -10V (V-) outputs to power external circuitry (see the *Typical Operating Characteristics* section), except on the MAX225 and MAX245–MAX247, where these pins are not available. V+ and V- are not regulated, so the output voltage drops with increasing load current. Do not load V+ and V- to a point that violates the minimum $\pm 5V$ EIA/TIA-232E driver output voltage when sourcing current from V+ and V- to external circuitry.

When using the shutdown feature in the MAX222, MAX225, MAX230, MAX235, MAX236, MAX240, MAX241, and MAX245–MAX249, avoid using V+ and V- to power external circuitry. When these parts are shut down, V- falls to 0V, and V+ falls to +5V. For applications where a +10V external supply is applied to the V+ pin (instead of using the internal charge pump to generate +10V), the C1 capacitor must not be installed and the SHDN pin must be tied to VCC. This is because V+ is internally connected to VCC in shutdown mode.

RS-232 Drivers

The typical driver output voltage swing is $\pm 8V$ when loaded with a nominal $5k\Omega$ RS-232 receiver and VCC = +5V. Output swing is guaranteed to meet the EIA/TIA-232E and V.28 specification, which calls for $\pm 5V$ minimum driver output levels under worst-case conditions. These include a minimum $3k\Omega$ load, VCC = +4.5V, and maximum operating temperature. Unloaded driver output voltage ranges from (V+ -1.3V) to (V- +0.5V).

Input thresholds are both TTL and CMOS compatible. The inputs of unused drivers can be left unconnected since $400k\Omega$ input pull-up resistors to VCC are built in (except for the MAX220). The pull-up resistors force the outputs of unused drivers low because all drivers invert. The internal input pull-up resistors typically source $12\mu A$, except in shutdown mode where the pull-ups are disabled. Driver outputs turn off and enter a high-impedance state—where leakage current is typically microamperes (maximum $25\mu A$)—when in shutdown

mode, in three-state mode, or when device power is removed. Outputs can be driven to $\pm 15V$. The power-supply current typically drops to $8\mu A$ in shutdown mode. The MAX220 does not have pull-up resistors to force the outputs of the unused drivers low. Connect unused inputs to GND or VCC.

The MAX239 has a receiver three-state control line, and the MAX223, MAX225, MAX235, MAX236, MAX240, and MAX241 have both a receiver three-state control line and a low-power shutdown control. Table 2 shows the effects of the shutdown control and receiver three-state control on the receiver outputs.

The receiver TTL/CMOS outputs are in a high-impedance, three-state mode whenever the three-state enable line is high (for the MAX225/MAX235/MAX236/MAX239–MAX241), and are also high-impedance whenever the shutdown control line is high.

When in low-power shutdown mode, the driver outputs are turned off and their leakage current is less than $1\mu A$ with the driver output pulled to ground. The driver output leakage remains less than $1\mu A$, even if the transmitter output is backdriven between 0V and (VCC + 6V). Below -0.5V, the transmitter is diode clamped to ground with $1k\Omega$ series impedance. The transmitter is also zener clamped to approximately VCC + 6V, with a series impedance of $1k\Omega$.

The driver output slew rate is limited to less than $30V/\mu s$ as required by the EIA/TIA-232E and V.28 specifications. Typical slew rates are $24V/\mu s$ unloaded and $10V/\mu s$ loaded with 3Ω and $2500pF$.

RS-232 Receivers

EIA/TIA-232E and V.28 specifications define a voltage level greater than 3V as a logic 0, so all receivers invert. Input thresholds are set at 0.8V and 2.4V, so receivers respond to TTL level inputs as well as EIA/TIA-232E and V.28 levels.

The receiver inputs withstand an input overvoltage up to $\pm 25V$ and provide input terminating resistors with

Table 2. Three-State Control of Receivers

PART	SHDN	SHDN	EN	EN(R)	RECEIVERS
MAX223	—	Low High High	X Low High	—	High Impedance Active High Impedance
MAX225	—	—	—	Low High	High Impedance Active
MAX235 MAX236 MAX240	Low Low High	—	—	Low High X	High Impedance Active High Impedance

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

nominal 5k Ω values. The receivers implement Type 1 interpretation of the fault conditions of V.28 and EIA/TIA-232E.

The receiver input hysteresis is typically 0.5V with a guaranteed minimum of 0.2V. This produces clear output transitions with slow-moving input signals, even with moderate amounts of noise and ringing. The receiver propagation delay is typically 600ns and is independent of input swing direction.

Low-Power Receive Mode

The low-power receive-mode feature of the MAX223, MAX242, and MAX245-MAX249 puts the IC into shutdown mode but still allows it to receive information. This is important for applications where systems are periodically awakened to look for activity. Using low-power receive mode, the system can still receive a signal that will activate it on command and prepare it for communication at faster data rates. This operation conserves system power.

Negative Threshold—MAX243

The MAX243 is pin compatible with the MAX232A, differing only in that RS-232 cable fault protection is removed on one of the two receiver inputs. This means that control lines such as CTS and RTS can either be driven or left floating without interrupting communication. Different cables are not needed to interface with different pieces of equipment.

The input threshold of the receiver without cable fault protection is -0.8V rather than +1.4V. Its output goes positive only if the input is connected to a control line that is actively driven negative. If not driven, it defaults to the 0 or "OK to send" state. Normally, the MAX243's other receiver (+1.4V threshold) is used for the data line (TD or RD), while the negative threshold receiver is connected to the control line (DTR, DTS, CTS, RTS, etc.).

Other members of the RS-232 family implement the optional cable fault protection as specified by EIA/TIA-232E specifications. This means a receiver output goes high whenever its input is driven negative, left floating, or shorted to ground. The high output tells the serial communications IC to stop sending data. To avoid this, the control lines must either be driven or connected with jumpers to an appropriate positive voltage level.

Shutdown—MAX222-MAX242

On the MAX222, MAX235, MAX236, MAX240, and MAX241, all receivers are disabled during shutdown. On the MAX223 and MAX242, two receivers continue to operate in a reduced power mode when the chip is in shutdown. Under these conditions, the propagation delay increases to about 2.5 μ s for a high-to-low input transition. When in shutdown, the receiver acts as a CMOS inverter with no hysteresis. The MAX223 and MAX242 also have a receiver output enable input (\overline{EN} for the MAX242 and EN for the MAX223) that allows receiver output control independent of \overline{SHDN} (SHDN for MAX241). With all other devices, \overline{SHDN} (SHDN for MAX241) also disables the receiver outputs.

The MAX225 provides five transmitters and five receivers, while the MAX245 provides ten receivers and eight transmitters. Both devices have separate receiver and transmitter-enable controls. The charge pumps turn off and the devices shut down when a logic high is applied to the ENT input. In this state, the supply current drops to less than 25 μ A and the receivers continue to operate in a low-power receive mode. Driver outputs enter a high-impedance state (three-state mode). On the MAX225, all five receivers are controlled by the \overline{ENR} input. On the MAX245, eight of the receiver outputs are controlled by the \overline{ENR} input, while the remaining two receivers (RA5 and RB5) are always active. RA1-RA4 and RB1-RB4 are put in a three-state mode when \overline{ENR} is a logic high.

Receiver and Transmitter Enable Control Inputs

The MAX225 and MAX245-MAX249 feature transmitter and receiver enable controls.

The receivers have three modes of operation: full-speed receive (normal active), three-state (disabled), and low-power receive (enabled receivers continue to function at lower data rates). The receiver enable inputs control the full-speed receive and three-state modes. The transmitters have two modes of operation: full-speed transmit (normal active) and three-state (disabled). The transmitter enable inputs also control the shutdown mode. The device enters shutdown mode when all transmitters are disabled. Enabled receivers function in the low-power receive mode when in shutdown.

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Tables 1a–1d define the control states. The MAX244 has no control pins and is not included in these tables.

The MAX246 has ten receivers and eight drivers with two control pins, each controlling one side of the device. A logic high at the A-side control input ($\overline{\text{ENA}}$) causes the four A-side receivers and drivers to go into a three-state mode. Similarly, the B-side control input ($\overline{\text{ENB}}$) causes the four B-side drivers and receivers to go into a three-state mode. As in the MAX245, one A-side and one B-side receiver (RA5 and RB5) remain active at all times. The entire device is put into shutdown mode when both the A and B sides are disabled ($\text{ENA} = \text{ENB} = +5\text{V}$).

The MAX247 provides nine receivers and eight drivers with four control pins. The $\overline{\text{ENRA}}$ and $\overline{\text{ENRB}}$ receiver enable inputs each control four receiver outputs. The $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$ transmitter enable inputs each control four drivers. The ninth receiver (RB5) is always active. The device enters shutdown mode with a logic high on both $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$.

The MAX248 provides eight receivers and eight drivers with four control pins. The $\overline{\text{ENRA}}$ and $\overline{\text{ENRB}}$ receiver enable inputs each control four receiver outputs. The $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$ transmitter enable inputs control four drivers each. This part does not have an always-active receiver. The device enters shutdown mode and transmitters go into a three-state mode with a logic high on both $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$.

The MAX249 provides ten receivers and six drivers with four control pins. The $\overline{\text{ENRA}}$ and $\overline{\text{ENRB}}$ receiver enable inputs each control five receiver outputs. The $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$ transmitter enable inputs control three drivers each. There is no always-active receiver. The device enters shutdown mode and transmitters go into a three-state mode with a logic high on both $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$. In shutdown mode, active receivers operate in a low-power receive mode at data rates up to 20kbits/sec.

Applications Information

Figures 5 through 25 show pin configurations and typical operating circuits. In applications that are sensitive to power-supply noise, VCC should be decoupled to ground with a capacitor of the same value as C1 and C2 connected as close as possible to the device.

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

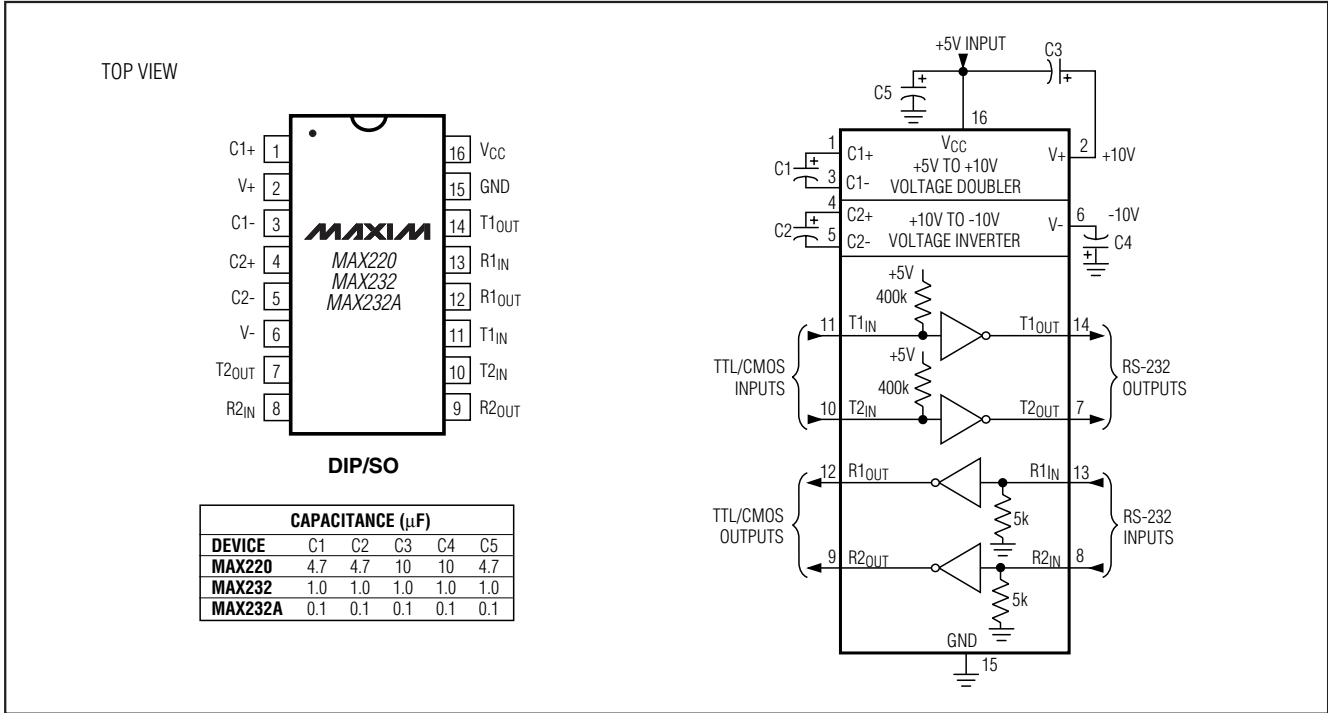


Figure 5. MAX220/MAX232/MAX232A Pin Configuration and Typical Operating Circuit

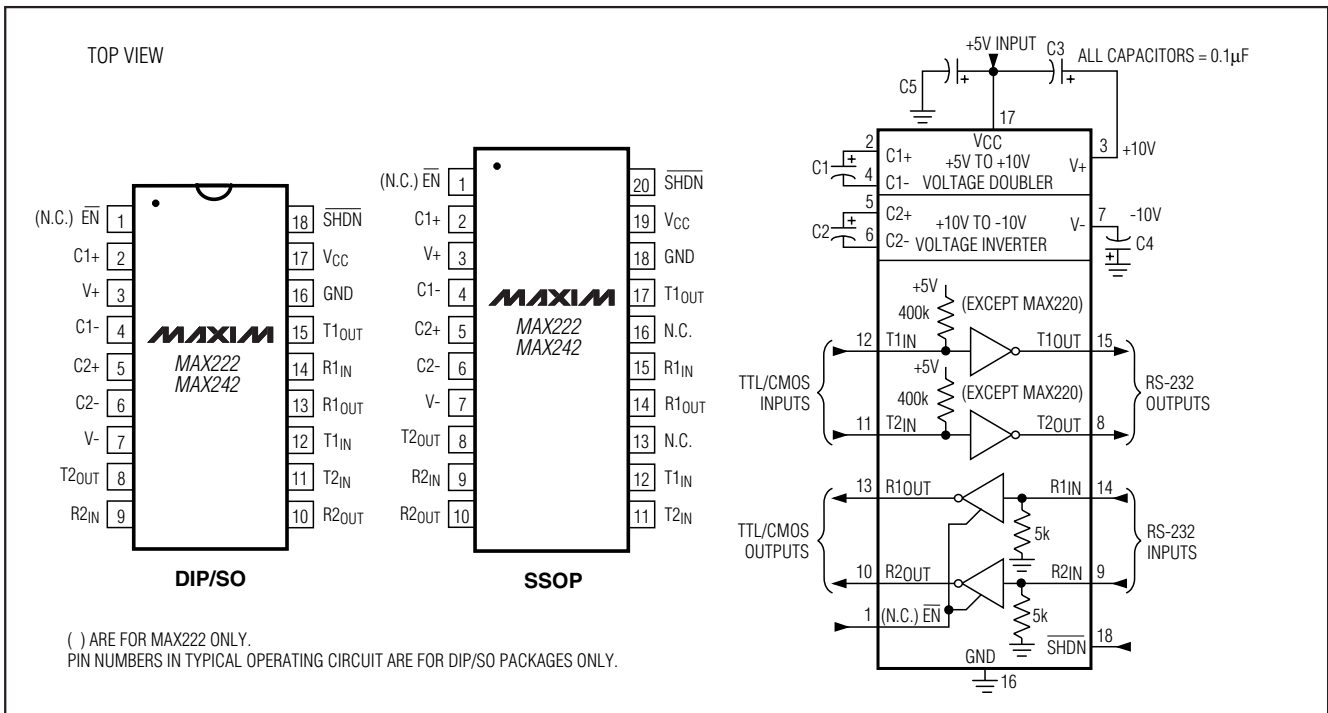


Figure 6. MAX222/MAX242 Pin Configurations and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

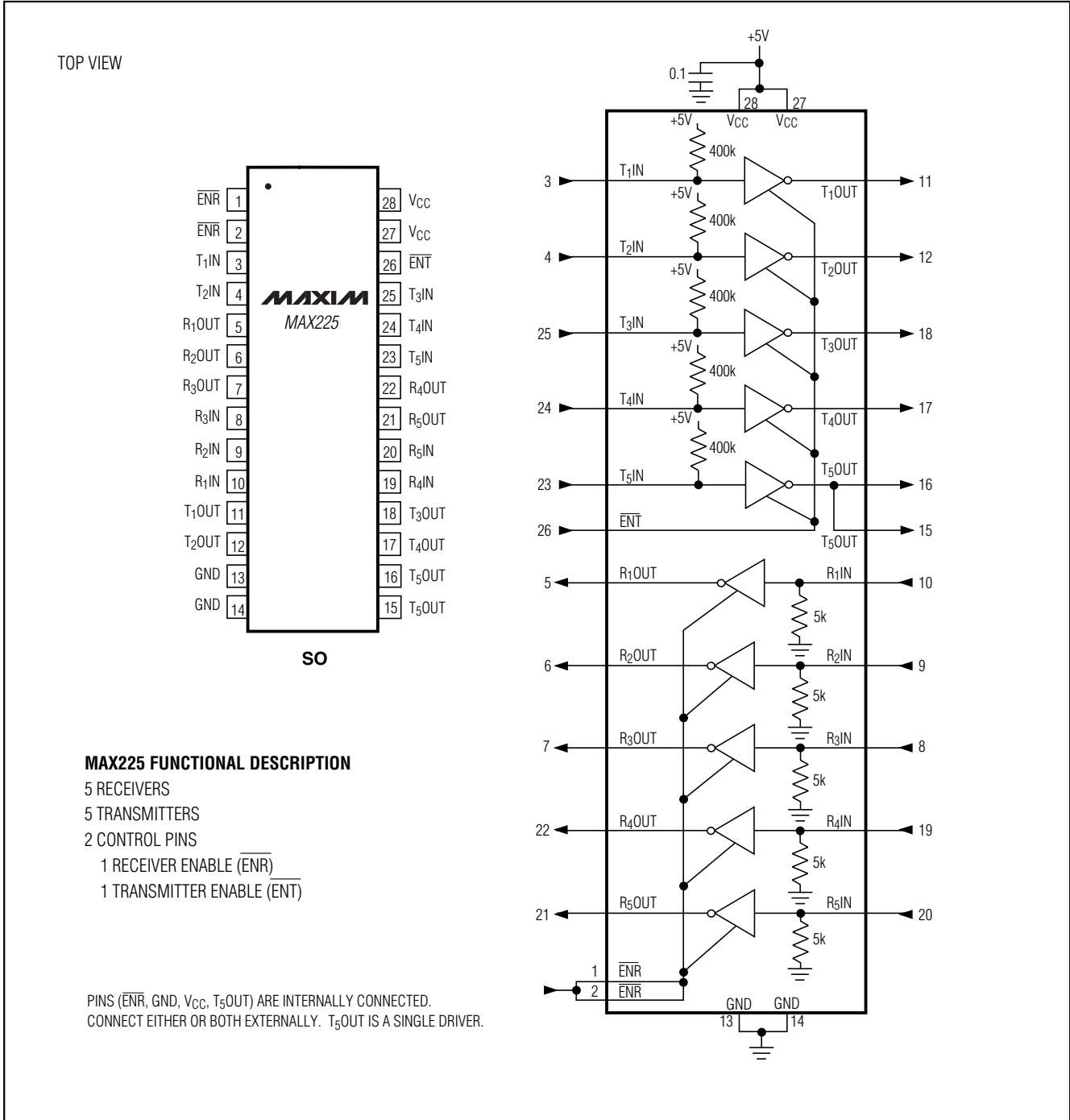


Figure 7. MAX225 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

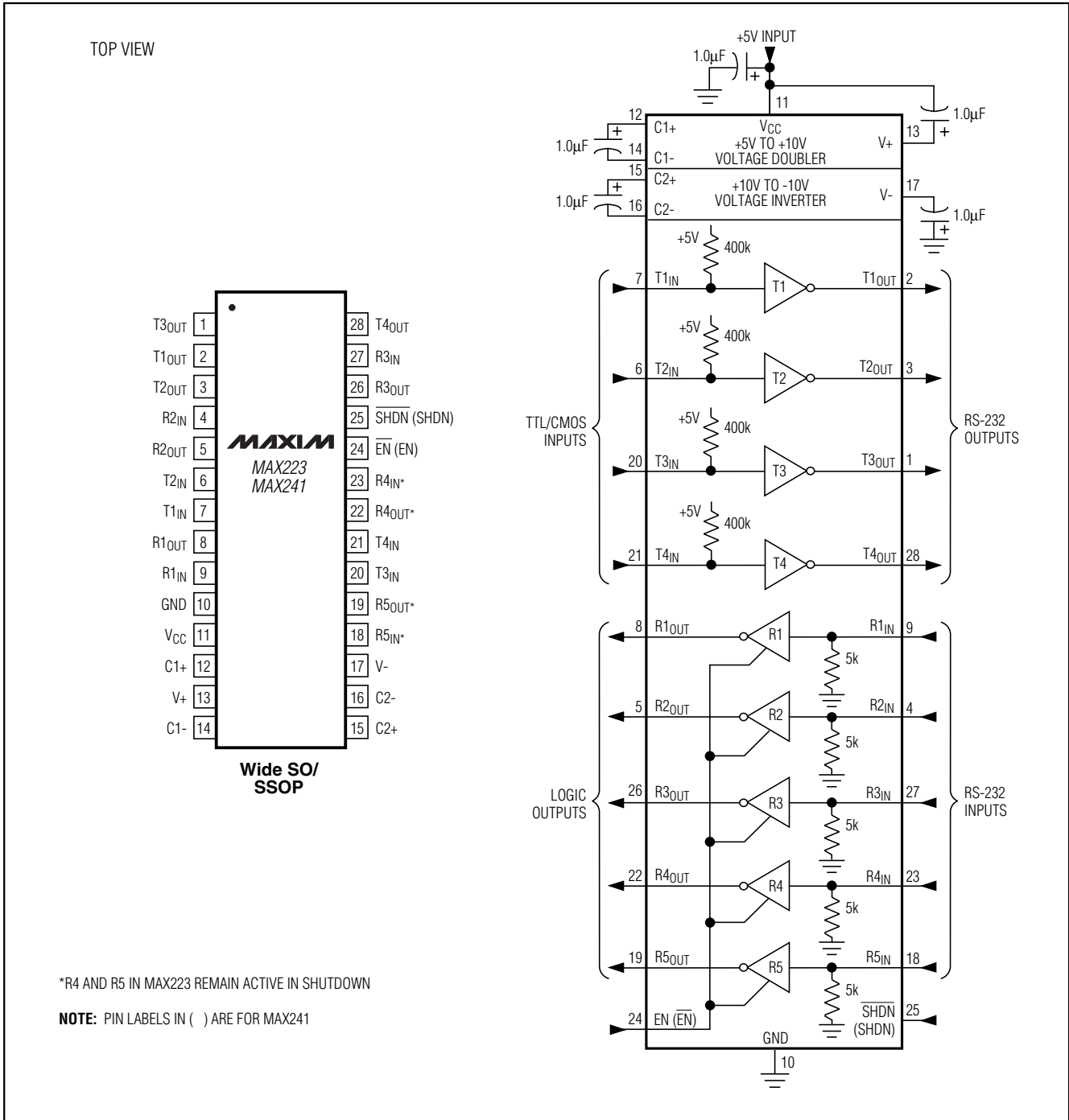


Figure 8. MAX223/MAX241 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

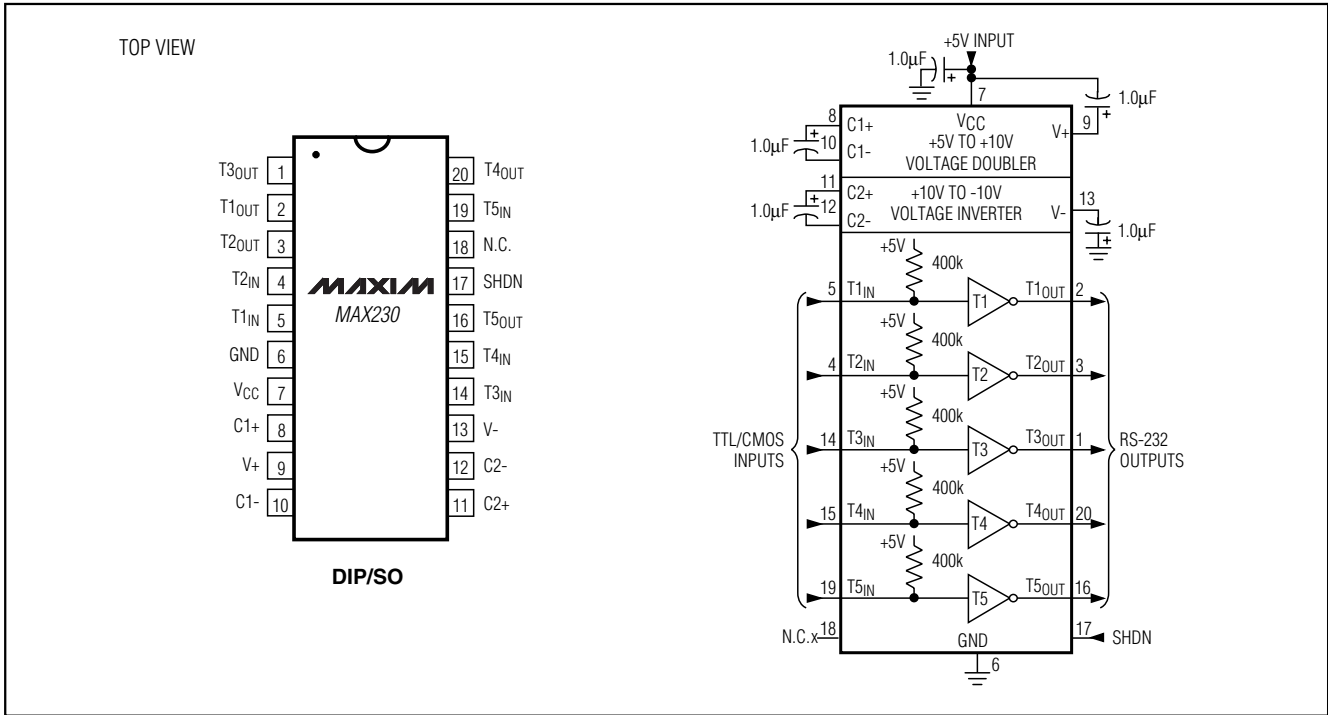


Figure 9. MAX230 Pin Configuration and Typical Operating Circuit

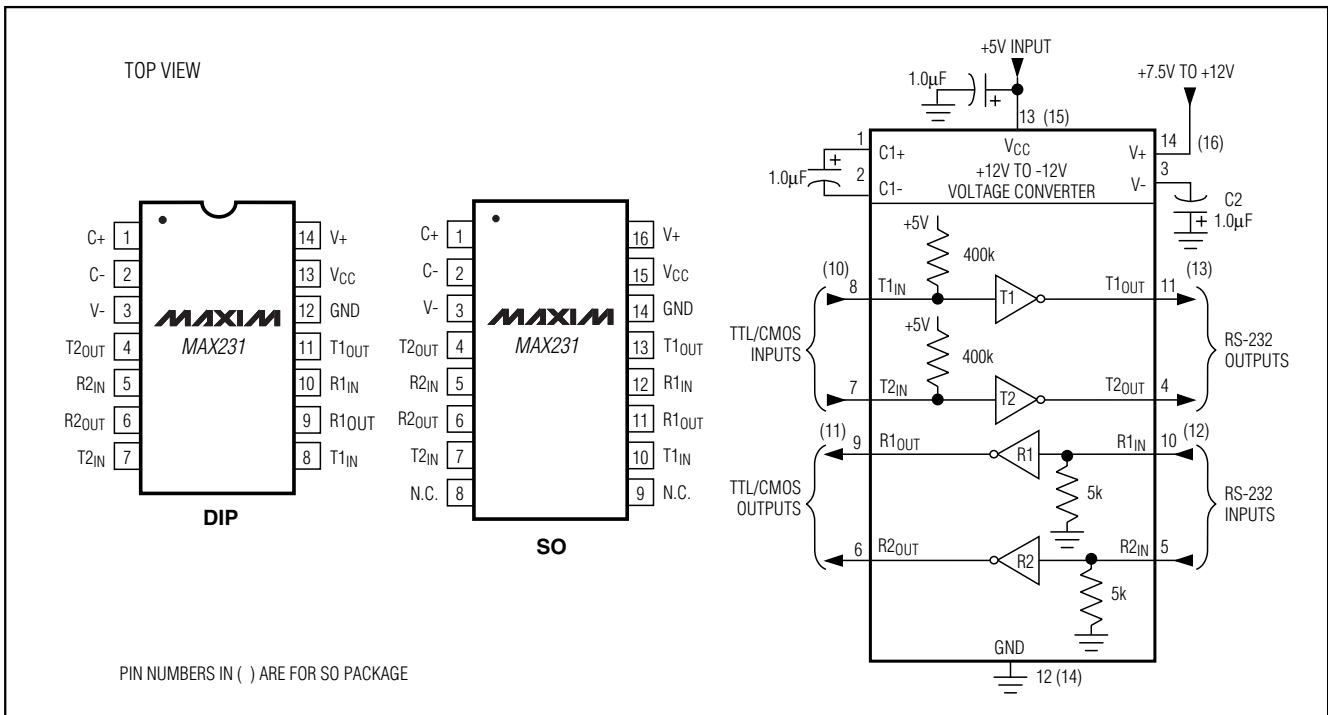


Figure 10. MAX231 Pin Configurations and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

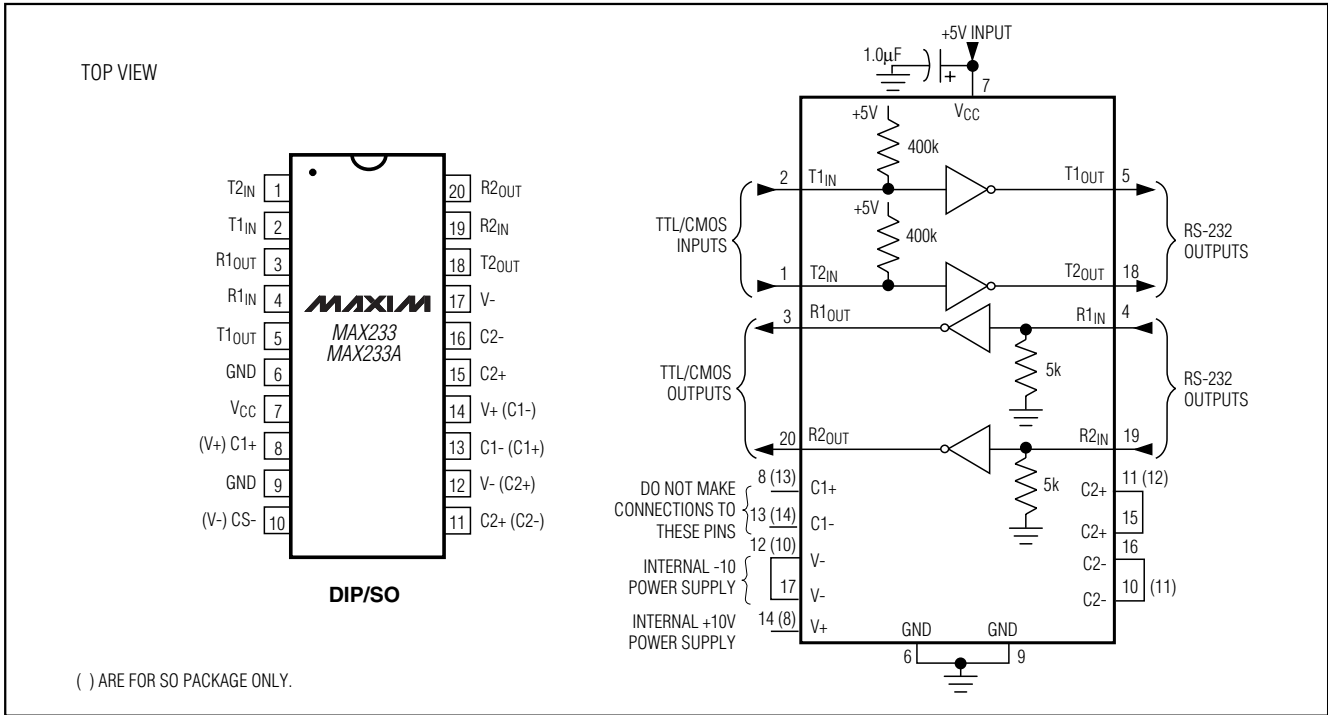


Figure 11. MAX233/MAX233A Pin Configuration and Typical Operating Circuit

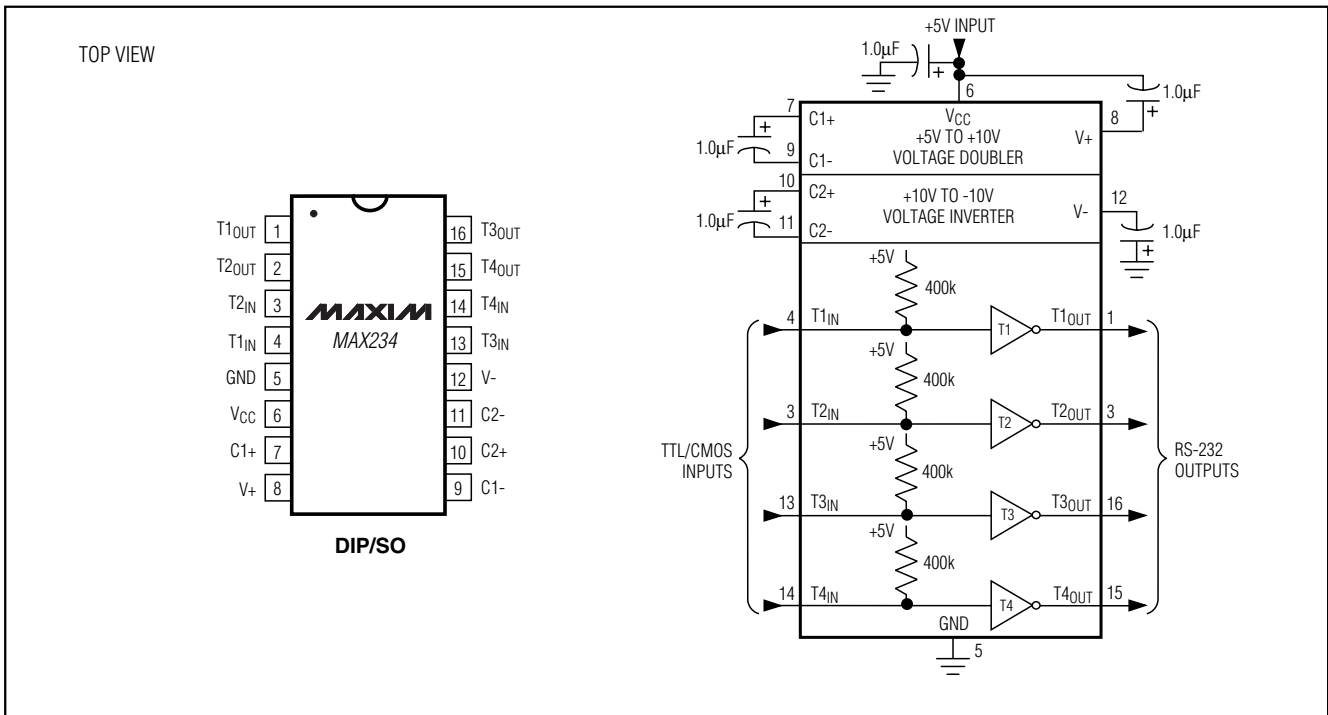


Figure 12. MAX234 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

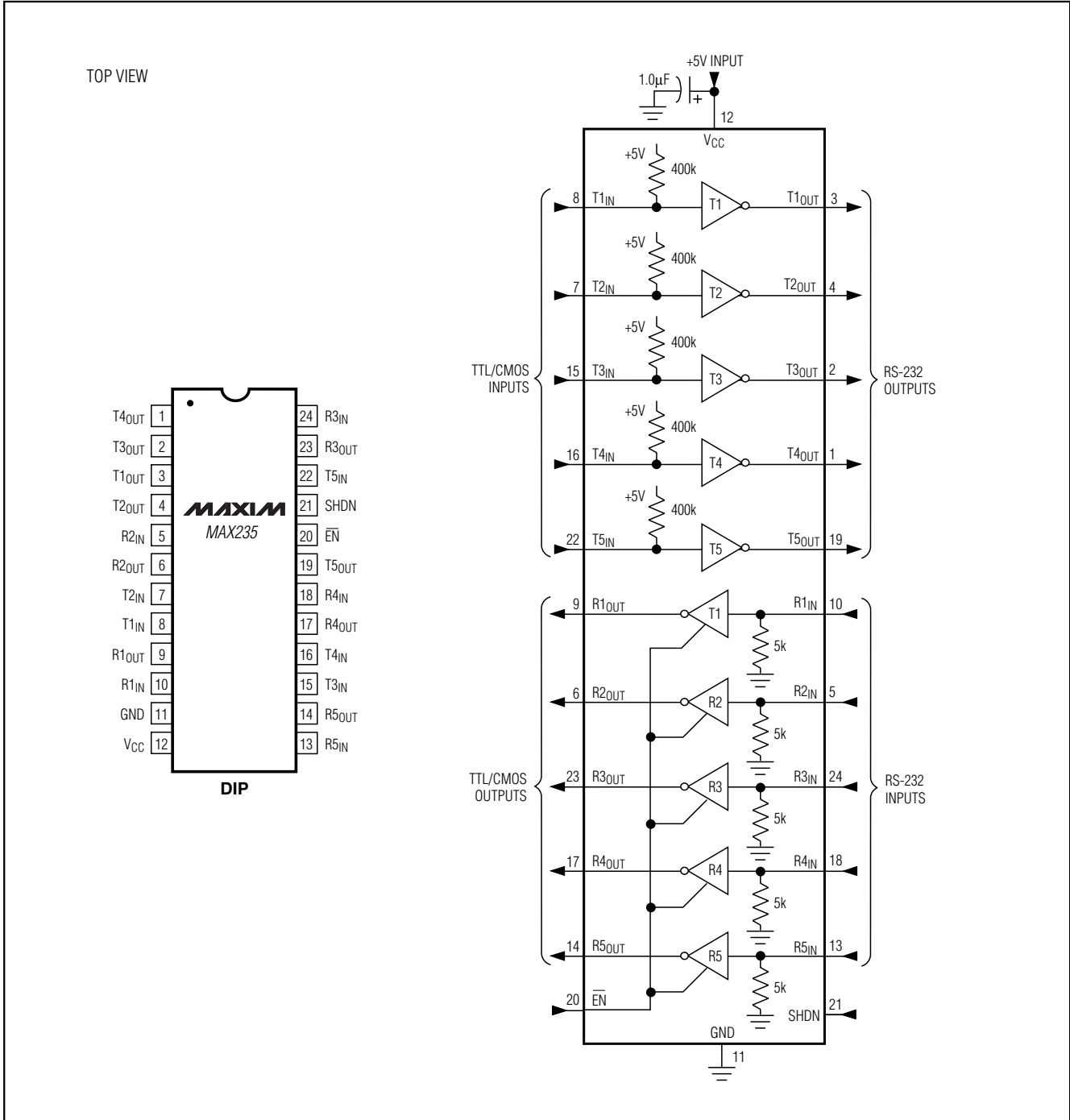


Figure 13. MAX235 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX2220-MAX249

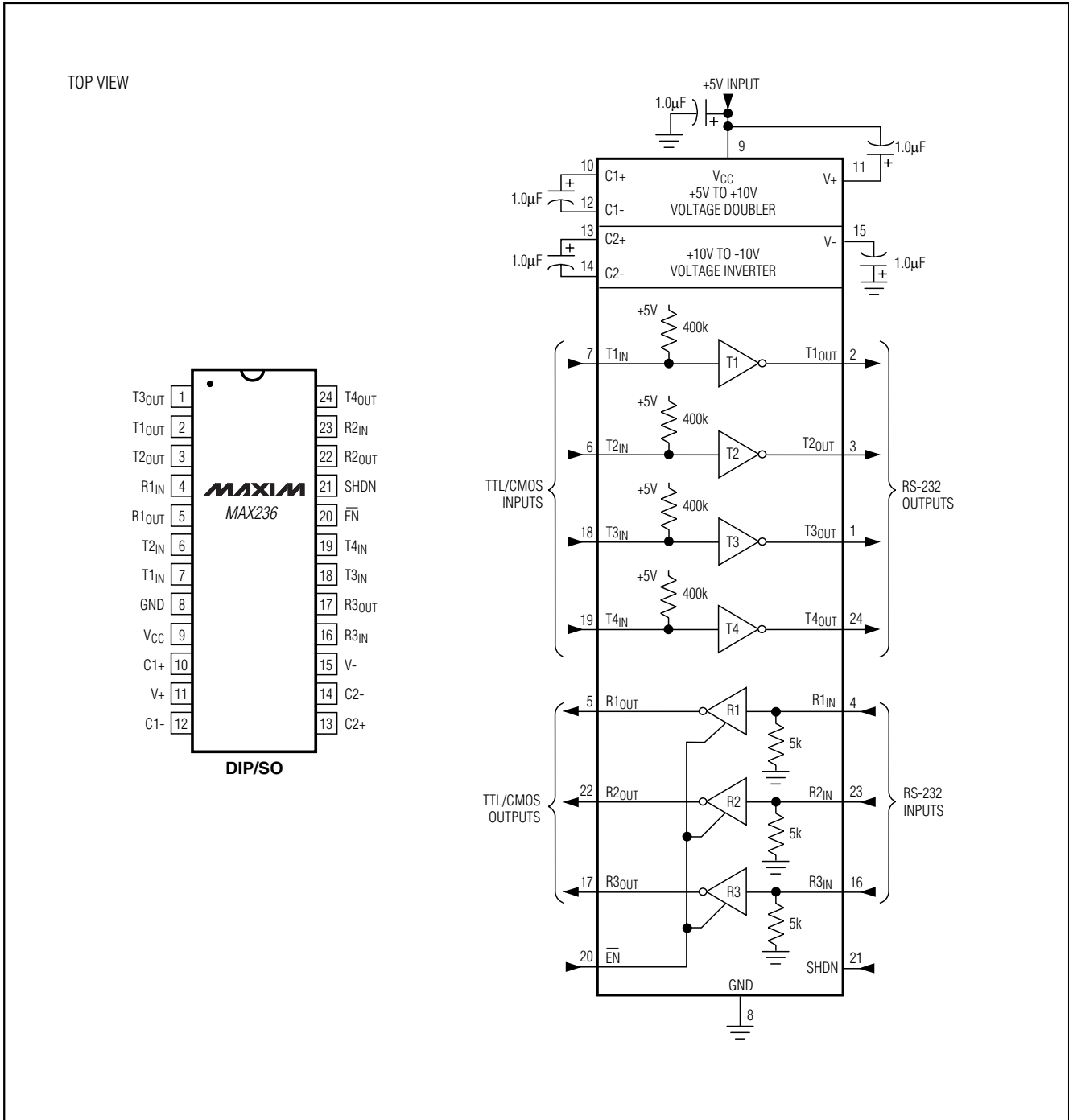


Figure 14. MAX236 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

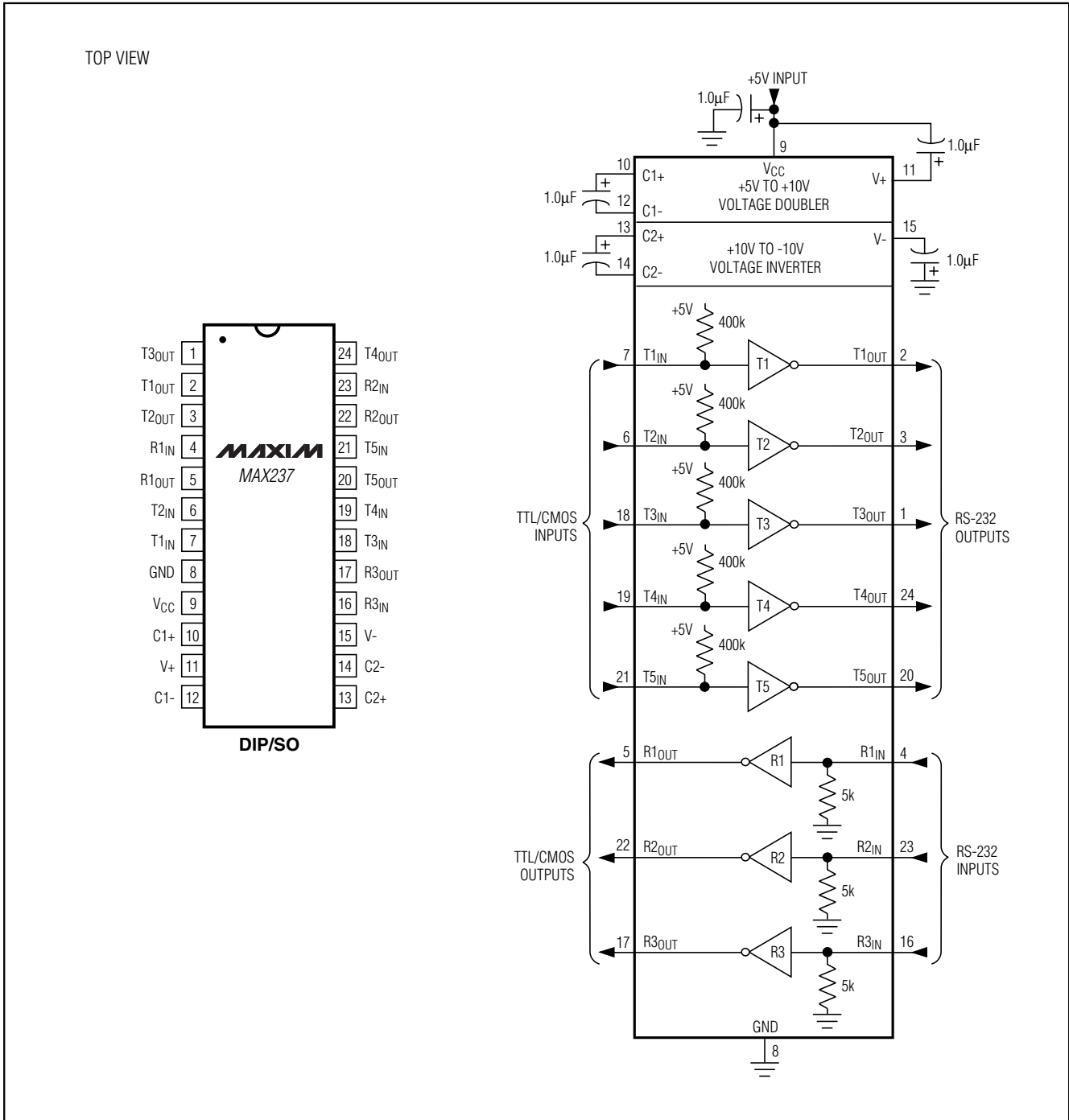


Figure 15. MAX237 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

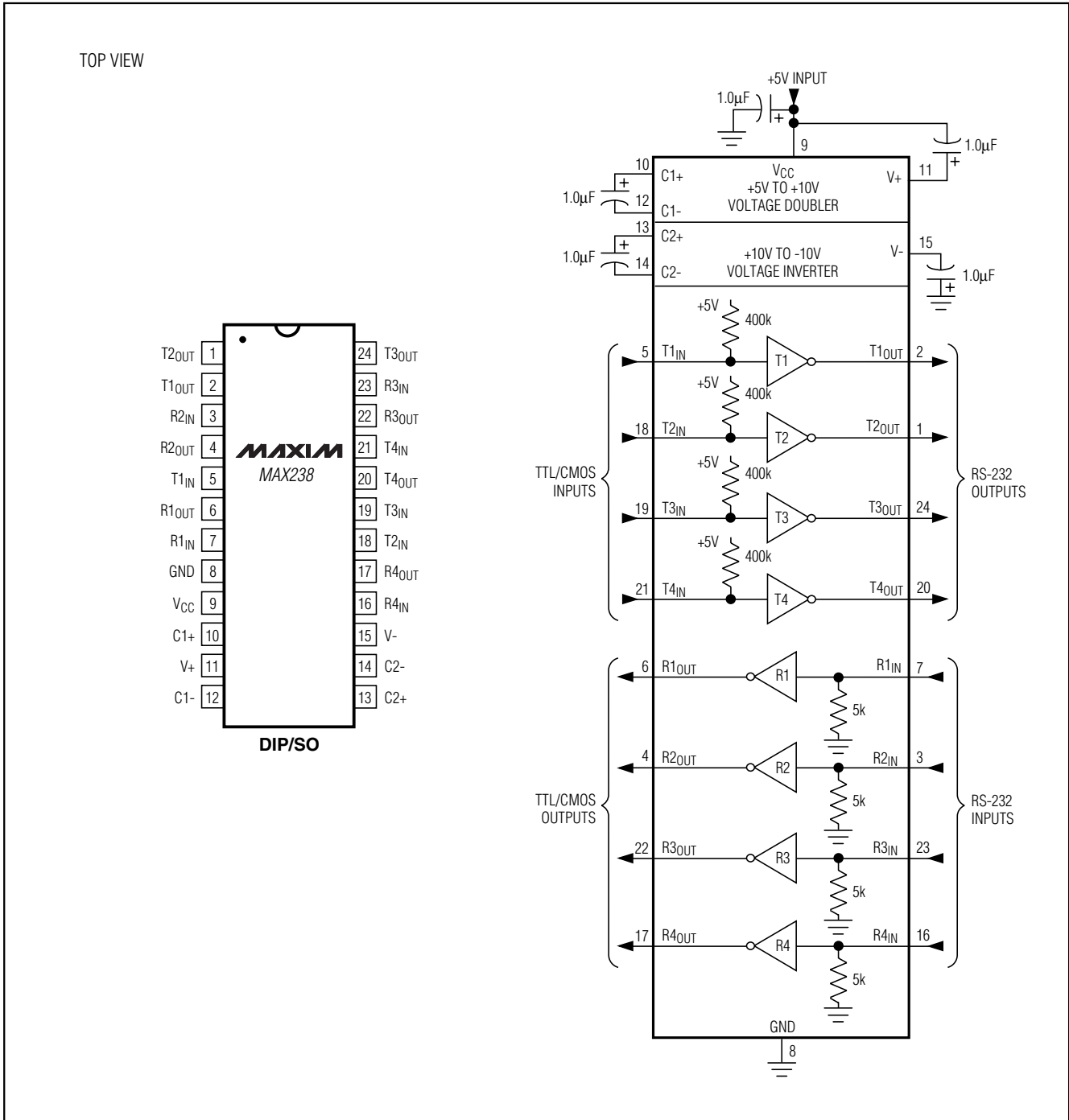
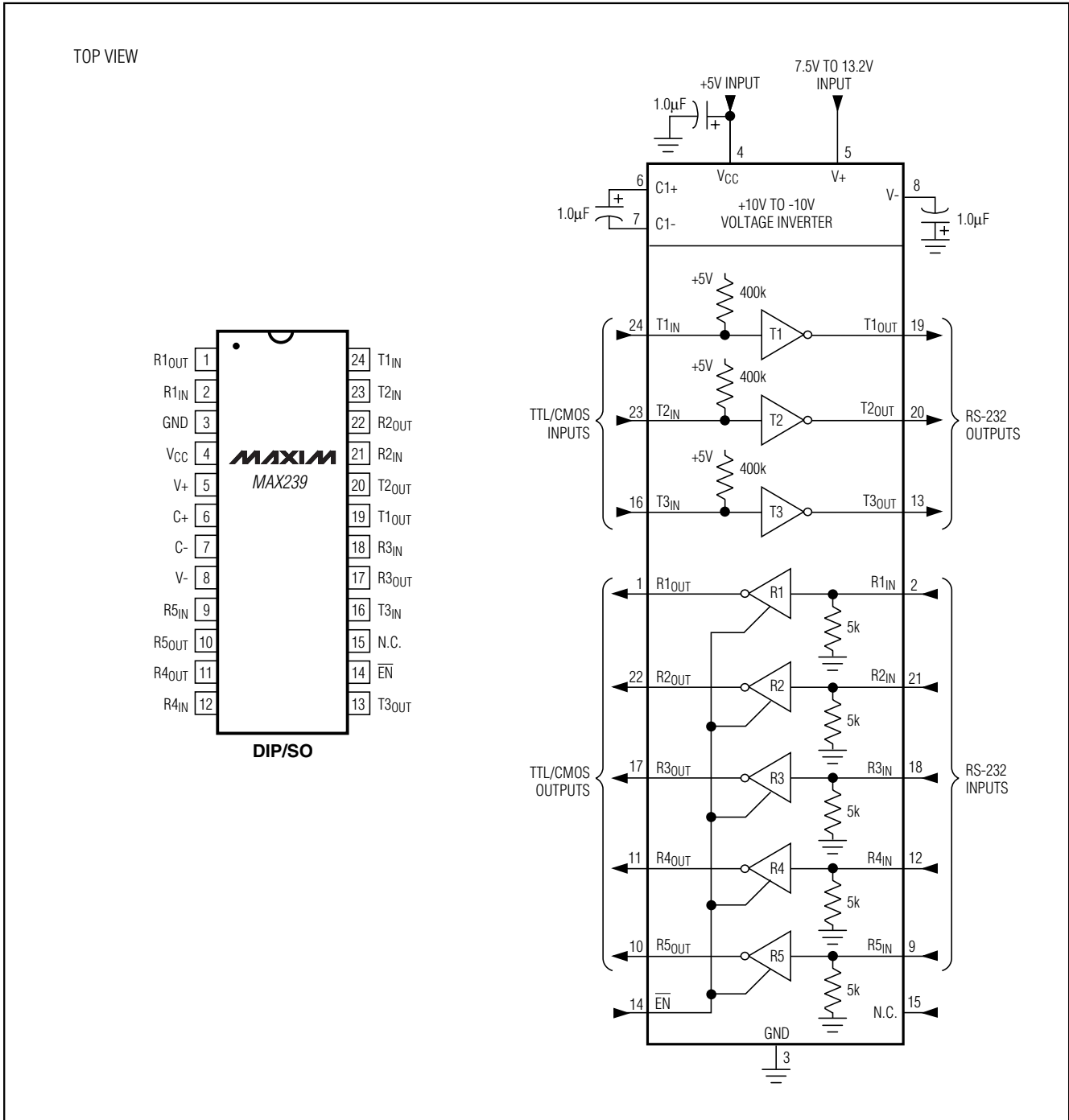


Figure 16. MAX238 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers



+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

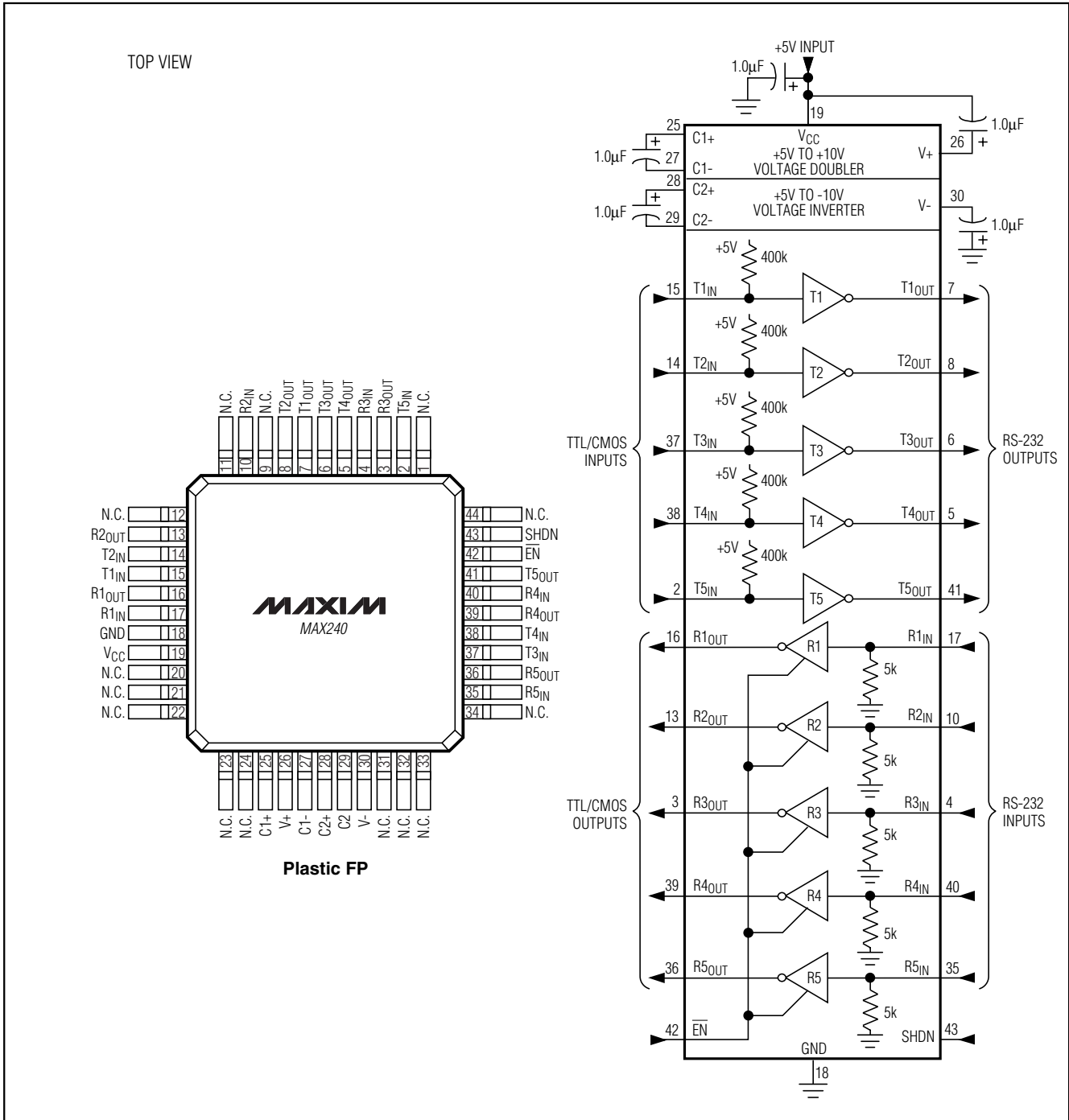


Figure 18. MAX240 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

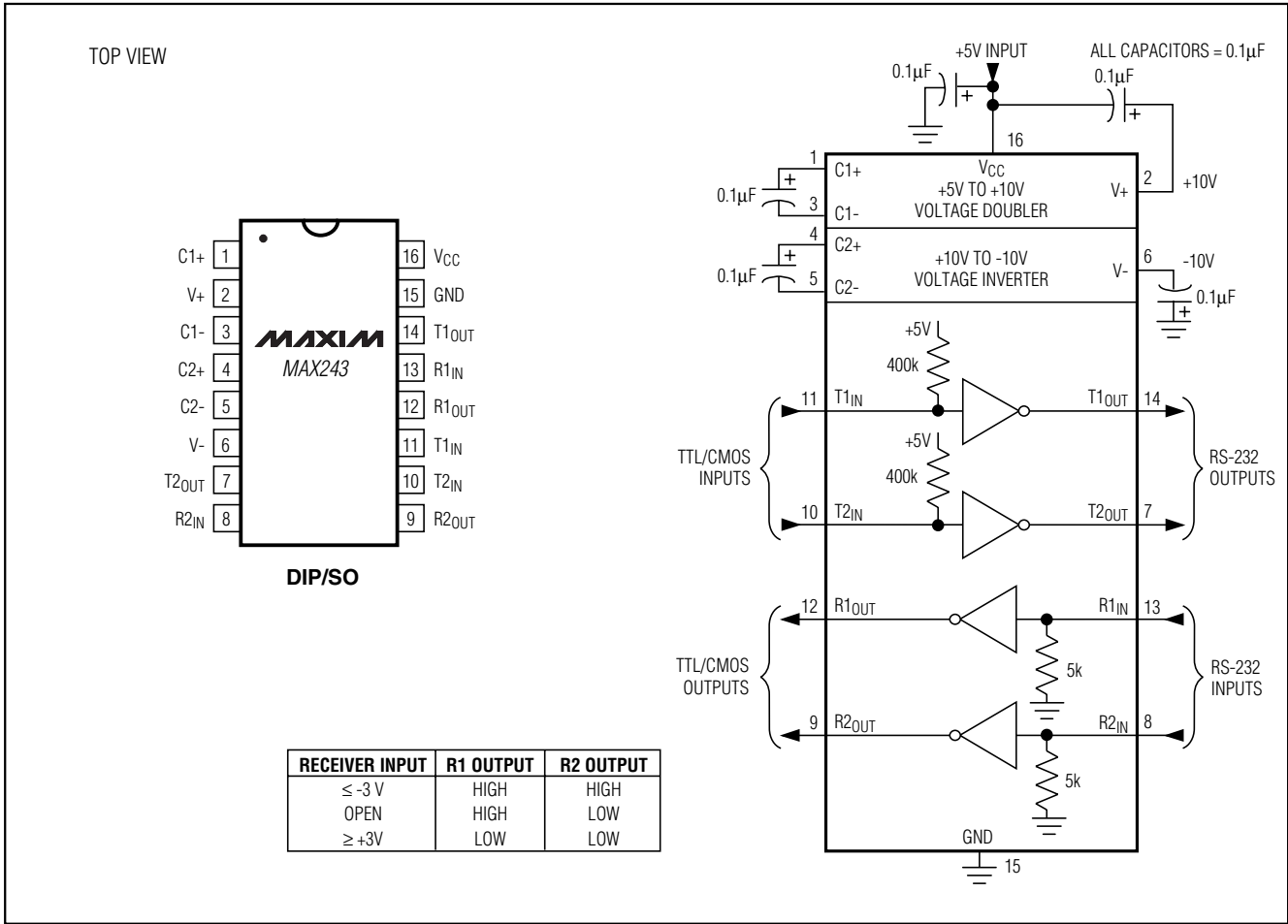


Figure 19. MAX243 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

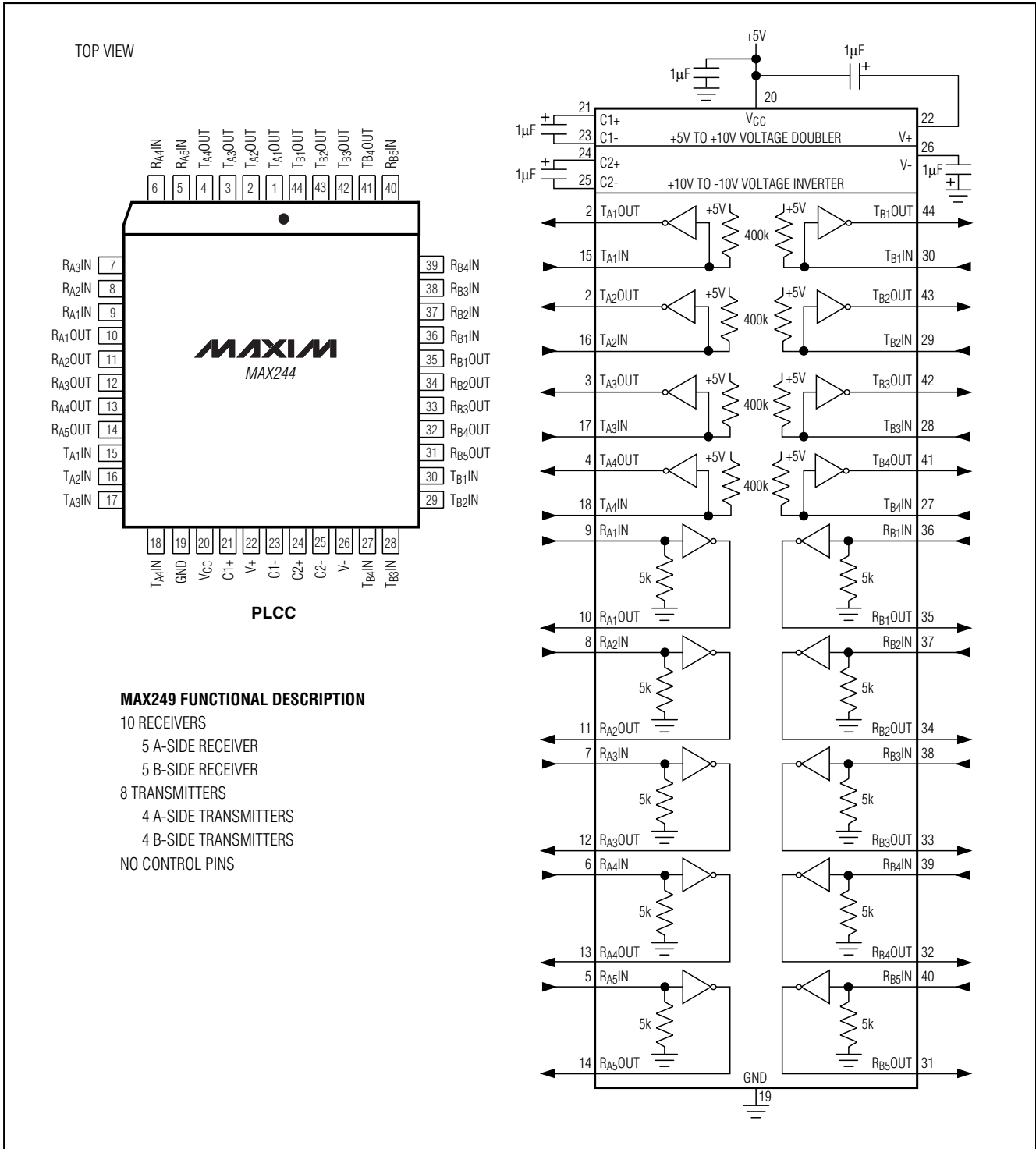


Figure 20. MAX244 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

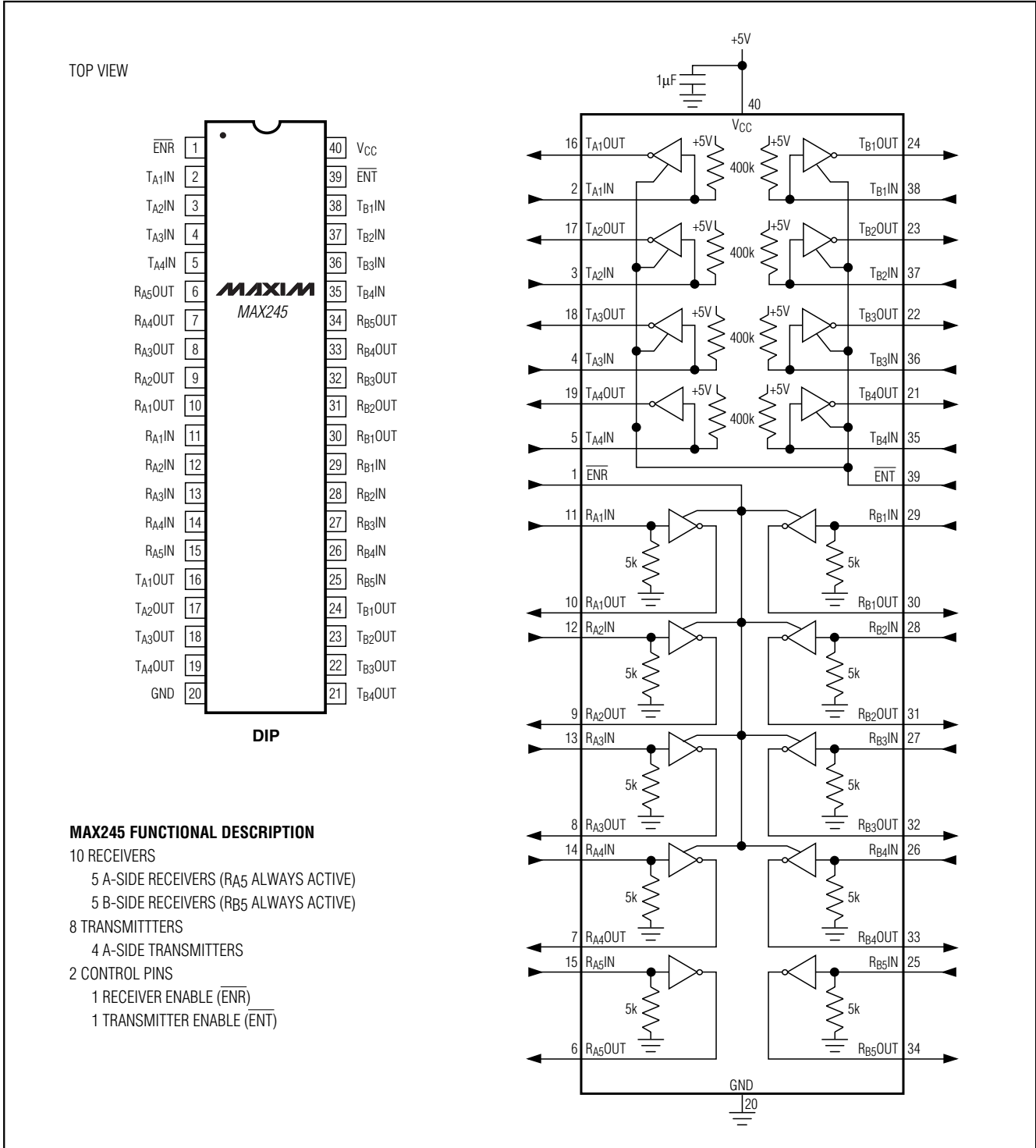


Figure 21. MAX245 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

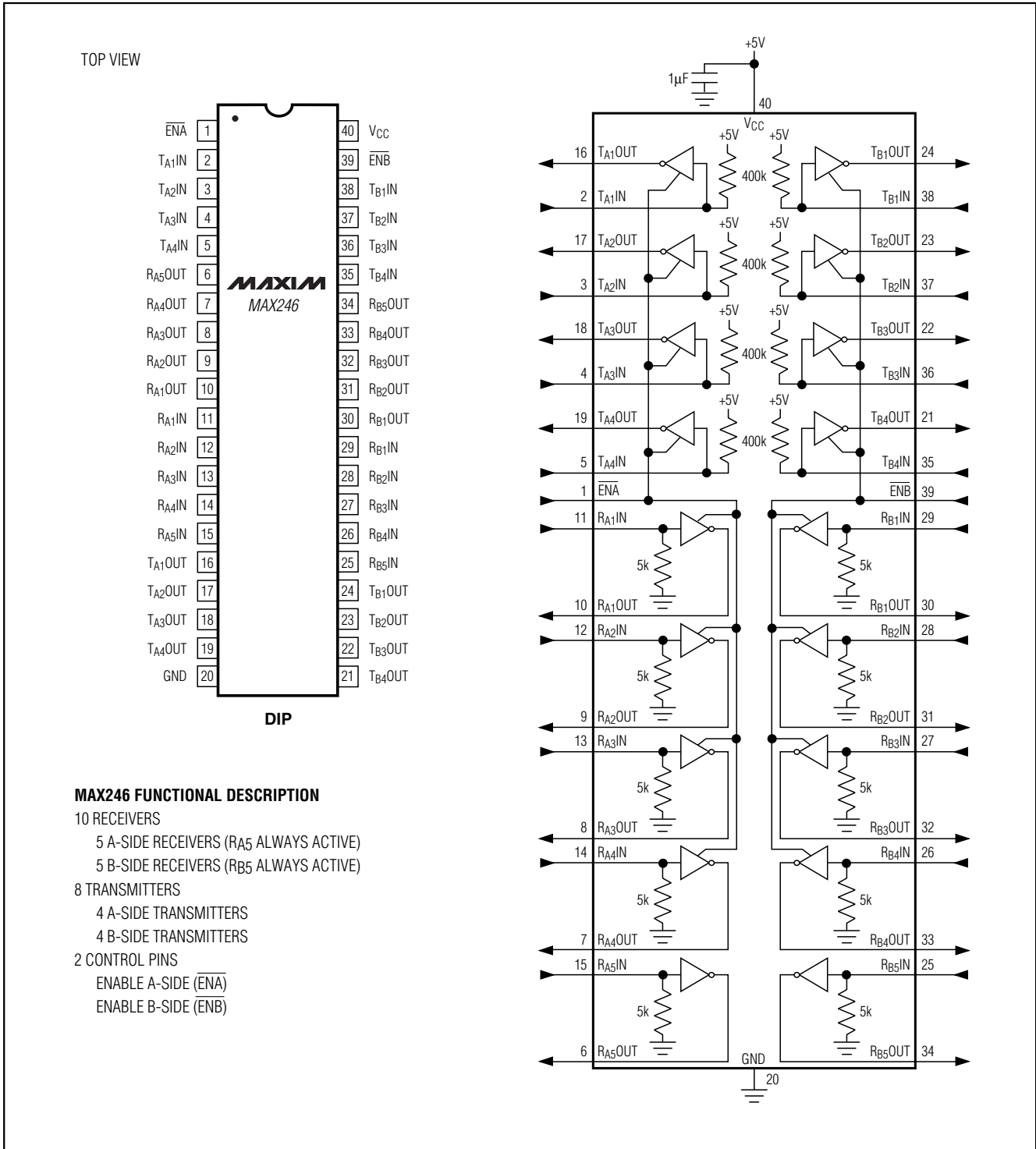
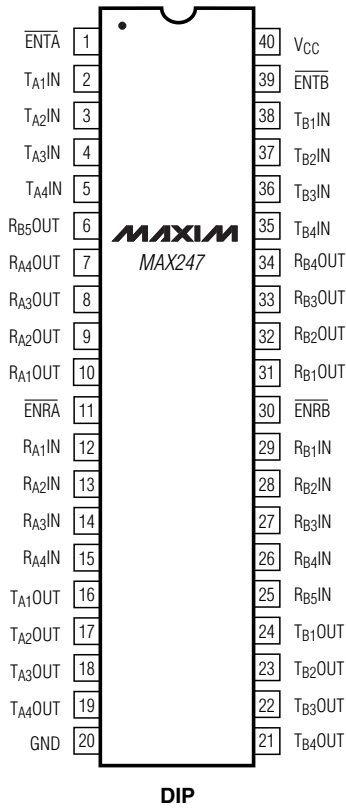


Figure 22. MAX246 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

TOP VIEW



MAX247 FUNCTIONAL DESCRIPTION

- 9 RECEIVERS
 - 4 A-SIDE RECEIVERS
 - 5 B-SIDE RECEIVERS (RB5 ALWAYS ACTIVE)
- 8 TRANSMITTERS
 - 4 A-SIDE TRANSMITTERS
 - 4 B-SIDE TRANSMITTERS
- 4 CONTROL PINS
 - ENABLE RECEIVER A-SIDE ($\overline{\text{ENRA}}$)
 - ENABLE RECEIVER B-SIDE ($\overline{\text{ENRB}}$)
 - ENABLE RECEIVER A-SIDE ($\overline{\text{ENTA}}$)
 - ENABLE RECEIVER B-SIDE ($\overline{\text{ENTB}}$)

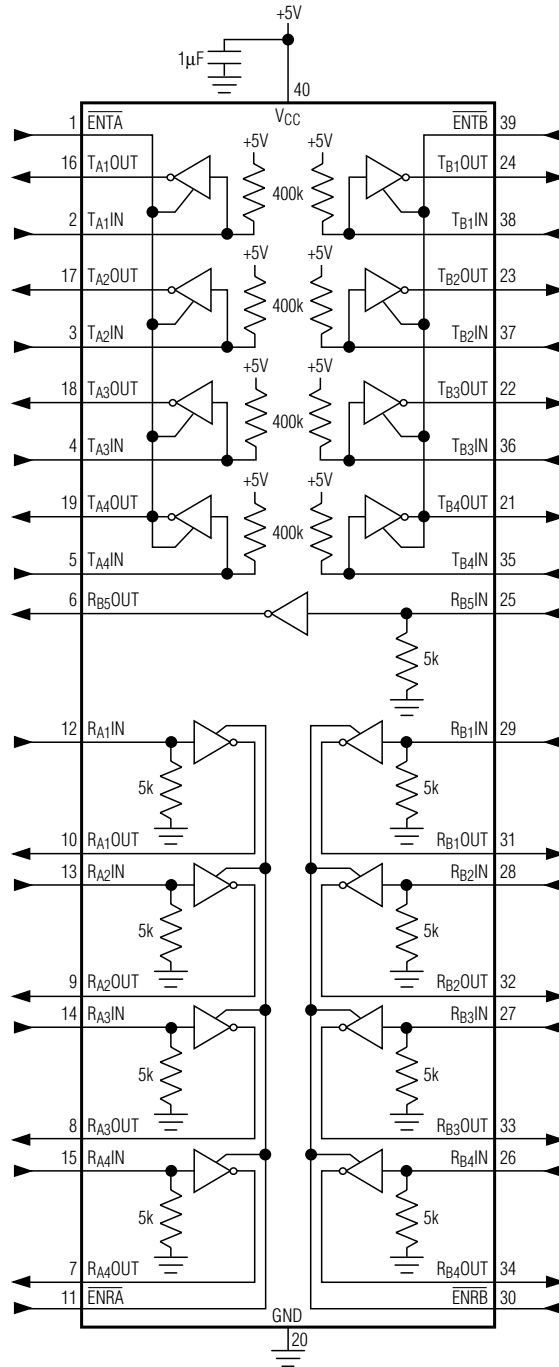


Figure 23. MAX247 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

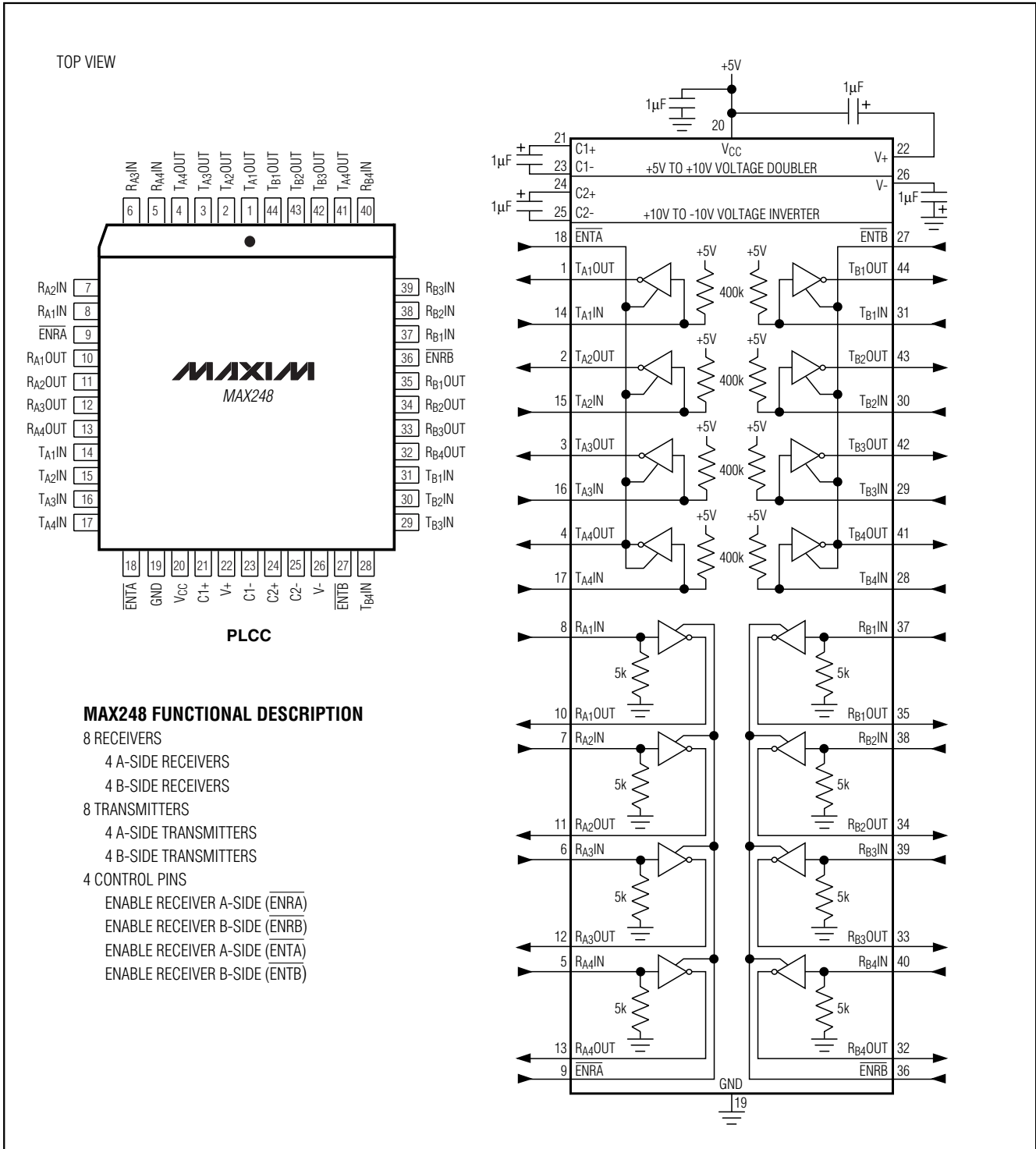


Figure 24. MAX248 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

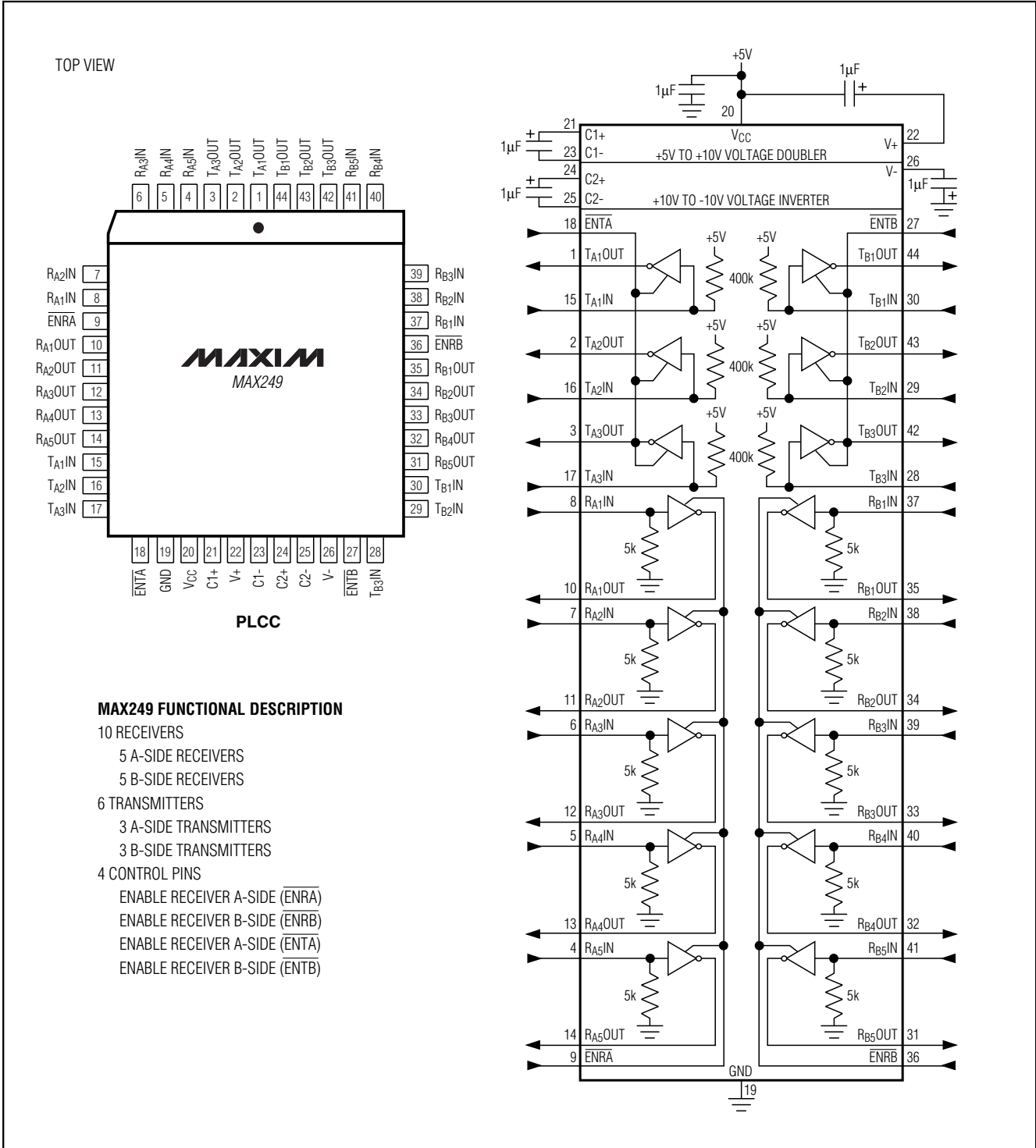


Figure 25. MAX249 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Ordering Information (continued)

MAX220-MAX249

PART	TEMP. RANGE	PIN-PACKAGE
MAX222 CPN	0°C to +70°C	18 Plastic DIP
MAX222CWN	0°C to +70°C	18 Wide SO
MAX222C/D	0°C to +70°C	Dice*
MAX222EPN	-40°C to +85°C	18 Plastic DIP
MAX222EWN	-40°C to +85°C	18 Wide SO
MAX222EJN	-40°C to +85°C	18 CERDIP
MAX222MJN	-55°C to +125°C	18 CERDIP
MAX223 CAI	0°C to +70°C	28 SSOP
MAX223CWI	0°C to +70°C	28 Wide SO
MAX223C/D	0°C to +70°C	Dice*
MAX223EAI	-40°C to +85°C	28 SSOP
MAX223EWI	-40°C to +85°C	28 Wide SO
MAX225 CWI	0°C to +70°C	28 Wide SO
MAX225EWI	-40°C to +85°C	28 Wide SO
MAX230 CPP	0°C to +70°C	20 Plastic DIP
MAX230CWP	0°C to +70°C	20 Wide SO
MAX230C/D	0°C to +70°C	Dice*
MAX230EPP	-40°C to +85°C	20 Plastic DIP
MAX230EWP	-40°C to +85°C	20 Wide SO
MAX230EJP	-40°C to +85°C	20 CERDIP
MAX230MJP	-55°C to +125°C	20 CERDIP
MAX231 CPD	0°C to +70°C	14 Plastic DIP
MAX231CWE	0°C to +70°C	16 Wide SO
MAX231CJD	0°C to +70°C	14 CERDIP
MAX231C/D	0°C to +70°C	Dice*
MAX231EPD	-40°C to +85°C	14 Plastic DIP
MAX231EWE	-40°C to +85°C	16 Wide SO
MAX231EJD	-40°C to +85°C	14 CERDIP
MAX231MJD	-55°C to +125°C	14 CERDIP
MAX232 CPE	0°C to +70°C	16 Plastic DIP
MAX232CSE	0°C to +70°C	16 Narrow SO
MAX232CWE	0°C to +70°C	16 Wide SO
MAX232C/D	0°C to +70°C	Dice*
MAX232EPE	-40°C to +85°C	16 Plastic DIP
MAX232ESE	-40°C to +85°C	16 Narrow SO
MAX232EWE	-40°C to +85°C	16 Wide SO
MAX232EJE	-40°C to +85°C	16 CERDIP
MAX232MJE	-55°C to +125°C	16 CERDIP
MAX232MLP	-55°C to +125°C	20 LCC
MAX232A CPE	0°C to +70°C	16 Plastic DIP
MAX232ACSE	0°C to +70°C	16 Narrow SO
MAX232ACWE	0°C to +70°C	16 Wide SO

MAX232AC/D	0°C to +70°C	Dice*
MAX232AEPE	-40°C to +85°C	16 Plastic DIP
MAX232AESE	-40°C to +85°C	16 Narrow SO
MAX232AEWE	-40°C to +85°C	16 Wide SO
MAX232AEJE	-40°C to +85°C	16 CERDIP
MAX232AMJE	-55°C to +125°C	16 CERDIP
MAX232AML P	-55°C to +125°C	20 LCC
MAX233 CPP	0°C to +70°C	20 Plastic DIP
MAX233EPP	-40°C to +85°C	20 Plastic DIP
MAX233A CPP	0°C to +70°C	20 Plastic DIP
MAX233ACWP	0°C to +70°C	20 Wide SO
MAX233AEPP	-40°C to +85°C	20 Plastic DIP
MAX233AEWP	-40°C to +85°C	20 Wide SO
MAX234 CPE	0°C to +70°C	16 Plastic DIP
MAX234CWE	0°C to +70°C	16 Wide SO
MAX234C/D	0°C to +70°C	Dice*
MAX234EPE	-40°C to +85°C	16 Plastic DIP
MAX234EWE	-40°C to +85°C	16 Wide SO
MAX234EJE	-40°C to +85°C	16 CERDIP
MAX234MJE	-55°C to +125°C	16 CERDIP
MAX235 CPG	0°C to +70°C	24 Wide Plastic DIP
MAX235EPG	-40°C to +85°C	24 Wide Plastic DIP
MAX235EDG	-40°C to +85°C	24 Ceramic SB
MAX235MDG	-55°C to +125°C	24 Ceramic SB
MAX236 CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX236CWG	0°C to +70°C	24 Wide SO
MAX236C/D	0°C to +70°C	Dice*
MAX236ENG	-40°C to +85°C	24 Narrow Plastic DIP
MAX236EWG	-40°C to +85°C	24 Wide SO
MAX236ERG	-40°C to +85°C	24 Narrow CERDIP
MAX236MRG	-55°C to +125°C	24 Narrow CERDIP
MAX237 CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX237CWG	0°C to +70°C	24 Wide SO
MAX237C/D	0°C to +70°C	Dice*
MAX237ENG	-40°C to +85°C	24 Narrow Plastic DIP
MAX237EWG	-40°C to +85°C	24 Wide SO
MAX237ERG	-40°C to +85°C	24 Narrow CERDIP
MAX237MRG	-55°C to +125°C	24 Narrow CERDIP
MAX238 CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX238CWG	0°C to +70°C	24 Wide SO
MAX238C/D	0°C to +70°C	Dice*
MAX238ENG	-40°C to +85°C	24 Narrow Plastic DIP

* Contact factory for dice specifications.

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Ordering Information (continued)

PART	TEMP. RANGE	PIN-PACKAGE
MAX238EWG	-40°C to +85°C	24 Wide SO
MAX238ERG	-40°C to +85°C	24 Narrow CERDIP
MAX238MRG	-55°C to +125°C	24 Narrow CERDIP
MAX239 CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX239CWG	0°C to +70°C	24 Wide SO
MAX239C/D	0°C to +70°C	Dice*
MAX239ENG	-40°C to +85°C	24 Narrow Plastic DIP
MAX239EWG	-40°C to +85°C	24 Wide SO
MAX239ERG	-40°C to +85°C	24 Narrow CERDIP
MAX239MRG	-55°C to +125°C	24 Narrow CERDIP
MAX240 CMH	0°C to +70°C	44 Plastic FP
MAX240C/D	0°C to +70°C	Dice*
MAX241 CAI	0°C to +70°C	28 SSOP
MAX241CWI	0°C to +70°C	28 Wide SO
MAX241C/D	0°C to +70°C	Dice*
MAX241EAI	-40°C to +85°C	28 SSOP
MAX241EWI	-40°C to +85°C	28 Wide SO
MAX242 CAP	0°C to +70°C	20 SSOP
MAX242CPN	0°C to +70°C	18 Plastic DIP
MAX242CWN	0°C to +70°C	18 Wide SO
MAX242C/D	0°C to +70°C	Dice*
MAX242EPN	-40°C to +85°C	18 Plastic DIP
MAX242EWN	-40°C to +85°C	18 Wide SO
MAX242EJN	-40°C to +85°C	18 CERDIP
MAX242MJN	-55°C to +125°C	18 CERDIP

MAX243 CPE	0°C to +70°C	16 Plastic DIP
MAX243CSE	0°C to +70°C	16 Narrow SO
MAX243CWE	0°C to +70°C	16 Wide SO
MAX243C/D	0°C to +70°C	Dice*
MAX243EPE	-40°C to +85°C	16 Plastic DIP
MAX243ESE	-40°C to +85°C	16 Narrow SO
MAX243EWE	-40°C to +85°C	16 Wide SO
MAX243EJE	-40°C to +85°C	16 CERDIP
MAX243MJE	-55°C to +125°C	16 CERDIP
MAX244 CQH	0°C to +70°C	44 PLCC
MAX244C/D	0°C to +70°C	Dice*
MAX244EQH	-40°C to +85°C	44 PLCC
MAX245 CPL	0°C to +70°C	40 Plastic DIP
MAX245C/D	0°C to +70°C	Dice*
MAX245EPL	-40°C to +85°C	40 Plastic DIP
MAX246 CPL	0°C to +70°C	40 Plastic DIP
MAX246C/D	0°C to +70°C	Dice*
MAX246EPL	-40°C to +85°C	40 Plastic DIP
MAX247 CPL	0°C to +70°C	40 Plastic DIP
MAX247C/D	0°C to +70°C	Dice*
MAX247EPL	-40°C to +85°C	40 Plastic DIP
MAX248 CQH	0°C to +70°C	44 PLCC
MAX248C/D	0°C to +70°C	Dice*
MAX248EQH	-40°C to +85°C	44 PLCC
MAX249 CQH	0°C to +70°C	44 PLCC
MAX249EQH	-40°C to +85°C	44 PLCC

* Contact factory for dice specifications.

Maxim cannot assume responsibility for use of any circuitry other than circuitry entirely embodied in a Maxim product. No circuit patent licenses are implied. Maxim reserves the right to change the circuitry and specifications without notice at any time.

36 **Maxim Integrated Products, 120 San Gabriel Drive, Sunnyvale, CA 94086 (408) 737-7600**