



Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

Email & Skype: info@chipsmall.com Web: www.chipsmall.com


Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China



Robotics with the Boe-Bot

Student Guide

VERSION 3.0

PARALLAX 

WARRANTY

Parallax warrants its products against defects in materials and workmanship for a period of 90 days from receipt of product. If you discover a defect, Parallax will, at its option, repair or replace the merchandise, or refund the purchase price. Before returning the product to Parallax, call for a Return Merchandise Authorization (RMA) number. Write the RMA number on the outside of the box used to return the merchandise to Parallax. Please enclose the following along with the returned merchandise: your name, telephone number, shipping address, and a description of the problem. Parallax will return your product or its replacement using the same shipping method used to ship the product to Parallax.

14-DAY MONEY BACK GUARANTEE

If, within 14 days of having received your product, you find that it does not suit your needs, you may return it for a full refund. Parallax will refund the purchase price of the product, excluding shipping/handling costs. This guarantee is void if the product has been altered or damaged. See the Warranty section above for instructions on returning a product to Parallax.

COPYRIGHTS AND TRADEMARKS

This documentation is Copyright 2003-2010 by Parallax Inc. By downloading or obtaining a printed copy of this documentation or software you agree that it is to be used exclusively with Parallax microcontrollers and products. Any other uses are not permitted and may represent a violation of Parallax copyrights, legally punishable according to Federal copyright or intellectual property laws. Any duplication of this documentation for commercial uses is expressly prohibited by Parallax Inc. Duplication for educational use, in whole or in part, is permitted subject to the following conditions: the material is to be used solely in conjunction with Parallax microcontrollers and products, and the user may recover from the student only the cost of duplication. Check with Parallax for approval prior to duplicating any of our documentation in part or whole for any other use.

BASIC Stamp, Board of Education, Boe-Bot, Stamps in Class, and SumoBot are registered trademarks of Parallax Inc. HomeWork Board, PING)), Parallax, the Parallax logo, Propeller, and Spin are trademarks of Parallax Inc. If you decide to use any of these words on your electronic or printed material, you must state that "(trademark) is a (registered) trademark of Parallax Inc." upon the first use of the trademark name. Other brand and product names herein are trademarks or registered trademarks of their respective holders.

ISBN 9781928982531

3.0.0-10.11.10-HKTP

DISCLAIMER OF LIABILITY

Parallax Inc. is not responsible for special, incidental, or consequential damages resulting from any breach of warranty, or under any legal theory, including lost profits, downtime, goodwill, damage to or replacement of equipment or property, or any costs of recovering, reprogramming, or reproducing any data stored in or used with Parallax products. Parallax is also not responsible for any personal damage, including that to life and health, resulting from use of any of our products. You take full responsibility for your BASIC Stamp application, no matter how life-threatening it may be.

ERRATA

While great effort is made to assure the accuracy of our texts, errors may still exist. Occasionally an errata sheet with a list of known errors and corrections for a given text will be posted on the related product page at www.parallax.com. If you find an error, please send an email to editor@parallax.com.

Table of Contents

Preface	5
About Version 3.0	6
Audience.....	6
Support Forums.....	7
Resources for Educators	8
Foreign Translations	9
About the Author.....	9
Special Contributors	9
Chapter 1 : Your Boe-Bot's Brain	11
Hardware and Software	12
Activity #1 : Getting the Software.....	12
Activity #2 : Using the Help File for Hardware Setup.....	17
Summary	19
Chapter 2 : Your Boe-Bot's Servo Motors	23
Introducing the Continuous Rotation Servo	23
Activity #1 : Building and Testing the LED Circuit.....	24
Activity #2 : Tracking Time and Repeating Actions with a Circuit.....	27
Activity #3 : Connecting the Servo Motors.....	40
Activity #4 : Centering the Servos.....	49
Activity #5 : How To Store Values and Count.....	53
Activity #6 : Testing the Servos	58
Summary	67
Chapter 3 : Assemble and Test Your Boe-Bot	73
Activity #1 : Assembling the Boe-Bot Robot	73
Activity #2 : Re-Test the Servos	82
Activity #3 : Start/Reset Indicator Circuit and Program.....	86
Activity #4 : Testing Speed Control with the Debug Terminal.....	92
Summary	98
Chapter 4 : Boe-Bot Navigation	103
Activity #1 : Basic Boe-Bot Maneuvers.....	103
Activity #2 : Tuning the Basic Maneuvers.....	109
Activity #3 : Calculating Distances.....	112
Activity #4 : Maneuvers—Ramping.....	117
Activity #5 : Simplify Navigation with Subroutines	120
Activity #6 : Advanced Topic—Building Complex Maneuvers in EEPROM.....	126
Summary	136

Chapter 5 : Tactile Navigation with Whiskers	143
Tactile Navigation	143
Activity #1 : Building and Testing the Whiskers	144
Activity #2 : Field Testing the Whiskers	152
Activity #3 : Navigation with Whiskers	155
Activity #4 : Artificial Intelligence and Deciding When You're Stuck.....	160
Summary	165
Chapter 6 : Light-Sensitive Navigation with Phototransistors.....	169
Introducing the Phototransistor.....	169
Activity #1 : A Simple Binary Light Sensor	171
Activity #2 : Measure Light Levels with Phototransistors.....	179
Activity #3 : Light Sensitivity Adjustment	189
Activity #4 : Light Measurements for Roaming	194
Activity #5 : Routine for Roaming Toward Light	203
Activity #6 : Test Navigation Routine with the Boe-Bot.....	212
Summary	216
Chapter 7 : Navigating with Infrared Headlights.....	221
Infrared Light	221
Activity #1 : Building and Testing the IR Object Detectors	223
Activity #2 : Field Testing for Object Detection and Infrared Interference	230
Activity #3 : Infrared Detection Range Adjustments	234
Activity #4 : Object Detection and Avoidance	237
Activity #5 : High-Performance IR Navigation	239
Activity #6 : The Drop-Off Detector.....	242
Summary	248
Chapter 8 : Robot Control with Distance Detection	255
Determining Distance with the Same IR LED/Detector Circuit	255
Activity #1 : Testing the Frequency Sweep	255
Activity #2 : Boe-Bot Shadow Vehicle	262
Activity #3 : Following a Stripe.....	271
Activity #4 : More Boe-Bot Activities and Projects Online.....	278
Summary	280
Appendix A : Parts List and Kit Options.....	289
Appendix B : Resistor Color Codes and Breadboarding Rules.....	293
Appendix C : Boe-Bot Navigation Contests.....	299
Index	303

Preface

Robots are used in the auto, medical, and manufacturing industries, in all manner of exploration vehicles, and, of course, in many science fiction films. The word "robot" first appeared in a Czechoslovakian satirical play, Rossum's Universal Robots, by Karel Capek in 1920. Robots in this play tended to be human-like. From this point onward, it seemed that many science fiction stories involved these robots trying to fit into society and make sense out of human emotions. This changed when General Motors installed the first robots in its manufacturing plant in 1961. These automated machines presented an entirely different image from the "human form" robots of science fiction.

Building and programming a robot is a combination of mechanics, electronics, and problem solving. What you're about to learn while doing the activities and projects in this text will be relevant to real-world applications that use robotic control, the only differences being the size and sophistication. The mechanical principles, example program listings, and circuits you will use are very similar to, and sometimes the same as, industrial applications developed by engineers.

The goal of this text is to get students interested in and excited about the fields of engineering, mechatronics, and software development as they design, construct, and program an autonomous robot. This series of hands-on activities and projects will introduce students to basic robotic concepts using the Parallax Boe-Bot[®] robot, called the "Boe-Bot." Its name comes from the Board of Education[®] carrier board that is mounted on its wheeled chassis. An example of a Boe-Bot with an infrared obstacle detection circuit built on the Board of Education solderless prototyping area is shown below in Figure P-1.

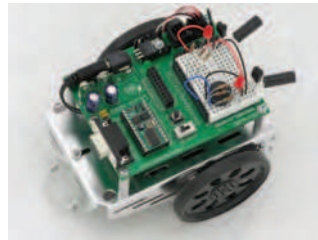


Figure P-1
Parallax Inc.'s
Boe-Bot[®] Robot

The activities and projects in this text begin with an introduction to your Boe-Bot's brain, the Parallax BASIC Stamp[®] 2 microcontroller, and then move on to construction, testing,

and calibration of the Boe-Bot. After that, you will program the Boe-Bot for basic maneuvers, and then proceed to adding sensors and writing programs that make it react to its surroundings and perform autonomous tasks.

ABOUT VERSION 3.0

This is the first revision of this title since 2004. The major changes include:

- Replacement of the cadmium sulfide photoresistor with an RoHS-compliant light sensor of a type that will be more common in product design going forward. This required a rewrite of Chapter 6.
- Moving the “Setup and Testing” portion of Chapter 1 and the Hardware and Troubleshooting appendices to the Help file. This was done to support both serial and USB hardware connections, and other programming connections as our products and technologies continue to expand. This also allows for the dynamic maintenance of the Hardware and Troubleshooting material.
- Removal of references to the Parallax CD, which has been removed from our kits, reducing waste and ensuring that customers download the most recent BASIC Stamp Editor software and USB drivers available for their operating systems (www.parallax.com/go/Boe-Bot).

In addition, small errata items noted in the previous version (2.2) have been corrected. The material still aims for the same goals, and all of the same programming concepts and commands are covered, along with a few new ones. Finally, page numbers have been changed so the PDF page and the physical page numbers are the same, for ease of use.

AUDIENCE

This text is designed to be an entry point to technology literacy, and an easy learning curve for embedded programming and introductory robotics. The text is organized so that it can be used by the widest possible variety of students as well as independent learners. Middle-school students can try the examples in this text in a guided tour fashion by simply following the check-marked instructions with instructor supervision. At the other end of the spectrum, pre-engineering students’ comprehension and problem-solving skills can be tested with the questions, exercises and projects (with solutions) in each chapter summary. The independent learner can work at his or her own pace, and obtain assistance through the Stamps in Class forum cited below.

SUPPORT FORUMS

Parallax maintains free, moderated forums for our customers, covering a variety of subjects:

- Propeller Chip: for all discussions related to the multicore Propeller microcontroller and development tools product line.
- BASIC Stamp: Project ideas, support, and related topics for all of the Parallax BASIC Stamp models.
- Sensors: Discussion relating to Parallax's wide array of sensors, and interfacing sensors with Parallax microcontrollers.
- Stamps in Class: Students, teachers, and customers discuss Parallax's education materials and school projects here.
- Robotics: For all Parallax robots and custom robots built with Parallax processors and sensors.
- Wireless: Topics include XBee, GSM/GPRS, telemetry and data communication over amateur radio.
- PropScope: Discussion and technical assistance for this USB oscilloscope that contains a Propeller chip.
- The Sandbox: Topics related to the use of Parallax products but not specific to the other forums.
- Projects: Post your in-process and completed projects here, made from Parallax products.

RESOURCES FOR EDUCATORS

We have a variety of resources for this text designed to support educators.

Stamps in Class “Mini Projects”

To supplement our texts, we provide a bank of projects for the classroom. Designed to engage students, each “Mini Project” contains full source code, “How it Works” explanations, schematics, and wiring diagrams or photos for a device a student might like to use. Many projects feature an introductory video, to promote self-study in those students most interested in electronics and programming. Just follow the Stamps in Class “Mini Projects” link at www.parallax.com/Education.

Educators Courses

These hands-on, intensive 1 or 2 day courses for instructors are taught by Parallax engineers or experienced teachers who are using Parallax educational materials in their classrooms. Visit www.parallax.com/Education → Educators Courses for details.

Parallax Educator’s Forum

In this free, private forum, educators can ask questions and share their experiences with using Parallax products in their classrooms. Supplemental education materials are also posted here. To enroll, email education@parallax.com for instructions; proof of status as an educator will be required.

Supplemental Educational Materials

Select Parallax educational texts have an unpublished set of questions and solutions posted in our Parallax Educators Forum; we invite educators to copy and modify this material at will for the quick preparation of homework, quizzes, and tests. PowerPoint presentations and test materials prepared by other educators may be posted here as well.

Copyright Permissions for Educational Use

No site license is required for the download, duplication and installation of Parallax software for educational use with Parallax products on as many school or home computers as needed. Our Stamps in Class texts and BASIC Stamp Manual are all available as free PDF downloads, and may be duplicated as long as it is for educational use exclusively with Parallax microcontroller products and the student is charged no more than the cost of duplication. The PDF files are not locked, enabling selection of text and images to prepare handouts, transparencies, or PowerPoint presentations.

FOREIGN TRANSLATIONS

Many of our Stamps in Class texts have been translated into other languages; these texts are free downloads and subject to the same Copyright Permissions for Educational Use as our original versions. To see the full list, click on the Tutorials & Translations link at www.parallax.com/Education. These were prepared in coordination with the Parallax Volunteer Translator program. If you are interested in participating in our Volunteer Translator program, email translations@parallax.com.

ABOUT THE AUTHOR

Andy Lindsay joined Parallax Inc. in 1999, and has since authored eleven books and numerous articles and product documents for the company. The last three versions of *Robotics with the Boe-Bot* were designed and updated based on observations and educator feedback that Andy collected while traveling the nation and abroad teaching Parallax Educator Courses and events. Andy studied Electrical and Electronic Engineering at California State University, Sacramento, and is a contributing author to several papers that address the topic of microcontrollers in pre-engineering curricula. When he's not writing educational material, Andy does product and application and product engineering for Parallax.

SPECIAL CONTRIBUTORS

The Parallax team assembled to prepare this edition includes: excellent department leadership by Aristides Alvarez, lesson design and technical writing by Andy Lindsay; cover art by Jen Jacobs; graphic illustrations by Rich Allred and Andy Lindsay; nitpicking, editing, and layout by Stephanie Lindsay. Special thanks go to Ken Gracey, founder of the Stamps in Class program, and to Tracy Allen and Phil Pilgrim for consulting in the selection of the light sensor used in this version to replace the cadmium-sulfide photoresistor. Stephanie is particularly grateful to John Kauffman for his last-minute review of the revised Chapter 6.

Chapter 1: Your Boe-Bot's Brain

Parallax, Inc's Boe-Bot® robot is the focus of the activities, projects, and contests in this book. The Boe-Bot and a close-up of its BASIC Stamp® 2 programmable microcontroller brain are shown in Figure 1-1. The BASIC Stamp 2 module is both powerful and easy to use, especially with a robot.

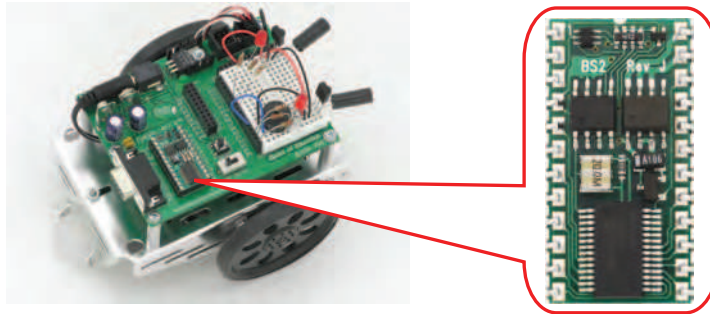


Figure 1-1
BASIC Stamp
Module on a
Boe-Bot Robot

The activities in this text will guide you through writing simple programs that make the BASIC Stamp and your Boe-Bot do four essential robotic tasks:

1. Monitor sensors to detect the world around it
2. Make decisions based on what it senses
3. Control its motion (by operating the motors that make its wheels turn)
4. Exchange information with its Roboticist (that will be you!)



The programming language you will use to accomplish these tasks is called **PBASIC**, which stands for:

- Parallax - Company that invented and manufactures BASIC Stamp microcontrollers
- Beginners - Made for beginners to learn how to program computers
- All-purpose - Powerful and useful for solving many different kinds of problems
- Symbolic - Using symbols (terms that resemble English word/phrases)
- Instruction - To tell a computer what to do
- Code - In terms that the computer (and you) can understand



What's a Microcontroller? It's a programmable device that is designed into your digital wristwatch, cell phone, calculator, clock radio, etc. In these devices, the microcontroller has been programmed to sense when you press a button, make electronic beeping noises, and control the device's digital display. They are also built into factory machinery, cars, submarines, and spaceships because they can be programmed to read sensors, make decisions, and orchestrate devices that control moving parts.

The *What's a Microcontroller?* Student Guide is the recommended first text for beginners. It is full of examples of how to use microcontrollers, and how to make the BASIC Stamp the brain of your own microcontrolled inventions. It's available for free download from www.parallax.com/go/WAM, and it's also included in the BASIC Stamp Editor Help as a PDF file. It is included in the BASIC Stamp Activity Kit and BASIC Stamp Discovery Kit, which are carried by many electronic retailers. These kits can also be purchased directly from Parallax, either online at www.parallax.com/go/WAM or by phone at (888) 512-1024.

HARDWARE AND SOFTWARE

Getting started with BASIC Stamp microcontroller modules is similar to getting started with a brand-new PC or laptop. The first things that most people have to do is take it out of the box, plug it in, install and test some software, and maybe even write some software of their own using a programming language. If this is your first time using a BASIC Stamp module, you will be doing all these same activities. If you are in a class, your hardware may already be all set up for you. If this is the case, your teacher may have other instructions. If not, this chapter will take you through all the steps of getting your new BASIC Stamp microcontroller up and running.

ACTIVITY #1: GETTING THE SOFTWARE

The BASIC Stamp Editor (version 2.5 or higher) is the software you will use in most of the activities and projects in this text. You will use this software to write programs that the BASIC Stamp module will run. You can also use this software to display messages sent by the BASIC Stamp that help you understand what it senses.

Computer System Requirements

You will need a personal computer to run the BASIC Stamp Editor software. Your computer will need to have the following features:

- Microsoft Windows 2K/XP/Vista/7 or newer operating system
- An available serial or USB port
- Internet access and an Internet browser program

Downloading the Software from the Internet

It is important to always use the latest version of the BASIC Stamp Editor software if possible. The first step is to go to the Parallax web site and download the software.

- ✓ Using a web browser, go to www.parallax.com/basicstampsoftware.

Figure 1-2: BASIC Stamp Editor download page at www.parallax.com/basicstampsoftware



Use the “Click Here to Download” button to get the latest version of the software.

- ✓ Click on the [Click Here to Download](#) button to download the latest version of the BASIC Stamp Windows Editor software.

- ✓ A File Download window will open, asking you if you want to run or to save this file (Figure 1-3). Click Save.

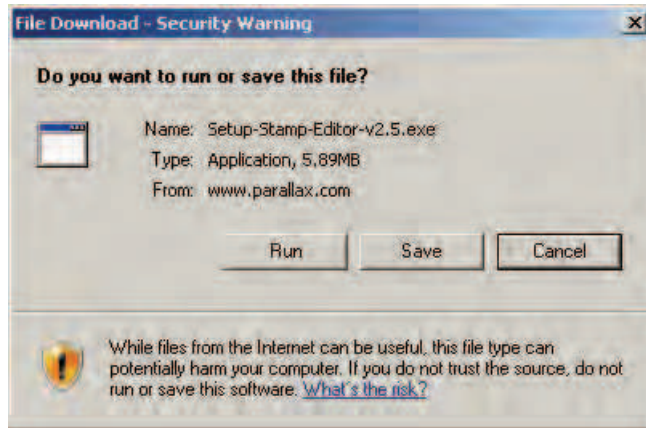


Figure 1-3
File Download Window

Click Save, then save the file to your computer.

- ✓ Follow the prompts that appear. When the download is complete, click Run. You may see messages from your operating system asking you to verify that you wish to continue with installation. Always agree that you want to continue.

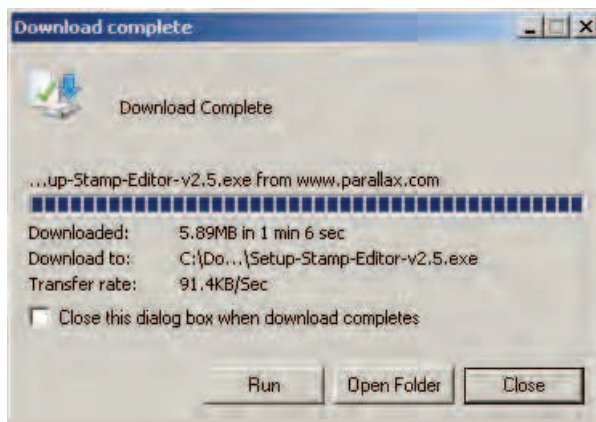


Figure 1-4
Download Complete Message

Click Run.

If prompted, always confirm you want to continue.

- ✓ The BASIC Stamp Editor Installer window will open (Figure 1-5). Click Next and follow the prompts, accepting all defaults.



Figure 1-5
BASIC Stamp Editor
Installer Window

Click Next.

- ✓ **IMPORTANT:** When the “Install USB Driver” message appears (Figure 1-6), leave the checkmark in place for the Automatically install/update driver (recommended) box, and then click Next.

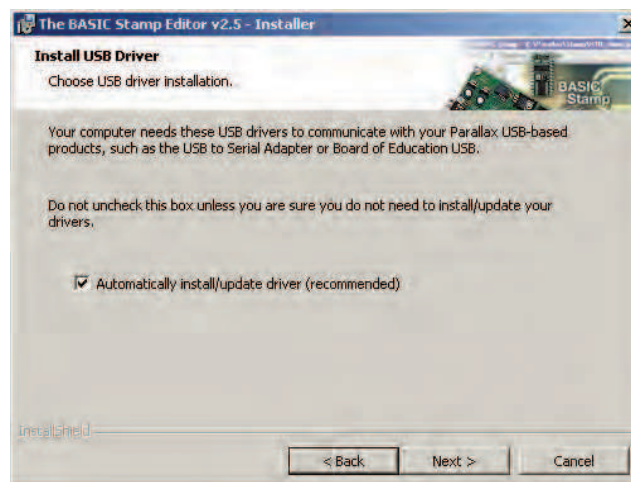



Figure 1-6
Install USB Driver
Message

Leave the box
checked, and click
Next.

- ✓ When the “Ready to Install the Program” message appears, click the Install button. A progress bar may appear, and this could take a few minutes.

At this point, an additional window may appear behind the current window while the USB drivers are updating. This window will eventually close on its own when the driver installation is complete. If you don’t see this window, it does not indicate a problem.

 **About USB drivers.** The USB drivers that install with the BASIC Stamp Windows Editor installer by default are necessary to use any Parallax hardware connected to your computer’s USB port. VCP stands for Virtual COM Port, and it will allow your computer’s USB port to look and be treated as a standard RS232 serial port by Parallax hardware.

USB Drivers for Different Operating Systems The USB VCP drivers included in the BASIC Stamp Windows Editor software are for certain Windows operating systems only. For more information, visit www.parallax.com/usbdrivers.

- ✓ When the window tells you that installation has been successfully completed, click Finish (Figure 1-7).



Figure 1-7
BASIC Stamp
Editor Installation
Completed

Click Finish.

ACTIVITY #2: USING THE HELP FILE FOR HARDWARE SETUP

In this section you will run the BASIC Stamp Editor's Help file. Within the Help file, you will learn about the different BASIC Stamp programming boards available for the Stamps in Class program, and determine which one you are using. Then, you will follow the steps in the Help to connect your hardware to your computer and test your BASIC Stamp programming system.

Running the BASIC Stamp Editor for the first time

- ✓ If you see the BASIC Stamp Editor icon on your computer desktop, double-click it (Figure 1-8).
- ✓ Or, click on your computer's Start menu, then choose All Programs ▶ Parallax Inc ▶ BASIC Stamp Editor 2.5 ▶ BASIC Stamp Editor 2.5.



Figure 1-8
BASIC Stamp Editor
Desktop Icon

*Double-click to launch
the program.*

- ✓ On the BASIC Stamp Editor's toolbar, click Help on the toolbar (Figure 1-9) and then select BASIC Stamp Help... from the drop-down menu.

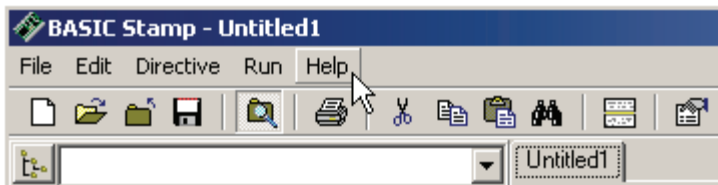


Figure 1-9
Opening the Help Menu

*Click Help, then choose
BASIC Stamp Help from
the drop-down menu.*

Figure 1-10: BASIC Stamp Editor Help



- ✓ Click on the [Getting Started with Stamps in Class](#) link on the bottom of the Welcome page, as shown in the lower right corner of Figure 1-10.

Following the Directions in the Help File

From here, you will follow the directions in the Help file to complete these tasks:

- Identify which BASIC Stamp development board you are using
- Connect your development board to your computer
- Test your programming connection
- Troubleshoot your programming connection, if necessary
- Write your first PBASIC program for your BASIC Stamp
- Power down your hardware when you are done

When you have completed the activities in the Help file, return to this book and continue with the Summary below before moving on to Chapter 2.

What do I do if I get stuck?

If you run into problems while following the directions in this book or in the Help file, you have many options to obtain free Technical Support:



- **Forums:** sign up and post a message in our free, moderated Stamps in Class forum at forums.parallax.com.
- **Email:** send an email to support@parallax.com.
- **Telephone:** In the Continental United States, call toll-free to 888-99-STAMP (888-997-8267). All others call (916) 624-8333.
- **More resources:** Visit www.parallax.com/support.

SUMMARY

This chapter guided you through the following:

- An introduction to the BASIC Stamp module
- Where to get the free BASIC Stamp Editor software you will use in just about all of the experiments in this text
- How to install the BASIC Stamp Editor software
- How to use the BASIC Stamp Editor's Help and the BASIC Stamp Manual
- An introduction to the BASIC Stamp module, Board of Education, and HomeWork Board
- How to set up your BASIC Stamp hardware
- How to test your software and hardware
- How to write and run a PBASIC program
- Using the **DEBUG** and **END** commands, **CR** control character, and **DEC** formatter.

- A brief introduction to ASCII code
- How to disconnect the power to your Board of Education or HomeWork Board when you're done

Questions

1. What device will be the brain of your Boe-Bot?
2. When the BASIC Stamp sends a character to your PC/laptop, what type of numbers are used to send the message through the programming cable?
3. What is the name of the window that displays messages sent from the BASIC Stamp to your PC/laptop?
4. What PBASIC commands did you learn in this chapter?

Exercises

1. Explain what the asterisk does in this command: `DEBUG DEC 7 * 11`
2. Guess what the Debug Terminal would display if you ran this command: `DEBUG DEC 7 + 11`
3. There is a problem with these two commands. When you run the code, the numbers they display are stuck together so that it looks like one large number instead of two small ones. Modify these two commands so that the answers appear on different lines in the Debug Terminal.

```
DEBUG DEC 7 * 11
DEBUG DEC 7 + 11
```

Projects

1. Use `DEBUG` to display the solution to the math problem: $1 + 2 + 3 + 4$.
2. Save FirstProgramYourTurn.bs2 under another name. If you were to place the `DEBUG` command shown below on the line just before the `END` command in the program, what other lines could you delete and still have it work the same? Modify the copy of the program to test your hypothesis (your prediction of what will happen).

```
DEBUG "What's 7 X 11?", CR, "The answer is: ", DEC 7 * 11
```

Solutions

- Q1. A BASIC Stamp 2 microcontroller module.
 Q2. Binary numbers, that is, 0's and 1's.
 Q3. The Debug Terminal.
 Q4. **DEBUG** and **END**
 E1. It multiplies the two operands 7 and 11, resulting in a product of 77. The asterisk is the multiply operator.
 E2. The Debug Terminal would display: 18
 E3. To fix the problem, add a carriage return using the **CR** control character and a comma.

```
DEBUG DEC 7 * 11
DEBUG CR, DEC 7 + 11
```

- P1. Here is a program to display a solution to the math problem: 1+2+3+4.

```
' What's a Microcontroller - Ch01Prj01_Add1234.bs2
'{$STAMP BS2}
'{$PBASIC 2.5}

DEBUG "What's 1+2+3+4?"
DEBUG CR, "The answer is: "
DEBUG DEC 1+2+3+4

END
```

- P2. The last three **DEBUG** lines can be deleted. An additional **CR** is needed after the "Hello" message.

```
' What's a Microcontroller - Ch01Prj02_FirstProgramYourTurn.bs2
' BASIC Stamp sends message to Debug Terminal.

' {$STAMP BS2}
' {$PBASIC 2.5}

DEBUG "Hello, it's me, your BASIC Stamp!", CR
DEBUG "What's 7 X 11?", CR, "The answer is: ", DEC 7 * 11

END
```

The output from the Debug Terminal is:

```
Hello, it's me, your BASIC Stamp!
What's 7 X 11?
The answer is: 77
```

This output is the same as it was with the previous code. This is an example of using commas to output a lot of information, using only one **DEBUG** command with multiple elements in it.

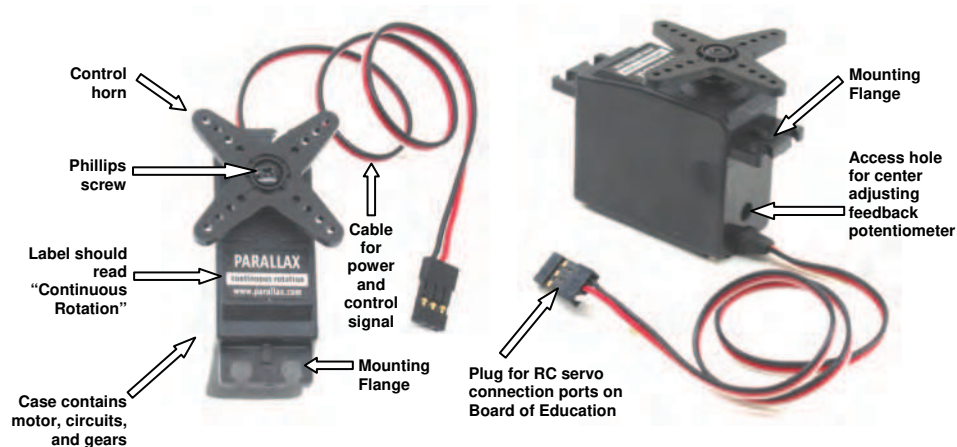
Chapter 2: Your Boe-Bot's Servo Motors

This chapter will guide you through connecting, adjusting, and testing the Boe-Bot's motors. In order to do that, you will need to understand certain PBASIC commands and programming techniques that will control the direction, speed, and duration of servo motions. Therefore, Activities #1, #2, and #5 will introduce you to these programming tools, and then Activities #3, #4, and #6 will show you how to apply them to the servos. Since precise servo control is key to the Boe-Bot's performance, completing these activities before mounting the servos into the Boe-Bot chassis is both important and necessary!

INTRODUCING THE CONTINUOUS ROTATION SERVO

The Parallax Continuous Rotation servos shown in Figure 2-1 are the motors that will make the Boe-Bot's wheels turn. This figure points out the servos' external parts. Many of these parts will be referred to as you go through the instructions in this and the next chapter.


Figure 2-1 Parallax Continuous Rotation Servo



Note: You might find it useful to bookmark this page so that you can refer back to it later.

Standard Servos vs. Continuous Rotation Servos: Standard servos are designed to receive electronic signals that tell them what position to hold. These servos control the positions of radio controlled airplane flaps, boat rudders, and car steering. Continuous rotation servos receive the same electronic signals, but instead of holding certain positions, they turn at certain speeds and directions. Continuous rotation servos are ideal for controlling wheels and pulleys.

Servo Control Horn - 4-point Star vs. Round: It doesn't make a difference. So long as it is labeled "continuous rotation" it's the servo for your Boe-Bot. You will be removing the control horn with a wheel.



The image shows two Parallax Boe-Bot servos side-by-side. The one on the left has a black control horn with a white 4-pointed star shape in the center. The one on the right has a black control horn with a white circle in the center. Both servos are black with 'PARALLAX' and 'www.parallax.com' printed on them.

ACTIVITY #1: BUILDING AND TESTING THE LED CIRCUIT

Controlling a servo motor's speed and direction involves a program that makes the BASIC Stamp send the same message, over and over again. The message has to repeat itself around 50 times per second for the servo to maintain its speed and direction. This activity has a few PBASIC example programs that demonstrate how to repeat the same message over and over again and control the timing of the message.

Displaying Messages at Human Speeds

You can use the **PAUSE** command to tell the BASIC Stamp to wait for a while before executing the next command.

PAUSE Duration

The number that you put to the right of the **PAUSE** command is called the **Duration** argument, and it's the value that tells the BASIC Stamp how long it should wait before moving on to the next command. The units for the **Duration** argument are thousandths of a second (ms). So, if you want to wait for one second, use a value of 1000. Here's how the command should look:

```
PAUSE 1000
```

If you want to wait for twice as long, try:

```
PAUSE 2000
```



A second is abbreviated “s.” In this text, when you see 1 s, it means one second.

A millisecond is one thousandth of a second, and it is abbreviated “ms.” The command **PAUSE 1000** delays the program for 1000 ms, which is 1000/1000 of a second, which is one second, or 1 s. Got it?

2

Example Program: TimedMessages.bs2

There are lots of different ways to use the **PAUSE** command. This example program uses **PAUSE** to delay between printing messages that tell you how much time has elapsed. The program should wait one second before it sends the “One second elapsed...” message and another two seconds before it displays the “Three seconds elapsed . . .” message.

- ✓ If you have a Board of Education, move the 3-position switch from position-0 to position-1.
- ✓ If you have a HomeWork Board, reconnect the 9 V battery to the battery clip.
- ✓ Enter the program below into the BASIC Stamp Editor.
- ✓ Save the program under the name TimedMessages.bs2.
- ✓ Run the program, and then watch for the delay between messages.

```
' Robotics with the Boe-Bot - TimedMessages.bs2
' Show how the PAUSE command can be used to display messages at human speeds.

' {$STAMP BS2}
' {$PBASIC 2.5}

DEBUG "Start timer..."

PAUSE 1000
DEBUG CR, "One second elapsed..."

PAUSE 2000
DEBUG CR, "Three seconds elapsed..."

DEBUG CR, "Done."

END
```



From here onward, the three instructions that came before this program will be phrased like this:

- ✓ Enter, save, and run TimedMessages.bs2.