



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




Propeller Education Kit Labs

Fundamentals

Version 1.2

(web release 2)

By Andy Lindsay

PARALLAX 

WARRANTY

Parallax Inc. 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 Inc. 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 Inc. 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 © 2006-2010 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 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 is permitted, subject to the following Conditions of Duplication: Parallax Inc. grants the user a conditional right to download, duplicate, and distribute this text without Parallax's permission. This right is based on the following conditions: the text, or any portion thereof, may not be duplicated for commercial use; it may be duplicated only for educational purposes when used solely in conjunction with Parallax products, and the user may recover from the student only the cost of duplication.

This text is available in printed format from Parallax Inc. Because we print the text in volume, the consumer price is often less than typical retail duplication charges.

Propeller, Penguin, and Spin are trademarks of Parallax Inc. BASIC Stamp, Stamps in Class, Boe-Bot, SumoBot, Scribbler, Toddler, and SX-Key are registered trademarks of Parallax, Inc. If you decide to use any trademarks of Parallax Inc. on your web page or in printed material, you must state that (trademark) is a (registered) trademark of Parallax Inc." upon the first appearance of the trademark name in each printed document or web page. Other brand and product names herein are trademarks or registered trademarks of their respective holders.

ISBN 9781928982555

1.2.0-10.07.12-HKTP — (WEB RELEASE 2)

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 Inc. 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 Propeller microcontroller application, no matter how life-threatening it may be.

INTERNET DISCUSSION LISTS

We maintain active web-based discussion forums for people interested in Parallax products. These lists are accessible from www.parallax.com via the Support → Discussion Forums menu. These are the forums that we operate from our web site:

- [Propeller Chip](#) – This list is specifically for our customers using Propeller chips and products.
- [BASIC Stamp](#) – This list is widely utilized by engineers, hobbyists and students who share their BASIC Stamp projects and ask questions.
- [Stamps in Class](#)[®] – Created for educators and students, subscribers discuss the use of the Stamps in Class series of tutorials in their courses. The list provides an opportunity for both students and educators to ask questions and get answers.
- [Parallax Educators](#) – A private forum exclusively for educators and those who contribute to the development of Stamps in Class and Propeller Education materials. Parallax created this group to obtain feedback on our educational materials and to provide a place for educators to develop and share classroom resources.
- [Robotics](#) – Designed for Parallax robots, this forum is intended to be an open dialogue for robotics enthusiasts using the Boe-Bot[®], SumoBot[®], Scribbler[®] or their own custom robots built with Parallax microcontrollers and sensors..
- [Sensors](#) – A place to discuss interfacing Parallax sensors to microcontrollers.
- [PropScope](#) – Discussion and technical assistance for using this PC-based digital storage oscilloscope and logic analyzer, designed with a Propeller P8X32A on board.
- [HYDRA Game Development](#) – For creating and sharing games on this Propeller P8X32A-based system.

ERRATA

While great effort is made to assure the accuracy of our texts, errors may still exist. If you find an error, please let us know by sending an email to editor@parallax.com. We continually strive to improve all of our educational materials and documentation, and frequently revise our texts. Occasionally, an errata sheet with a list of known errors and corrections for a given text will be posted to our web site, www.parallax.com. Please check the individual product page's free downloads for an errata file.

Table of Contents

PREFACE..... 5

1: PROPELLER MICROCONTROLLER & LABS OVERVIEW 7

 The Propeller Microcontroller 7

 The Propeller Education Kit Hardware..... 12

 The Propeller Education Kit Labs..... 14

2: SOFTWARE, DOCUMENTATION & RESOURCES 17

 Download Software and Documentation..... 17

 Useful Web Sites..... 18

 Tech Support Resources 18

3: SETUP AND TESTING LAB FOR 40-PIN DIP PE PLATFORM 19

 The PE Platform 19

 Procedure Overview..... 23

 Inventory Equipment and Parts..... 24

 Assemble the Breadboards..... 25

 Set up PE Platform Wiring and Voltage Regulators..... 27

 Test the PE Platform Wiring..... 29

 Socket the Propeller Chip and EEPROM..... 30

 Load a Test Program and Test the I/O Pins 32

 Before Changing or Adjusting Circuits 37

 Propeller Supply Voltage Regulation – It’s Important! 37

 Troubleshooting for the 40-Pin DIP PE Platform Setup..... 39

4: I/O AND TIMING BASICS LAB..... 45

 Introduction..... 45

 Propeller Nomenclature..... 46

 Lights on with Direction and Output Register Bits..... 47

 I/O Pin Group Operations..... 49

 Reading an Input, Controlling an Output..... 50

 Timing Delays with the System Clock..... 51

 System Clock Configuration and Event Timing..... 53

 More Output Register Operations 55

 Conditional Repeat Commands 57

 Operations in Conditions and Pre and Post Operator Positions..... 58

 Some Operator Vocabulary..... 60

 Shifting LED Display..... 61

 Variable Example 62

 Timekeeping Applications 64

 Study Time 66

5: METHODS AND COGS LAB..... 69

 Introduction..... 69

 Defining a Method’s Behavior with Local Variables 70

 Calling a Method 70

 Launching Methods into Cogs..... 73

 How Much Stack Space for a Method Launched into a Cog? 75

 Method Calls and the Result Variable..... 77

 Cog ID Indexing..... 78

 Study Time 80

Table of Contents

6: OBJECTS LAB	83
Introduction	83
Method Call Review	85
Calling Methods in Other Objects with Dot Notation	85
Objects that Launch Processes into Cogs	88
Conventions for Start and Stop Methods in Library Objects	92
Documentation Comments	92
Public vs. Private methods	95
Multiple Object Instances	96
Propeller Chip – PC Terminal Communication	97
Parallax Serial Terminal.spin and Other Library Objects	102
Sending Values from Parallax Serial Terminal to the Propeller Chip	106
Terminal I/O Pin Input State Display	109
Terminal LED Output Control	111
The DAT Block and Address Passing	112
The Float and FloatString Objects	114
Objects that Use Variable Addresses	115
Passing Starting Addresses to Objects that Work with Variable Lists	117
Study Time	120
7: COUNTER MODULES AND CIRCUIT APPLICATIONS LAB	125
Introduction	125
How Counter Modules Work	126
Measuring RC Decay with a Positive Detector Mode	126
D/A Conversion – Controlling LED Brightness with DUTY Modes	135
Special Purpose Registers	140
Generating Piezospeaker Tones with NCO Mode	143
Applications - IR Object and Distance Detection with NCO and DUTY Modes	153
Counting Transitions with POSEDGE and NEGEDGE Modes	158
PWM with the NCO Modes	162
Probe and Display PWM – Add an Object, Cog and Pair of Counters	165
PLL Modes for High-Frequency Applications	171
Metal Detection with an LC Circuit Using PLL and POS Detector Modes	176
Study Time	185
APPENDIX A: OBJECT CODE LISTINGS	191
Parallax Serial Terminal.spin	191
SquareWave.spin	200
APPENDIX B: STUDY SOLUTIONS	201
I/O and Timing Basics Lab Study Solutions	201
Methods and Cogs Lab Study Solutions	207
Objects Lab Study Solutions	209
Counter Modules and Circuit Applications Lab Study Solutions	214
APPENDIX C: PE KIT COMPONENTS LISTING	224
APPENDIX D: PROPELLER P8X32A BLOCK DIAGRAM	226
APPENDIX E: LM2940CT-5.0 CURRENT LIMIT CALCULATIONS	227
INDEX	229

Preface

Since the Propeller chip comes in a 40-Pin DIP package, a pluggable breadboard kit for the Propeller chip made a lot of sense. The support circuits for the Propeller chip, including EEPROM program memory, voltage regulators, crystal oscillator, and Propeller Plug programming tool are all also available in versions that can be plugged into a breadboard, so why not? It also makes a great deal of sense from the college and university lab standpoint. Provide a simple kit that students can afford, that is reusable, with a microcontroller that excels in a multitude of electronics, robotics, and embedded systems projects. With that in mind, the PE DIP Plus Kit was put together, as a bag that includes the Propeller microcontroller, “plus” all the other parts you might need to make it work.

The PE DIP Plus Kit made sense for folks who have already have breadboards and some experience, but what about a student who maybe just completed the Stamps in Class *What’s a Microcontroller* tutorial, and is interested in approaching the Propeller chip as a kit and tutorial as well? With this student in mind, another bag of parts was assembled, along with a series of activities that put the parts in the bag to work with the Propeller microcontroller. The bag of parts ended up with the name PE Project Parts, and the activities became the PE Kit Labs.

The PE Kit Labs in this text are written primarily for college and university students with some previous programming and electronics experience, preferably with microcontrollers. Subjects introduced include:

- Microcontroller basics such as I/O control and timing with the system clock
- Programming topics such as operators, method calls, and objects, and variable addresses
- Programmed multiprocessor control
- Microcontroller-circuit interactions with indicator lights, pushbuttons, circuits that sense the environment and can be measured with RC decay, frequency circuits (speakers), and frequency selective circuits
- Advanced topics include utilizing counter modules to perform tasks in the background

This collection of PE Kit Labs is intended give the reader a good start with programming the Propeller chip and using it in projects. However, this book is just a start. Introducing all aspects of the Propeller microcontroller with PE Kit Labs would take several such books, so additional labs are available online. More labs and applications will be posted periodically.

This text also includes pointers to the wealth of information available for the Propeller chip in the Propeller Manual, Propeller Datasheet, Propeller Forum, and Propeller Object Exchange, as well as examples of using these resources. The reader is especially encouraged to utilize the Propeller Manual as a reference while going through these labs. The Propeller Manual’s contents and index will provide references to more information about any topic introduced in these labs.

The Propeller Chip Forum at forums.parallax.com has a Propeller Education Kit Labs sticky-thread with links to discussions about each lab. The reader is encouraged to utilize this resource for posting questions about topics in the PE Kit Labs as well as comments and suggestions. Parallax collects this feedback and incorporates it into future revisions of each lab. Also, if you (or your students) prototyped something cool with the PE Kit, by all means, post your documented project to the forums so that others can see what you did and how you did it.

Preface

About Version 1.2

This revision replaces the cadmium sulfide based photoresistor with a cadmium-free phototransistor that meets the European Union's Restriction of Hazardous Substances guidelines.

The Setup and Testing instructions include additions for enhanced Propeller supply voltage stability.

In addition, the Parallax Serial Terminal object replaces the FullDuplexSerial Plus object.

Acknowledgements

Parallaxians:

- Author: Andy Lindsay, Applications Engineer
- Cover art: Jennifer Jacobs, Art Director
- Editing: Stephanie Lindsay, Technical Editor
- Illustrations: Andy Lindsay, with help from Rich Allred, Manufacturing Lead
- Photography: Rich Allred
- Review: Jessica Uelmen, Education Associate

Parallax Community – thanks to:

- Aaron Klapheck for commented code illustrating cog variable bookkeeping in the Advanced Topic: Inside `Start` and `Stop` methods section of the Objects Lab.
- Engineering students at University of California Davis and California State University Sacramento who used the PE Kit in their projects and submitted great questions and bug reports.
- Steve Nicholson for his incisive and thorough review of earlier drafts of the PE Kit Labs.

1: Propeller Microcontroller & Labs Overview

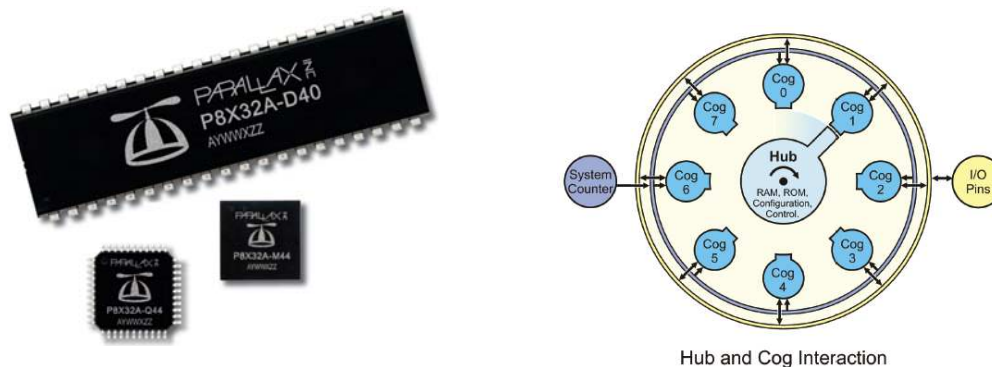
This chapter provides an abbreviated overview of the Propeller Microcontroller and some introductory information about the Propeller Education Kit and Labs. More detailed information about the Propeller microcontroller, its architecture, and programming languages can be found in the *Propeller Manual* and *Propeller Datasheet*. Both are available from the Downloads link at www.parallax.com/Propeller.

The Propeller Microcontroller

The Propeller Microcontroller in Figure 1-1 (a) is a single chip with eight built-in 32-bit processors, called *cogs*. Cogs can be programmed to function simultaneously, both independently and cooperatively with other cogs. In other words, cogs can all function simultaneously, but whether they function independently or cooperatively is defined by the program. Groups of cogs can be programmed to work together, while others work on independent tasks.

A configurable system clock supplies all the cogs with the same clock signal (up to 80 MHz). Figure 1-1 (b) shows how each cog takes turns at the option for exclusive read/write access of the Propeller chip's main memory via the *Hub*. Exclusive read/write access is important because it means that two cogs cannot try to modify the same item in memory at the same instance. It also prevents one cog from reading a particular address in memory at the same time another cog is writing to it. So, exclusive access ensures that there are never any memory access conflicts that could corrupt data.

Figure 1-1: Propeller Microcontroller Packages and Hub and Cog Interaction



(a) Propeller microcontrollers in 40-pin DIP, TSOP and QFN packages

(b) Excerpt from Propeller Block Diagram describing Hub and Cog interaction. See Appendix D: Propeller P8X32A Block Diagram

32 KB of the Propeller chip's main memory is RAM used for program and data storage, and another 32 KB is ROM, and stores useful tables such as log, antilog, sine, and graphic character tables. The ROM also stores boot loader code that cog 0 uses at startup and interpreter code that any cog can use to fetch and execute application code from main memory. Each cog also has the ability to read the states of any or all of the Propeller chip's 32 I/O pins at any time, as well as set their directions and output states at any time.

Propeller Microcontroller & Labs Overview

The Propeller chip's unique multiprocessing design makes a variety of otherwise difficult microcontroller applications relatively simple. For example, processors can be assigned to audio inputs, audio outputs, mouse, keyboard, and maybe a TV or LCD display to create a microcontroller based computer system, with processors left over to work on more conventional tasks such as monitoring inputs and sensors and controlling outputs and actuators. Figure 1-2 (a) shows a Propeller chip-generated video image that could be used in that this kind of application. The Propeller also excels as a robotic controller, with the ability to assign processors to tasks such as PWM DC motor control, video processing, sensor array monitoring, and high speed communication with nearby robots and/or PCs. Figure 1-2 (b) shows an example of a Propeller controlled balancing robot with video sensor. The initial prototype was developed with a Propeller Education Kit.

Although the Propeller chip is very powerful, that doesn't mean it is difficult to use. The Propeller chip also comes in handy for simple projects involving indicator lights, buttons, sensors, speakers, actuators, and smaller displays found in common product designs. You will see examples of such simple circuits in the following Propeller Education Kit Labs.

Figure 1-2: Application Examples



(a) Propeller microcontroller generated graphic TV display. This application also uses a standard PS/2 mouse to control the graphics (not shown).



(b) Hanno Sander's balancing robot, initial prototype developed with the Propeller Education Kit and ViewPort software. Photo courtesy of mydancebot.com.

Applications with the Propeller Chip

Programs for the Propeller chip are written with PC software and then loaded into the Propeller chip, typically via a USB connection. The languages supported by Parallax's free Propeller Tool software include a high-level language called Spin, and a low-level assembly language. Applications developed in Spin language can optionally contain assembly language code. These applications are stored on your PC as .spin files.



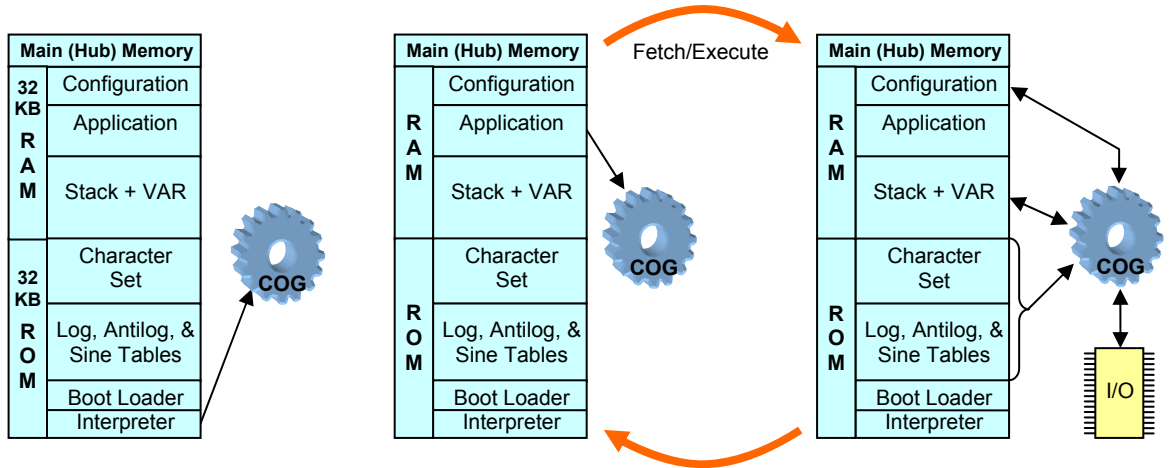
Other programming languages have been developed for programming the Propeller chip. Some are free and available through resources like the Parallax forums and Source Forge; others are available for purchase or free in a limited version through the Parallax web site and other companies that sell compilers.

Before a cog can start executing a Spin application, it has to first load an Interpreter from the Propeller chip's ROM (Figure 1-3 a). Spin applications get stored in main memory's RAM as tokens, which the interpreter code makes the cog repeatedly fetch and execute (Figure 1-3 b & c). A few

1: Propeller Microcontroller & Labs Overview

examples of actions the cog might take based on the token values are shown in Figure 1-3 (c). They include read/writes to configuration registers, variables, and I/O pins as well as reads from ROM. Cogs can also execute the machine codes generated by assembly language. As shown in Figure 1-4, these machine codes get loaded into the cog's 2 KB (512 longs) of RAM and executed at a very high speed, up to 20 million instructions per second (MIPS). Cog RAM not used by machine instructions can also provide high speed memory for the cog with four clock cycles (50 ns at 80 MHz) per read/write.

Figure 1-3: Cog Interpreting Spin Language



(a) Interpreter loaded into cog from Main Memory's ROM through Hub

(b) Cog fetches token from Main Memory's RAM

(c) Cog executes token. Examples include RAM, I/O or config read/write, or ROM read

A cog executing assembly language can also access the Propeller chip's main memory through the Hub. The Hub grants main memory access to each cog every 16 clock cycles. Depending on when the cog decides to check with main memory, the access time could take anywhere from 7 to 22 clock cycles, which equates to a worst case memory access time of 275 ns at 80 MHz. After the first access, assembly code can synchronize with the cog's round-robin access window to main memory, keeping the subsequent access times fixed at 16 clock cycles (200 ns).

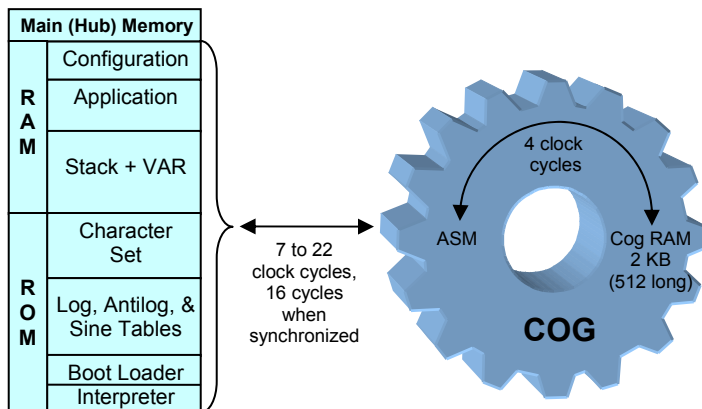


Figure 1-4: Cog Executing Assembly Language

Propeller Microcontroller & Labs Overview

Since each cog has access to the Propeller chip's RAM in main memory, they can work cooperatively by exchanging information. The Spin language has built-in features to pass the addresses of one or more variables used in code to other objects and cogs. This makes cog cooperation very simple. Code in one cog can launch code into another cog and pass it one or more variable addresses (see Figure 1-5). These variable addresses can then be used for the two cogs to exchange information.

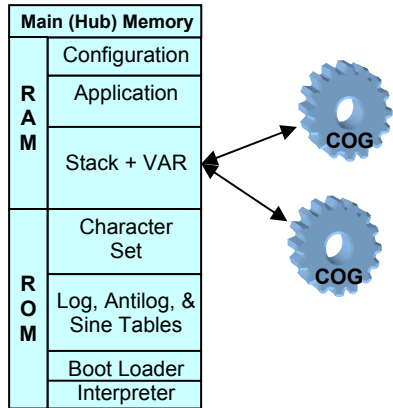


Figure 1-5: Two (or more) Cog's Working Cooperatively through Shared Memory

The Propeller chip's cogs are numbered cog 0 through cog 7. After the application is loaded into the Propeller chip, it loads an interpreter into cog 0, and this interpreter starts executing Spin code tokens stored in main memory. Commands in the Spin code can then launch blocks of code (which might be Spin or assembly language) into other cogs as shown in Figure 1-6. Code executed by the other cogs can launch still other cogs regardless of whether they are Spin or assembly, and both languages can also stop other cogs for the sake of ending unnecessary processes or even replacing them with different ones.

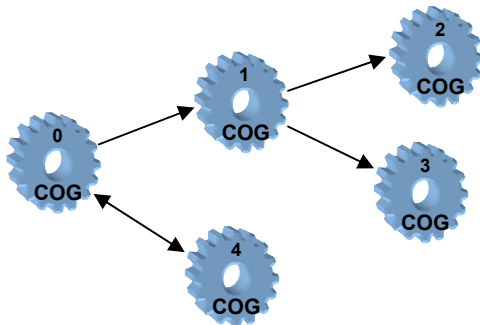


Figure 1-6: Cog Launching

Code in one cog launching other cogs, which can in turn launch others...

Cogs can also stop other cogs to free them up for other tasks.

Writing Application Code

Spin is an object-based programming language. Objects are designed to be the building blocks of an application, and each .spin file can be considered an object. While an application can be developed as a single object (one program), applications are more commonly a collection of objects. These objects can provide a variety of services. Examples include solutions for otherwise difficult coding problems, communication with peripheral devices, controlling actuators and monitoring sensors. These building block objects are distributed through the Propeller Object Exchange (obex.parallax.com) and also in the Propeller Tool software's Propeller Library folder. Incorporating these pre-written objects into an application can significantly reduce its complexity and development time.

1: Propeller Microcontroller & Labs Overview

Figure 1-7 shows how objects can be used as application building blocks, in this case, for a robot that maintains a distance between itself and a nearby object it senses. The application code in the Following Robot.spin object makes use of pre-written objects for infrared detection (IR Detector.spin), control system calculations (PID.spin), and motor drive (Servo Control.spin). Note that these pre-written objects can in turn use other objects to do their jobs. Instead of harvesting objects that do jobs for your application, you can also write them from scratch, and if they turn out to be useful, by all means, submit them for posting to the Propeller Object Exchange at obex.parallax.com.

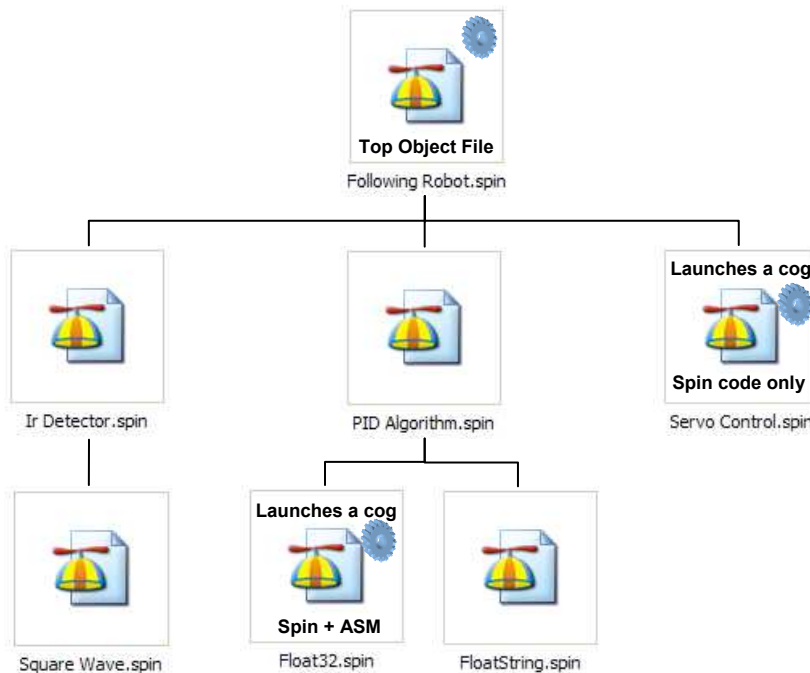


Figure 1-7: Object Building Blocks for Applications

In Figure 1-7, the Following Robot.spin object is called the top object file. This file has the first executable line of code where the Propeller chip starts when the application runs. In every case, cog 0 is launched and begins executing code from the top object. Our top object example, Following Robot.spin, contains code to initialize the three objects below it, making it the “parent object” of the three. Two of these three building blocks in turn initialize “child object” building blocks of their own. Two of the building block objects launch additional cogs to do their jobs, so a total of three cogs are used by this application. Regardless of whether a parent object launches a cog to execute Spin code or assembly code, the child objects have built-in Spin code and documentation that provide a simple interface for code in their parent objects to control/monitor them.

Though it is not shown in our example, recall from Figure 1-6 that an object can launch more than one cog. Also, an object can launch a process into a cog and then shut it down again to make it available to other objects. Although any object can actually start and stop any cog, it's a good practice to make stopping a cog the responsibility of the object that started it.

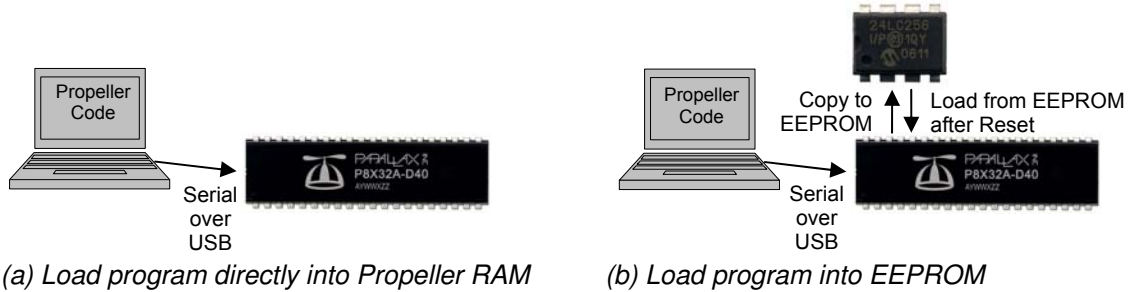
How the Propeller Chip Executes Code

The Parallax Propeller Tool software can be used to develop applications and load them into the Propeller chip. When an application is loaded into the Propeller chip, the Spin code is compiled into tokens and the optional assembly code is compiled into machine codes. The Propeller Tool then

Propeller Microcontroller & Labs Overview

transfers the application to the Propeller chip, typically with a serial-over-USB connection. The programmer can choose to load it directly into the Propeller chip's main RAM, or into an EEPROM (electrically erasable programmable read-only memory). As shown in Figure 1-8, if the program is loaded directly into RAM, the Propeller chip starts executing it immediately. If the program is loaded into an EEPROM, the Propeller chip copies this information to RAM before it starts executing.

Figure 1-8: Loading a Program into RAM or EEPROM



Loading programs from a PC into RAM takes around 1 second, whereas loading programs into EEPROM takes a few seconds (under 10 seconds for most). While loading programs into RAM can be a lot quicker for testing the results of changes during code development, programs should be loaded into EEPROM when the application is deployed, or if it is expected to restart after a power cycle or reset. Programs loaded into RAM are volatile, meaning they can be erased by a power interruption or by resetting the Propeller chip. In contrast, programs loaded into EEPROM are nonvolatile. After a power cycle or reset, the Propeller chip copies the program from EEPROM into RAM and then starts executing it again.

The Propeller Education Kit Hardware

The Propeller Education (PE) Kit is a complete Propeller microcontroller development system that can be used for projects and product prototypes. This kit also includes parts for projects that are documented by the PE Kit Labs. These labs will help you learn how to develop applications with the Propeller Microcontroller.



Figure 1-9: Propeller Education Kit (40-Pin DIP Version)

1: Propeller Microcontroller & Labs Overview

The PE Kit comes in two different versions: 40-pin DIP and PropStick USB. Both feature an arrangement of interlocking breadboards with the following parts mounted on them:

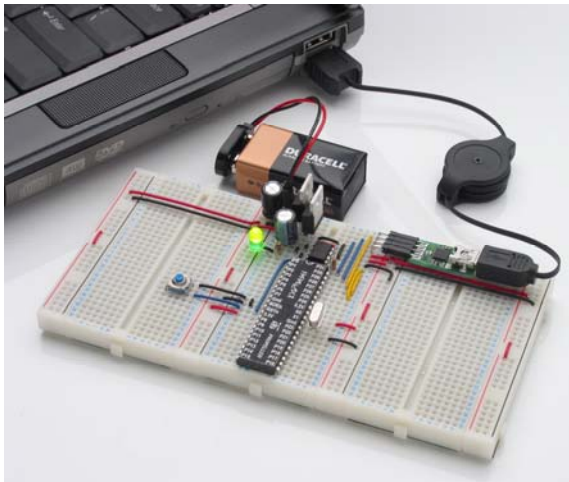
- Propeller microcontroller
- 5.0 V and 3.3 V voltage regulators
- EEPROM for non-volatile program storage
- 5.00 MHz external crystal oscillator for precise clock signal
- Reset button for manual program restarts
- LED power indicator
- 9 V battery-to-breadboard connector
- Serial to USB connection for downloads and bidirectional communication with the PC.

Collectively, the interlocking breadboards with Propeller microcontroller system mounted on it are referred to in this document as the PE Platform. The PE Platform with the 40-pin DIP kit is also shown in Figure 1-10 (a). With this platform, each part and circuit in the list above is plugged directly into the breadboard. Although this version of the PE Platform takes a little while to build and test, the advantage is that any given part can be replaced at a very low cost.

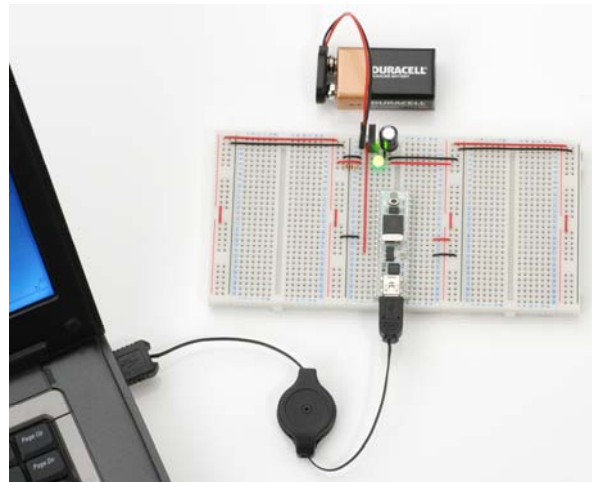
The PE Platform with the PropStick USB is shown in Figure 1-10 (a). The PropStick USB module is a small printed circuit board (PCB) with surface-mount versions of all the parts and circuits listed above (except the external 5 V regulator circuit). The PCB itself has pins so that it can be plugged into the breadboard. While this arrangement makes it quick wire up the PE Platform and get started, it can be relatively expensive to replace the PropStick USB rather than individual components if something gets damaged.

Figure 1-10: PE Kit Platforms

(a) 40-pin DIP Version



(b) PropStick USB Version



The Propeller Education Kit Labs

The Propeller Education Kit Labs include the ones printed in this text as well as additional labs and applications available for download from www.parallax.com/go/PEKit. The labs in this text demonstrate how to connect circuits to the Propeller microcontroller and write programs that make the Propeller chip interact with the circuits. The programs also utilize the Spin programming language's features as well as the Propeller microcontroller's multiprocessing capabilities.

Prerequisites

These labs assume prior microcontroller experience. Although the Setup and Testing labs provide wiring diagrams, the rest do not. At a minimum, you should have experience building circuits from schematics as well as experience with some form of computer or microcontroller programming language.



Resources for Beginners: For introductions to building circuits, microcontroller programming, and much more "prior microcontroller experience", try either the BASIC Stamp Activity Kit or BASIC Stamp Discovery kit. Both kits have everything you'll need to get started, including a BASIC Stamp 2 microcontroller, project board, the introductory level *What's a Microcontroller?* text and parts for every activity. The *What's a Microcontroller?* text is also available for free PDF download from www.parallax.com/go/WAM, and both kits are available for purchase from the web site as well as from a variety of electronics retailers and distributors. To find a retailer or distributor near you, check the Distributors list under the Company category at the Parallax web site.

PE Kit Labs in This Text

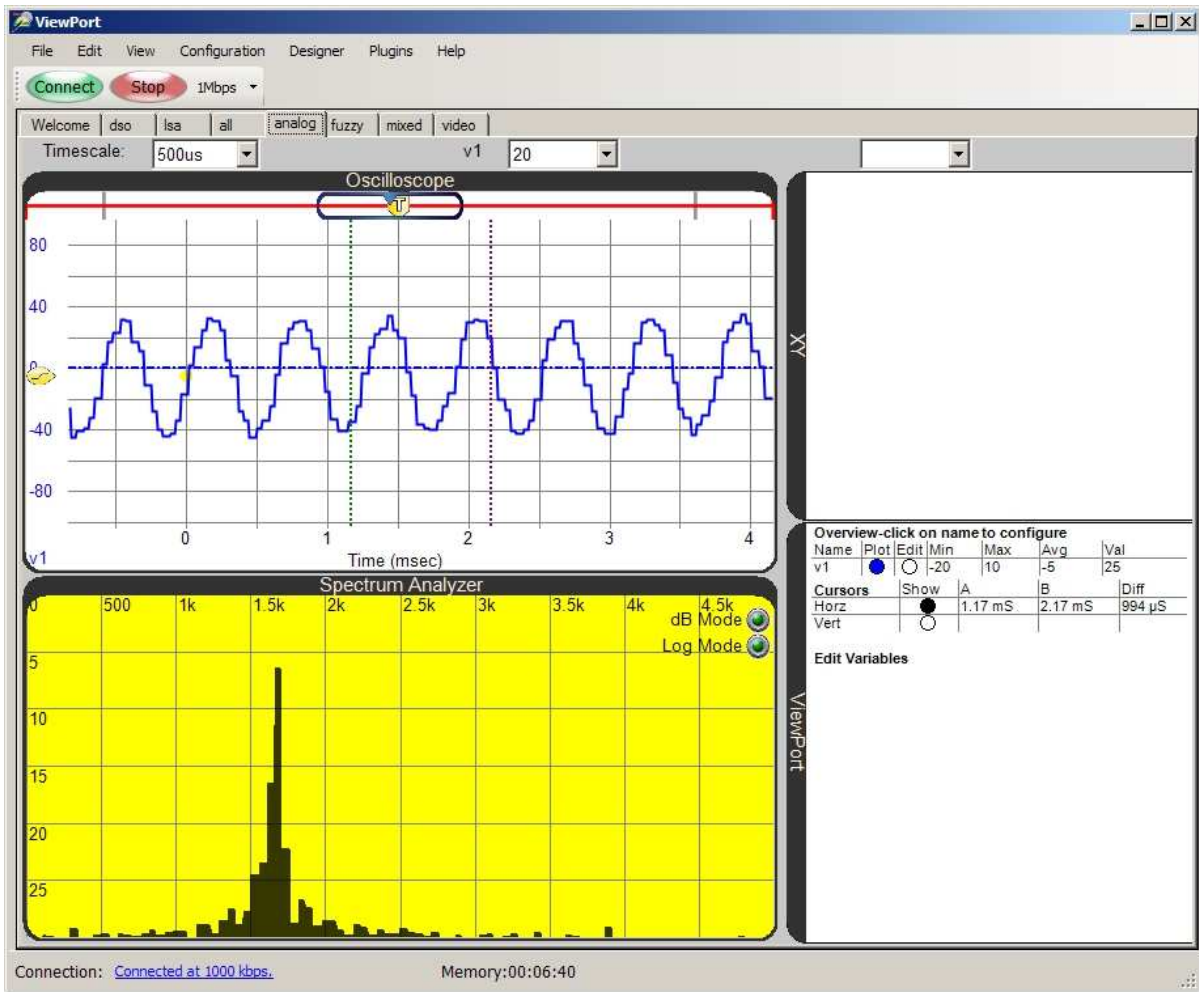
- **Software, Documentation & Resources** – Download Propeller software and documentation, and install the software.
- **Setup and Testing Lab for 40-Pin DIP PE Platform** – Hardware preparation. If you have the **PropStick USB Version** of the PE Kit, use its alternative Setup and Testing Lab. It is a free download from www.parallax.com/go/PEKit.
- **I/O and Timing Basics Lab** – How to configure the Propeller chip's I/O pins, monitor input signals, transmit output signals, and coordinate when events happen based on the system clock.
- **Methods and Cogs** – How to write methods in Spin and optionally launch methods into one or more of the Propeller chip's cogs (processors).
- **Objects** – How use pre-written objects to simplify coding tasks, and how to write objects.
- **Counter Modules and Circuit Applications** – How to employ the counter modules built into each cog to perform measurements and control processes that require precise timing. (Each cog has two counter modules that can function in parallel with the cog's program thread.)

The last four labs (I/O and Timing through Counter Modules and Circuit Applications) have questions, exercises, and projects at the end of the chapter with answers in Appendix B: Study Solutions, starting on page 201. For best results, hand-enter the code examples as you go through the labs. It'll give your mind time to consider each line of code along with the concepts and techniques introduced in the various sections of each lab.

More PE Kit Labs & Applications Online

To find additional labs and applications that build on the concepts introduced in this book, go to www.parallax.com/go/PEKit. You will find links to PDFs and discussions for each of the labs in this text along with additional material, like the ViewPort lab excerpt shown in Figure 1-11. Some of these labs will utilize the parts on the PE Kit, and others require additional parts, most of which are available from Parallax or other electronics suppliers.

Figure 1-11: ViewPort Lab Excerpt

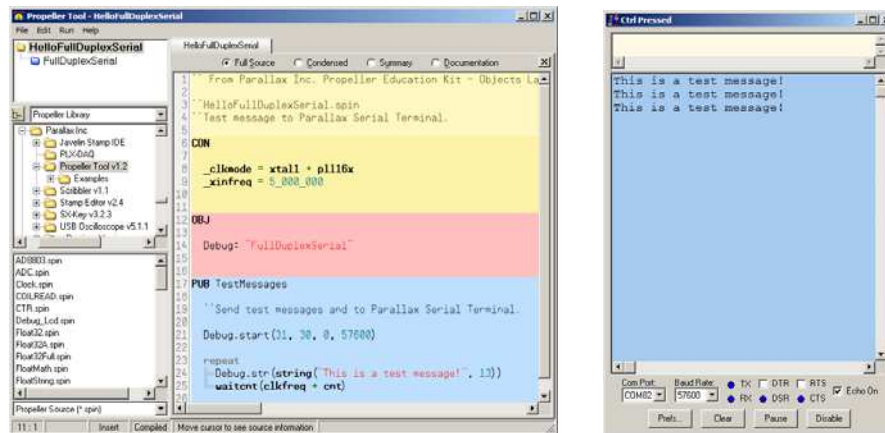


Oscilloscope and spectrum analyzer display signals generated by a microphone as someone whistles into it. The Propeller chip samples these signals and forwards them to the ViewPort PC Software. This is one of the activities featured in the ViewPort Lab.

2: Software, Documentation & Resources

The labs in this book require the free software shown in Figure 2-1. You'll use the Propeller Tool to write programs for the Propeller chip, and the Parallax Serial Terminal will provide bidirectional text communication between the PC and Propeller chip. Both are included in the Propeller Tool Software Installer (please use v1.3 or newer). The installer also includes PDF versions of this book, the Propeller Manual, and Propeller Datasheet, along with the example code.

Figure 2-1: Propeller Tool (left) and Parallax Serial Terminal (right)



Download Software and Documentation

- ✓ Go to www.parallax.com/Propeller → Downloads & Articles,
- ✓ Download the Propeller Tool Software v1.3 or newer (requires Win2K/XP/Vista/7)
- ✓ Install the Propeller Tool software by running the setup program and following the prompts. When you get to the Install Optional Driver Step shown below, make sure to leave the *Automatically install/update driver* box checked. (You can also check for the latest drivers at www.parallax.com/usbdriers.)



- ✓ If you are using the PropStick USB version of the PE Kit, be sure to locate its separate Setup and Testing Lab PDF file from www.parallax.com/go/PEKit.

Accessing the Software and Documentation

The installer should make program shortcuts to both the Propeller Tool and Parallax Serial Terminal. They are also accessible from the Start menu → All Programs → Parallax Inc → Propeller Tool. The same Start menu path will also lead you to a Reference folder with many documentation resources, as shown in Figure 2-2. These PDF files can also be opened through the Propeller Tool's Help menu.

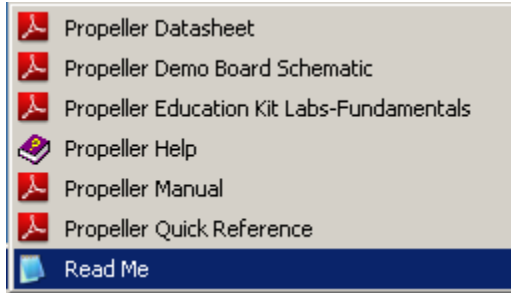


Figure 2-2: PDF Resources included with the Propeller Tool software

Accessing the Example Code

The Propeller Tool includes several sets of example code, available from the File → Open From menu as shown in Figure 2-3. The example code for this book is included.

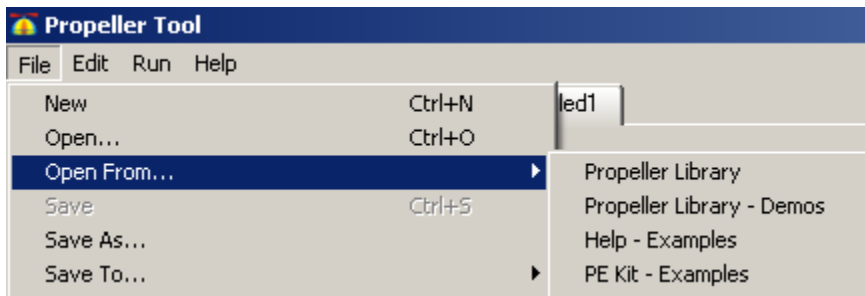


Figure 2-3: Example code sets in the File → Open From menu

Useful Web Sites

In addition to www.parallax.com/Propeller, there are a couple of other web sites where you can get answers to questions as well as objects to reduce your development time on Propeller projects.

- Additional labs and resources for your Propeller Education Kit: www.parallax.com/go/PEKit
- Object exchange: <http://obex.parallax.com>
- Propeller Chip forum: <http://forums.parallax.com> → Propeller

Tech Support Resources

Parallax Inc. offers several avenues for free technical support services:

- Email: support@parallax.com
- Fax: (916) 624-8003
- Telephone: Toll free in the U.S: (888) 99-STAMP; or (916) 624-8333. Please call between the hours of 7:00 am and 5:00 pm Pacific time, or leave us a message.
- Propeller Chip Forum: Propeller Chip forum: <http://forums.parallax.com> → Propeller. Here you will find an active, moderated forum dedicated to the Propeller chip, frequented by both Parallax customers and employees.

3: Setup and Testing Lab for 40-Pin DIP PE Platform

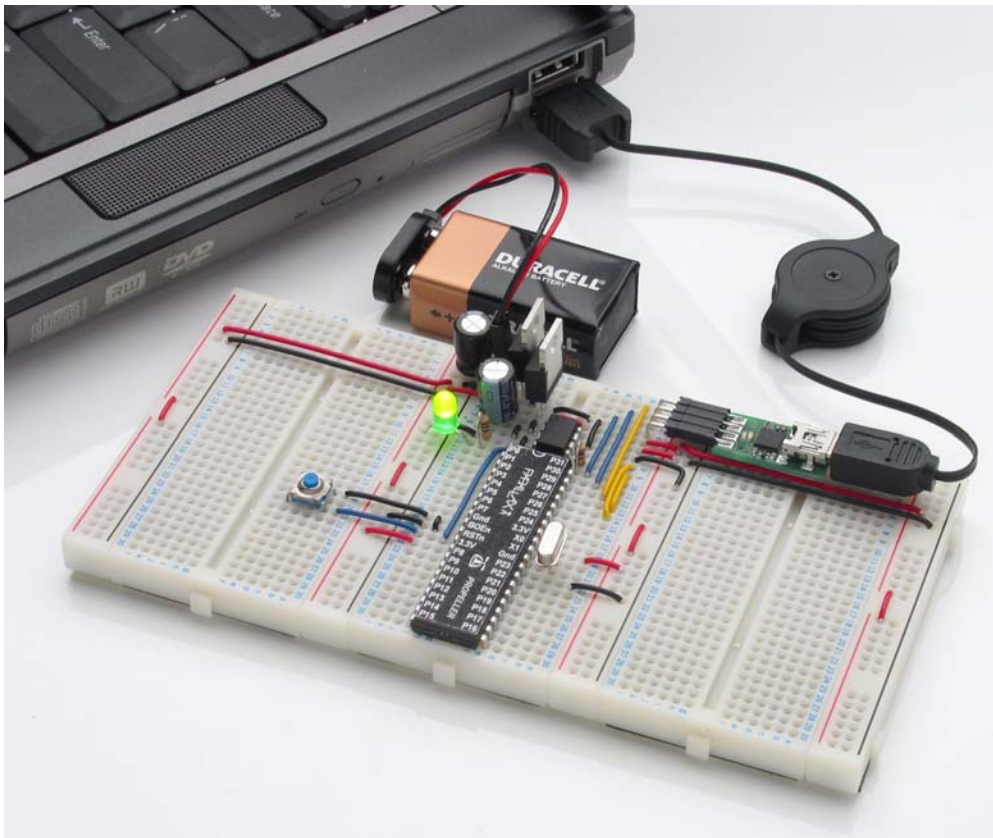
This is the Setup and Testing lab for the 40-pin DIP version of the PE Kit.

- ✓ If you have the 40-Pin DIP version of the PE Kit (#32305), continue here.
- ✓ If you have the PropStick USB version of the PE Kit (#32306), check for a separate printed Setup and Testing Lab - PropStick USB Version document included with your kit. It is also a free download from www.parallax.com/go/PEKit.

The PE Platform

The Propeller Education (PE) Kit Platform shown in Figure 3-1 makes a great reusable prototyping tool for electronics and robotics projects. It's also a great starting point for learning the basics of programming the Propeller microcontroller to be the embedded computer brain in your next invention. This lab introduces the 40-Pin DIP PE Platform and its components and features, and then guides you through assembling and testing your PE Platform.

Figure 3-1: PE Kit Platform (40-Pin DIP version)



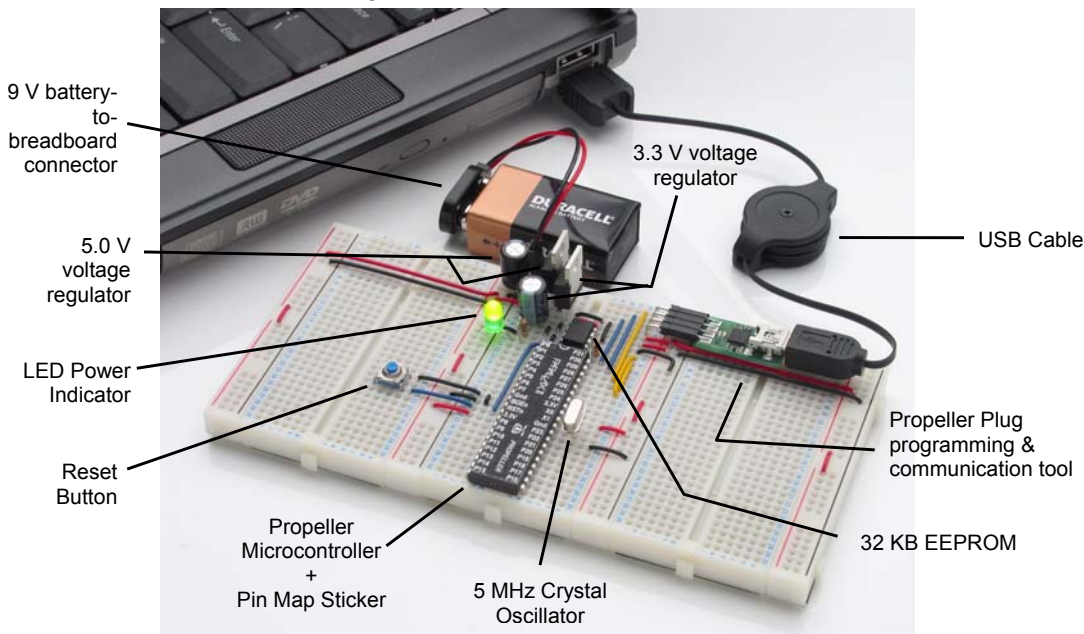
The PE Platform in Figure 3-1 is an array of breadboards connected side-by-side with the Propeller chip and support circuits mounted in the center. Each project circuit you build on the left or right breadboards will be adjacent to Propeller chip I/O pins for easy access. Each breadboard also has vertical power connectors on both sides so that ground and regulated 3.3 V are next to any given breadboard row. This arrangement makes most circuits very simple to wire and visually track. It also minimizes the wiring spaghetti and troubleshooting problems that can occur with tall individual breadboards.

PE Platform Components and Features

Figure 3-2 shows the 40-Pin DIP PE Platform's major components, including:

- Propeller microcontroller with pin map sticker affixed
- 9 V battery-to-breadboard connector
- 5.0 V and 3.3 V voltage regulators
- LED power indicator
- Reset button for manual program restarts
- 5.00 MHz external crystal oscillator for precise clock signal
- 32 KB EEPROM for non-volatile program storage
- Propeller Plug programming and communication tool for program downloads and bidirectional communication with the PC.

Figure 3-2: PE Kit Platform Components



Propeller Microcontroller

A Propeller Microcontroller in a 40-pin DIP package provides a breadboard friendly brain for the PE Platform. This amazing microcontroller has eight processors, called cogs. Its system clock can run at up to 80 MHz, and each cog can execute up to 20 million instructions per second (MIPS). Each cog takes turns at accessing the Propeller chip's main memory. This memory access combined with the Spin (high level) and Assembly (low level) languages created especially for the Propeller makes writing code for multiple processors very simple and straightforward. If you've ever written a BASIC subroutine and subroutine call (or a C function and function call, or a Java method and method call), making a different processor execute that subroutine/function/method takes just two more steps. You'll see lots of examples of this as you go through the PE Kit Labs.



Propeller Datasheet and Propeller Manual

The Propeller Datasheet provides a complete technical description of the Propeller microcontroller, and the Propeller Manual explains the chip's programming software and languages in detail. Both the Propeller Datasheet and Propeller Manual are included in PDF form in the Propeller Tool software. The printed version of the Propeller Manual is also available for purchase at www.parallax.com (#122-32000).

Reset Button

The reset button can be pressed and released to restart program execution. It can also be pressed and held to halt program execution. When released, the Propeller chip will load the program stored in PE Platform's EEPROM program and restart from the beginning.

9 V Battery-to-Breadboard Connector

This little gadget provides a simple, breadboard-friendly power supply connection. The recommended DC supply voltage across VIN–VSS is 6 to 9.5 VDC, and recommended power sources for VIN–VSS include:

- 9 V alkaline batteries
- Rechargeable 9 V batteries (common voltage ratings include 9 V, 8.4 V, and 7.2 V)



Always disconnect the battery from the connector and store separately. 9 V batteries should never be stored in the same box as the PE Kit platform or components because loose parts could short across the terminals. The 9 V battery should always be disconnected from the battery-to-breadboard connector and stored where its terminals cannot short across any metal objects or other conductive materials.

“Wall Warts”: The term “wall wart” commonly describes the DC supplies that draw power from AC wall outlets, and they often supply a much higher DC voltage than they are rated for. If you are going to use a wall wart, it's usually best to choose one that's rated for 6 V regulated DC output with a current capacity of 500 mA or more. The PE DIP Plus kit includes a 47 μF capacitor that can be placed across the battery inputs on the breadboard to provide the input capacitance required by the PE Platform's voltage regulator due to the wall wart's longer supply line.

5.0 V Regulator

The National Semiconductor LM2940CT-5.0 regulator is included in the PE Platform to make it convenient to supply 5 V components, such as the infrared detector introduced in the Counter Modules and Circuit Applications lab. A series resistor (typically 10 k Ω) should always be connected between a 5 V output and a Propeller I/O pin, which is 3.3 V. The 5 V regulator also serves as an intermediate stage between the battery input voltage and the 3.3 V regulator that supplies the Propeller chip.

The LM2940 voltage regulator circuit is designed to provide a 400 mA output current budget with a 9 V battery supply in the classroom or lab (at room temperature). This current budget can vary with supply voltage and temperature. For example, if the supply voltage reduced from 9 V to 7.5 V, the current budget increases to nearly 700 mA at room temperature. Another example, if the supply voltage is 9 V, but the ambient temperature is 100 °F (40 °C), the current budget drops to around 350 mA.



More Info:

- Appendix E: LM2940CT-5.0 Current Limit Calculations beginning on page 227 includes equations you can use to predict the PE Kit's 5 V regulator circuit's current budgets under various supply voltage and temperature conditions.
- The LM2940CT datasheet, available from www.national.com, has lots more information, including pointers for attaching a heatsink to the LM2940 to increase its current/temperature budget by improving its ability to dissipate heat

3.3 V Regulator

This National Semiconductor LM2937ET-3.3 regulator can draw up to 400 mA from the PE Platform's LM2940 (5 V regulator) at room temperature and supply the 3.3 V system with up to 360 mA of current. The 3.3 V system includes the Propeller chip, EEPROM, power LED, and the variety of 3.3 V circuits you will build in the PE Kit Labs.

Setup and Testing Lab

Keep in mind that if you have a power-hungry 5 V circuit, it subtracts current from the 5 V regulator's 400 mA output current budget, which in turn leaves the 3.3 V regulator with a smaller current budget to supply the rest of the system.

LED Power Indicator

This light turns on to indicate that power is connected to the board. It can also provide indications of dead batteries, short circuits, and even tell you if the Propeller Plug programming and communication tool is connected. As wired in this lab, it draws about 12 mA. After completing this lab, you can use a larger resistor for a less-bright indicator light that draws less current.

5.00 MHz Crystal Oscillator and Socket

The 5.00 MHz crystal oscillator provides the Propeller chip with a precise clock signal that can be used for time-sensitive applications such as serial communication, RC decay measurements and servo control. The Propeller chip has built-in phase locked loop circuitry that can use the 5.00 MHz oscillator signal to generate system clock frequencies of 5, 10, 40 or even 80 MHz.

The 5.00 MHz oscillator can also be replaced with a variety of other oscillators. A few examples include a programmable oscillator and a 60 MHz crystal. The Propeller chip also has a built-in RC oscillator that can be used in fast or slow modes (approximately 12 MHz and 20 kHz respectively). The internal oscillators are not nearly as precise as the 5.00 MHz oscillator, so if your project involves time-sensitive tasks such as serial communication, pulse width modulation for servo control, or TV signal generation, make sure to use the external 5.00 MHz oscillator.

32 KB EEPROM

The PE Platform's 32 KB EEPROM program and data storage memory is non-volatile, meaning it can't be erased by pressing and releasing the reset button or disconnecting and reconnecting power. This EEPROM memory should not be treated like RAM because each of its memory cells is only good for 1 million erase/write cycles. After that, the cell can actually wear out and no longer reliably store values. So, a program that modifies an EEPROM cell once every second would wear it out in only 11.6 days. On the other hand, if a cell gets modified every ten minutes, it'll be good for over 19 years.



EEPROM: Electrically Erasable Programmable Read-Only Memory

RAM: Random Access Memory.

Keep in mind that your application can use the Propeller chip's main memory (32 KB of which is RAM) for indefinite writes and rewrites at any frequency. It can then use the EEPROM to back up data that the application may need later, especially if that data has to live through disconnecting and reconnecting power. The EEPROM Datalogging Application (available at www.parallax.com → Propeller → Downloads & Articles) introduces an object that can be used to periodically back up values stored in RAM to EEPROM.

Propeller Plug Programming and Communication Tool

The Propeller Plug provides a serial-over-USB connection between the Propeller chip and PC for programming, communication, and debugging. This tool's blue LED indicates messages received from the PC, while the red one indicates messages transmitted to the PC. The FTDI chip labeled FT232 on the module converts USB signals from the PC to 3.3 V serial signals for the Propeller chip and vice versa.

On the PC side, a virtual COM port driver provided by FTDI is bundled with the Propeller Tool software you installed in the previous chapter. Aside from being necessary for the Propeller Tool software to load programs into the Propeller chip, the virtual COM port makes it convenient for the Propeller chip to communicate with serial software such as Parallax Serial Terminal.



More Virtual COM Port Info

After the FTDI virtual COM Port driver is installed by the Propeller Tool Installer, a Propeller Plug that gets connected to one of the PC's USB ports appears as a "USB Serial Port (COMXX)" in the Windows Device Manager's Ports (COM & LPT) list. The FTDI driver converts data placed in the COM port's serial transmit buffer to USB and sends it to the Propeller Plug's FT232 chip, and USB messages from the FT232 are converted to serial data and stored in the COM port's receive buffer. Serial communication software like the Propeller Tool and Parallax Serial Terminal use these COM port buffers to exchange information with peripheral serial devices.

Prerequisites

Please follow the directions in Software, Documentation & Resources, starting on page 17, before continuing here.

Procedure Overview

In this lab, you will assemble the PE Platform (40-Pin DIP version), following the steps listed below. It's important to follow the instructions for each step carefully, especially since you will be wiring up your own development platform (on the breadboard) instead of just plugging the Propeller microcontroller into a socket on a carrier PCB.

- Inventory Equipment and Parts
- Assemble the Breadboards
- Set up PE Platform Wiring and Voltage Regulators
- Test the PE Platform Wiring
- Socket the Propeller Chip and EEPROM
- Connect the Propeller Plug to the PC and PE Platform
- Connect Battery Power Supply
- Test Communication
- Load a Test Program and Test the I/O Pins
- Troubleshooting for the 40-pin DIP PE Platform Setup (if necessary)

Since the PE Platform will be the microcontroller system at the heart of the PE Kit Labs, all its electrical connections should be tested before proceeding to the next lab. By following all the steps in this lab, it will help rule out potential wiring errors, which can easily slip by unnoticed as you build the PE Platform circuits, and then cause unexpected problems in later labs.

Setup and Testing Lab

Inventory Equipment and Parts

Required:

- Computer with Microsoft Windows 2000, XP, or Vista and an available USB port
- 9 V alkaline battery (For this Setup and Testing lab, use a new 9 V, alkaline battery.)
- PE Kit's Breadboard Set (#700-32305), Propeller Plug (#32201), and Propeller DIP Plus Kit (130-32305) listed in the tables below

Optional, but useful:

- Small needle-nose pliers and wire cutter/stripper
- Multimeter (DC + AC Voltmeter and Ohmmeter)
- Digital storage oscilloscope, such as the Parallax USB Oscilloscope (#28014)
- Antistatic mat and bracelet

!	<p>ESD Precautions: Electrostatic discharge (ESD) can damage the integrated circuits (ICs) in this kit. If you have an antistatic bracelet and mat, use them. If you don't, the metal chassis of a PC plugged into a grounded outlet can also provide a safe and convenient way of losing static charge periodically before and while handling ICs. The part of the chassis that's typically exposed on a PC is the frame on the back. The monitor and peripheral ports are connected to it with metal screws. Touch that frame (not the ports) before opening the antistatic bags and then frequently while handling the parts.</p> <p>Here are some more tips for reducing the likelihood of a static zap to PE Kit parts: Avoid touching the metal pins on the ICs. Handle ICs by their black plastic cases. Also, if you know your work area conditions cause you to build up static charge and then zap nearby objects, find another work area that is less static prone. Likewise, if you know a particular sweater causes you to build up charge in a certain chair, don't wear that sweater while working with the PE Platform.</p>
---	--

- ✓ Gather the components listed in Table 3-1, Table 3-2, and Table 3-3.
- ✓ Open up the PE Project Parts bag and check its contents against the PE Project Parts list in Table C-2 in Appendix C: PE Kit Components Listing.

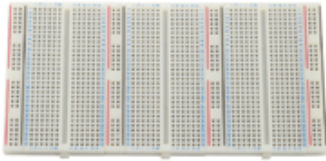
Table 3-1: Breadboard Set (#700-32305)			
700-00077	3	Breadboard, 12x30 sockets, 3.19" x 1.65"	
700-00081	4	Breadboard, 2x24 sockets, 3.19" x 0.5"	

















Table 3-2: Propeller Plug (#32201)			
	1	Propeller Plug	
805-00010	1	USB A to Mini B Retractable Cable	

Table 3-3: Propeller DIP Plus Kit (130-32305)		
Part Number	Quantity	Description
571-32305	1	9 V battery clip 
201-01085	2	Capacitor, Electrolytic, 6.3 V, 1000 μ F 
201-04740	1	Capacitor, Electrolytic, 25 V, 0.47 μ F 
150-01011	1	Resistor, CF, 5%, 1/4 watt, 100 Ω 
150-01030	1	Resistor, CF, 5%, 1/4 watt, 10 k Ω 
251-05000	1	Crystal 5.00 MHz, 20 pF, HC-49/ μ s 
350-00001	1	LED Green T1 $\frac{3}{4}$ 
400-00002	1	Pushbutton - normally open 
451-03601	1	2-pin m/m header 
451-00406	1	Extended right angle m/m 4 pin header with 0.1 spacing 
601-00513	1	3.3 V regulator, TO92 package 
601-00506	1	5.0 V regulator, TO92 package 
602-00032	1	32 kB EEPROM, DIP-8 
800-00016	6	Bags of 10 Jumper Wires 
P8X32A-D40 120-00003	1 1	Propeller Chip P8X32A - 40 pin DIP  Propeller DIP pin map sticker 

Parts and quantities subject to change without notice.

Assemble the Breadboards

The three 12-column \times 30-row prototyping breadboards in Figure 3-3 have sockets whose locations can be described by (column letter, row number). Each column has a letter along the top and bottom of the breadboard and each row has a number along the sides. Two examples of breadboard coordinates in the figure are (K, 3) on the center breadboard, and (C, 7) on the right breadboard. Each breadboard is organized in rows of six sockets; all the sockets in each row of six are connected by a metal bracket underneath. So, to connect two or more wires together, just plug them into the same row of six sockets.