


Understanding Signals

Student Guide

VERSION 1.0

PARALLAX 

WARRANTY

Parallax warrants its products against defects in materials and workmanship for a period of 90 days. If you discover a defect, Parallax will, at its option, repair, replace, or refund the purchase price. Simply call for a Return Merchandise Authorization (RMA) number, write the number on the outside of the box and send it back to Parallax. Please include your name, telephone number, shipping address, and a description of the problem. We 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 does not apply if the product has been altered or damaged.

COPYRIGHTS AND TRADEMARKS

This documentation is copyright 2003 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 uses is permitted, subject to details shown in the Preface.

BASIC Stamp is a registered trademark of Parallax, Inc. If you decided to use the name BASIC Stamp on your web page or in printed material, you must state that "BASIC Stamp is a registered trademark of Parallax, Inc." Other brand and product names are trademarks or registered trademarks of their respective holders.

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, and any costs or 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.

WEB SITE AND DISCUSSION LISTS

The Parallax web site (www.parallax.com) has many application downloads, products, customer applications and on-line ordering for the components used in this text. We also maintain several e-mail discussion lists for people interested in using Parallax products. These lists are accessible from www.parallax.com via the Support/Discussion Groups menu. These are the lists that we operate:

- [BASIC Stamps](#) – With over 2,500 subscribers, 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, this list has 500 subscribers who discuss the use of the Stamps in Class curriculum in their courses. The list provides an opportunity for students to ask educators questions, too.

- Parallax Educators – This focus group of 100 members consists exclusively of educators and those who contribute to the development of Stamps in Class curriculum. Parallax created this group to obtain feedback on our curriculum development and to provide a forum for educators to develop Teacher's Guides.
- Parallax Translators – Consisting of less than 10 people, the purpose of this list is to provide a conduit between Parallax and those who translate our documentation to languages other than English. Parallax provides editable Word documents to our translating partners and attempts to time the translations to coordinate with our publications.
- Toddler Robot – A customer created this discussion list to discuss applications and programming of the Parallax Toddler robot.
- SX Tech – Discussion of programming the SX microcontroller with Parallax assembly language tools, compilers (BASIC and C). Approximately 600 members.

Table of Contents

Preface	iii
Copyright and Reproduction	iv
Foreign Translations	iv
Special Contributors	iv
Chapter #1: Oscilloscope Basics	1
What is an Oscilloscope?	1
How Does an Oscilloscope Work?	2
Running the OPTAScope 81M for the First Time	5
Plot Area	6
Horizontal and Vertical Dials, Channel and Run/Stop Buttons	7
Plot Area Indicator.....	9
Display Screen	10
Files / Settings Tab	10
Trigger Tab.....	11
Cursors Tab	12
Measurements Tab	13
Activity #1: Viewing High and Low Signals	14
Activity #2: Using the Horizontal Dial and Edge Triggering	18
Summary	21
Exercises	21
Further Investigation	21
Chapter #2: Servo Pulse Square Waves	23
Pulse Width Modulation and Hobby Servos	23
Activity #1: Measuring Pulses for Servo Control.....	25
Activity #2: Measuring Time Varying Servo Pulse Widths	31
Summary	33
Exercises	33
Further Investigation	33
Chapter #3: Sine Waves	35
Sine Waves with the BASIC Stamp FREQOUT Command.....	35
Activity #1: Sine Wave Triggering.....	36
Activity #2: Sine Wave Frequency and Amplitude Measurement	41
Activity #3: Dual Sine Wave Measurement.....	43
Summary	46
Exercises	47
Further Investigation	48

Chapter 4: R/C Circuits and Variable Resistors	49
What are Capacitors?	49
Resistors and Capacitors in RC Networks	49
Activity #1: Verifying the Calculated Resistor/Capacitor Time Constant	52
Activity #2: Variable Resistors in an RC Network.....	57
Summary	63
Exercises	63
Further Investigation.....	63
Chapter 5: Synchronous Serial Communication	65
Activity #1: Capturing Synchronous Serial Communication	65
Summary	72
Exercises	72
Further Investigation.....	72
Chapter 6: Asynchronous Serial Communication	73
Activity #1: Displaying 8-bit Inverted Data	74
Activity #2: Displaying 8-bit True Data.....	79
Summary	80
Exercises	80
Further Investigation.....	80
Chapter 7: Pulse Width Modulation with Infrared	81
Activity #1: Infrared Signals for Object Detection	82
Activity #2: Decoding Infrared Remote Control Signals.....	91
Summary	97
Exercises	97
Further Investigation.....	97
Chapter 8: Amplifiers	99
Op-Amp Used as a Buffer	102
Op-Amp Used as a Voltage Amplifier	103
Activity #1: Sine Wave through an LM-358 Op-Amp	105
Activity #2: Inverting Amplifier with Adjustable DC Offset	112
Summary	116
Exercises	116
Further Investigation.....	116
Appendix A: System and Equipment Requirements	117
Appendix B: OPTAscope 81M Specifications	121
Index	123

Preface

This text demonstrates how to use the OPTAscope 81M as an oscilloscope by viewing common signals generated by sensors and the Parallax BASIC Stamp. Most of the circuits and examples used in this guide are drawn from other Parallax Stamps in Class educational texts: *What's a Microcontroller?*, *Basic Analog and Digital* and *Applied Sensors*.

Students completing this text should be able to accomplish the following:

- Configure an oscilloscope to trigger and capture a signal.
- Measure waveform frequency and amplitude for single and dual sine waves.
- View a repetitive signal as it is changed using BASIC Stamp code or by varying sensor inputs.
- Understand the differences between synchronous and asynchronous serial signals.
- Know two common uses of Operational Amplifiers (op-amps).
- Make viewable infrared communication from handheld remote controls.

Signals and waveforms are discussed throughout the Stamps in Class series, but truly understanding their form and speed requires capturing and viewing the signals on an oscilloscope. The knowledge gained will allow students to take the leap from using pre-written BASIC Stamp code to using a datasheet to develop their own synchronous serial code, or even to choosing resistive sensors most suitable for their own projects. Viewing the signals that power servos can lead students to better understanding and utilization of programming techniques used for robotics and control systems applications.

The OPTAscope 81M is a low-cost, USB-based oscilloscope made by Optimum Designs. Parallax generally has kept the hardware kits for the Stamps in Class texts under one hundred dollars (excluding robots), but the OPTAscope 81M provides a great opportunity to support the texts and complete students' electronic kits at a relatively low cost.

The OPTAscope is a small, affordable oscilloscope that can help students to create and work with advanced electronic systems. Even if students have used an oscilloscope once or twice before, these OPTAscope 81M exercises will provide a greater understanding of oscilloscopes as tools and of electronics in general. This in turn will enhance the

students' educational experiences while working with BASIC Stamp-generated signals in projects found throughout the Stamps in Class series of textbooks

COPYRIGHT AND REPRODUCTION

Parallax grants you under a conditional right the ability to download, duplicate, and distribute this text without our permission. The condition is that this text, or any portion thereof, should not be duplicated for commercial use resulting in expenses to the user beyond the marginal cost of printing. That is, nobody would profit from duplication of this text. Any educational institution wishing to produce duplicates for their students may do so without our permission. This text is also available in printed format from Parallax. Because we print the text in volume, the consumer price is often less than typical xerographic duplication charges.

FOREIGN TRANSLATIONS

Parallax educational texts may be translated to other languages with our permission (e-mail stampsinclass@parallax.com). If you plan on doing any translations please contact us so we can provide the correctly-formatted MS Word documents, images, etc. We also maintain a discussion group for Parallax translators, which you may join. Go to www.yahogroups.com and search for "Parallax Translators". This will ensure that you are kept current on our frequent text revisions.

SPECIAL CONTRIBUTORS

Doug Pientak of Optimum Designs in Forest Grove, Oregon wrote the initial draft of this curriculum. Optimum Designs provides custom electronic design and consulting services, and also manufactures PC-based test equipment such as the OPTAscope 81M. Doug Pientak gained six years of experience working with high speed digital storage oscilloscopes in the Research and Development labs at Intel Corporation.

Optimum Designs would like to provide special thanks to the entire Parallax Team, who has provided a great amount of support and ideas to Optimum Designs, Inc. In particular, Aristides Alvarez, Ken Gracey, Andy Lindsay and Stephanie Lindsay lent extensive aid in the formatting and technical editing of the final revision. The Parallax Team strives to design, manufacture, and sell the BASIC Stamp line of products with the customer truly in mind. They are a great group of people.

In addition, recognition and thanks must go to our customer Sid Weaver, whose volunteer beta-testing provided valuable insight that allowed us to enrich the contents of this text.

Chapter #1: Oscilloscope Basics

Using an oscilloscope makes it much easier to characterize and understand the signals coming from a sensor or microcontroller. *Understanding Signals* assumes you've never used an oscilloscope. For the OPTAscope 81M to be of value to you, we must first teach you how to use an oscilloscope.

WHAT IS AN OSCILLOSCOPE?

An oscilloscope is an electronic device that displays graphical representations of electrical signals or waveforms. These graphical representations have two components, time and voltage. Time is represented by the horizontal axis, and voltage is represented by the vertical axis. There are two main types of oscilloscopes: analog oscilloscopes, and digital storage oscilloscopes.



Analog Oscilloscope: traditionally an instrument that creates a waveform display by applying the input signal (conditioned and amplified) to the vertical axis of an electron beam moving across a cathode-ray tube (CRT) screen horizontally from left to right. The inside of the CRT is coated with phosphor to create a glowing trace wherever the beam hits.

Digital Oscilloscope: a type of oscilloscope that uses an analog-to-digital converter (ADC) to convert the measured voltage into digital information. There are three types: digital storage, digital phosphor, and digital sampling oscilloscopes. The Tektronix *XYZs of Oscilloscopes* article at www.tektronix.com contains a detailed discussion of oscilloscope features and example applications.

The OPTAscope 81M is a digital storage oscilloscope that will be the focus of *Understanding Signals*, shown in Figure 1-1. Digital storage oscilloscopes record the voltage for a period of time and then display it, allowing you to review the fluctuations of the signal during the period recorded with great detail.



Figure 1-1:
The OPTAscope 81M

This portable oscilloscope connects to your PC via a USB cable. It comes with 3 sets of 1x probes, a USB cable, and a software CD (not shown).

Figure 1-2 provides an example of 1 ms (millisecond) of time, read from left to right. To accomplish this, the oscilloscope samples the input signal, or waveform, at a very specific rate called the sample rate. The OPTAscope has a maximum sample rate of one million samples per second, (1Ms/s) when using one channel, half that when using both channels at once. The sample rate is the limiting factor that determines the maximum signal frequency that the OPTAscope can capture and display. This means that, with the OPTAscope, you can view any sine wave with a frequency of 60 kHz or less. Square waves can be viewed if they are 100 kHz or less. In either case, using the Zoom feature produces nice waveform images even at these extreme speeds.

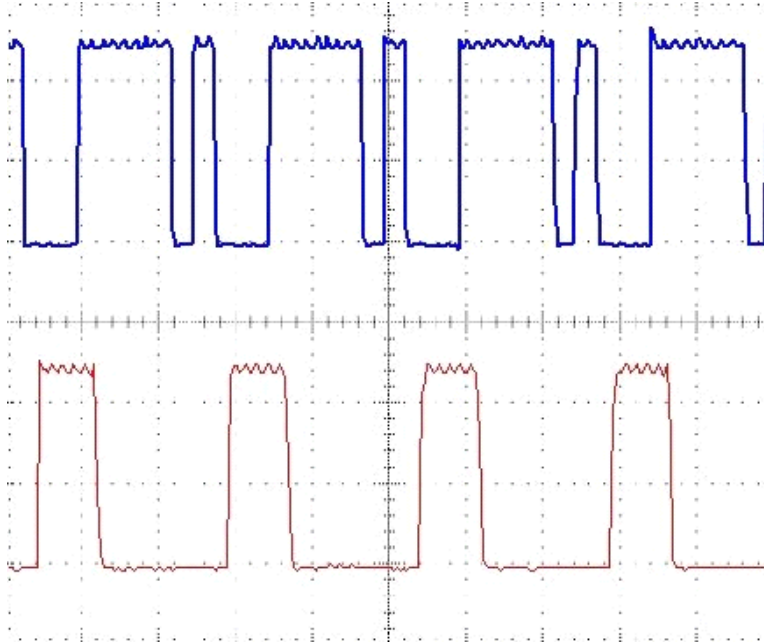


Figure 1-2:
OPTAscope 1Ms/s
example.

*Here is an example
of the output of the
OPTAscope. It is
displaying the
characteristics of
two signals during
the same period of
1 ms.*

*This dual signal
display ability
allows you to see
what the signals
look like and their
relationship to each
other - very
important in some
applications.*

HOW DOES AN OSCILLOSCOPE WORK?

We have said that digital oscilloscopes sample and record signals, and then display these signals for viewing. Functionally, this is very similar to a simple datalogger. The difference between the two is that dataloggers record data all of the time, but oscilloscopes may be *triggered* to store data at particular times.

The trigger controls *when* the oscilloscope will start recording the input signal. The trigger allows you to capture and view only the segment in which you are interested. The criteria controlling the trigger action is called a trigger event. The OPTAscope 81M has edge-triggered events. More advanced oscilloscopes have several types of configurable trigger events: pulse width events, pulse sequence events, and even more exotic event triggers that can trigger on the start of a video packet in a composite TV signal. This text will focus on the OPTAscope's edge-trigger event ability.

There are two types of trigger events supported by the OPTAscope: rising edge, and falling edge. A rising edge is described as the point at which a relatively low voltage signal ascends to a relatively higher voltage. The converse is a falling edge. A rising edge trigger event will occur when the voltage of the signal rises above the set trigger point, and a falling edge trigger event occurs when the voltage falls below the trigger point. Upon a valid trigger event, the OPTAscope will begin recording the sampled signal input and continue until its memory is full.

When the OPTAscope displays the captured signal, the trigger event will be centered in the display for you to see. A rising edge-triggered signal is depicted in Figure 1-3.

Anything to the left of the trigger event is called “pre-trigger” because it happens before the trigger event. Anything to the right of the trigger event is called “post trigger”. This is important because when attempting to trigger the oscilloscope to capture a specific event, you need to set your trigger level to get as close as possible to the event you want to see.

In Figure 1-3, only one signal is displayed on the screen. With OPTAscope 81M you can have two signals displayed on the screen at the same time. Both signals are sampled or recorded concurrently, which allows you to view both signals (called channels) during the same period of time.

During the exercises in this text, you will use the OPTAscope to trigger, capture, and display signals. In doing so, you will learn the function and use of oscilloscopes, and gain a greater understanding of the different signal types you will encounter. Ultimately, you will also see how these skills can be useful working with different electronics applications.

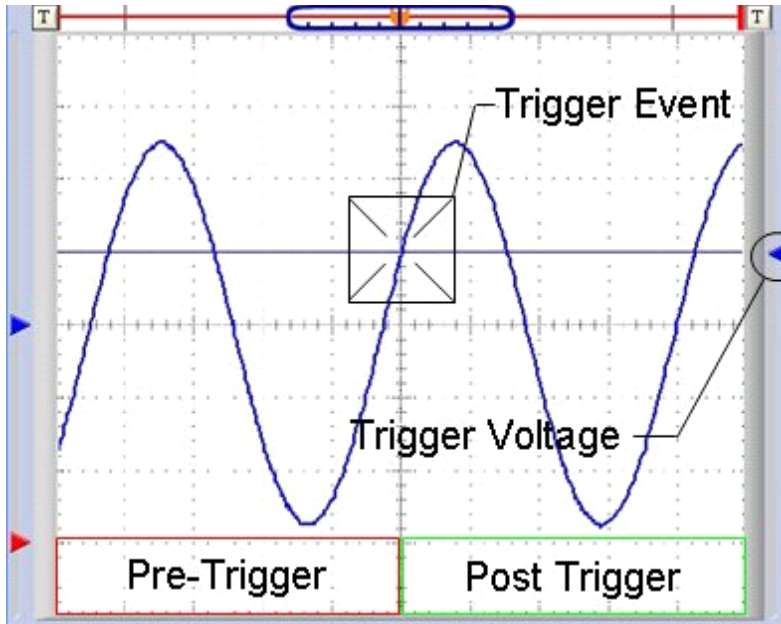


Figure 1-3:
Example of a
rising trigger
event, and the
resulting
waveform

OSCILLOSCOPE SAFETY

Before test-driving your OPTAscope 81M, it is important for you to be aware of the general safety guidelines for working with oscilloscopes. Working with oscilloscopes requires you to work on circuits with live voltage. Live voltage can and does KILL people every day. It is your responsibility to learn the following safety guidelines and practice them faithfully while working with live circuits. Failing to do so can result in equipment damage, and severe personal injury or death. Below you will find a list of safety rules that you are to use as a guideline while working with live circuits and the OPTAscope.

- Remove metallic jewelry and watches before starting.
- Make sure your hands are clean and dry while working with the OPTAscope.
- Work upon an anti-stat pad that is properly grounded.
- Keep your work area free from food and beverages.
- Do not attempt to measure any voltage that could be 20 Vpp or higher.
- If anything you are working on gets hot, turn it off immediately.
- ALWAYS disconnect the power supply to the circuit you are measuring before walking away from your workstation.

RUNNING THE OPTASCOPE 81M FOR THE FIRST TIME

Setting up the OPTAscope is a matter of installing the software from the CD provided, and connecting the OPTAscope to your PC via a standard USB cable (included). You may also obtain the latest version of the software from www.parallax.com. If the OPTAscope is not immediately recognized after driver installation see the “Hardware Installation” section of the on-line help file from your Start → Programs → Optimum Designs folder.

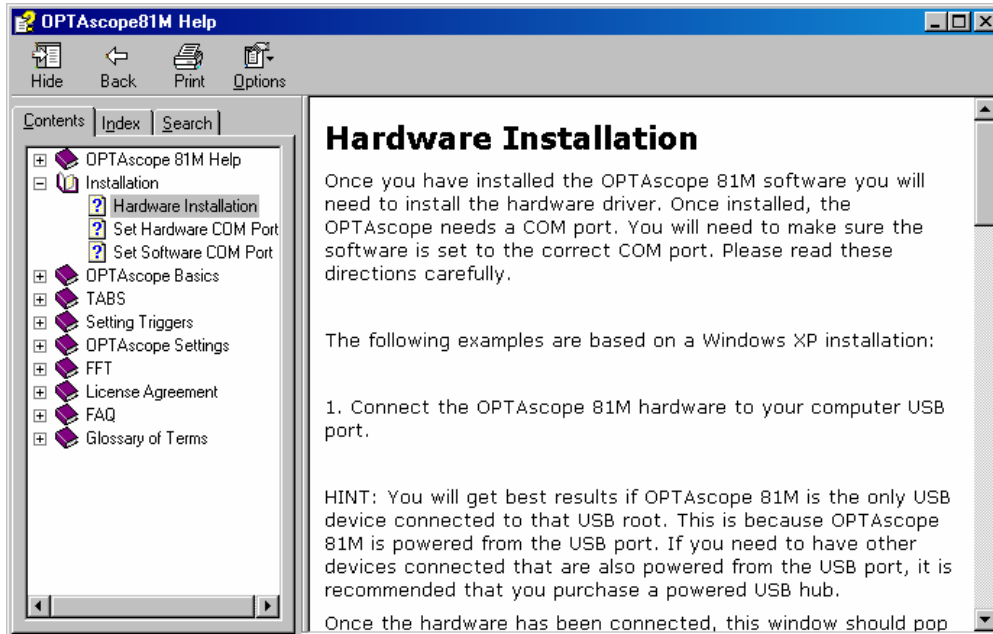



Important note to lab instructors and system administrators: Each OPTAscope has a unique USB-ID. If the computers in the lab are configured so that the students are not permitted to install new hardware, make sure that each OPTAscope has a label indicating which PC it was installed to. Students should then be instructed to use the PC indicated on the OPTAscope's label (or check out the OPTAscope with a label that corresponds with one on the PC they are using).

The reason unique IDs are used is because it gives you the option of running more than one OPTAscope on a single PC. This can be accomplished because the OPTAscope software assigns a unique COM port to each OPTAscope. This means you can open more than one instance of the OPTAscope software and use each to monitor signals using a different OPTAscope.

This section will guide you through the OPTAscope's basic hardware and software settings. This will give you an overview of how the OPTAscope works, and point out the various features you will be using in the upcoming exercises.

Figure 1-4: The OPTAscope hardware setup detailed in the on-line help file



 **Don't be afraid** if you change the settings to an unknown state. You can always reload the factory default settings. To do this, select File → Load Factory Default Settings from the pull down menu. Follow the directions and this will bring you back to a default installation state. Resetting to defaults will erase your calibration data and can reset your COM port setting if you answer yes to both questions.

Plot Area

The Plot Area is where the signals are displayed after the oscilloscope samples and records them. Notice the graph made up of 10 divisions horizontally and 8 divisions vertically. These divisions can be used to measure the signal's voltage (vertical divisions) and time duration (horizontal divisions). With the OPTAscope 81M you can have one or two signals in the Plot Area (Figure 1-5).

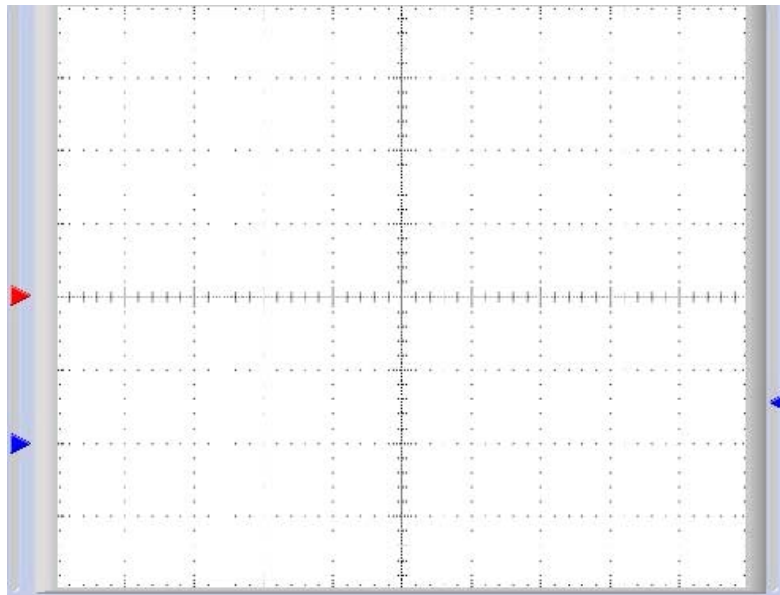


Figure 1-5:
The Plot Area

Note the horizontal and vertical divisions grid.

The red and blue arrows to the left are handles that allow you to adjust the vertical position of the signals. This will allow you to arrange the signals within the Plot Area as you wish, perhaps separating them for clarity or overlaying them for comparison.

The blue arrow to the right adjusts the trigger voltage. When you move this arrow a line will be displayed on the Plot Area representing the trigger voltage, then it will disappear three seconds after you stop moving the arrow. The trigger voltage arrow will change color depending on which channel you have selected as the trigger source. This arrow will not show if you set the Trigger Source switch to External.

Horizontal Dial, Vertical Dial (Volts/Div), Channel Buttons and Run/Stop Button

Recall that the OPTAscope can display graphs of two signals, each with two components: voltage and time. The Horizontal dial sets the time base of the oscilloscope, by choosing the amount of time represented by each division in the Plot Area. See Figure 1-6.

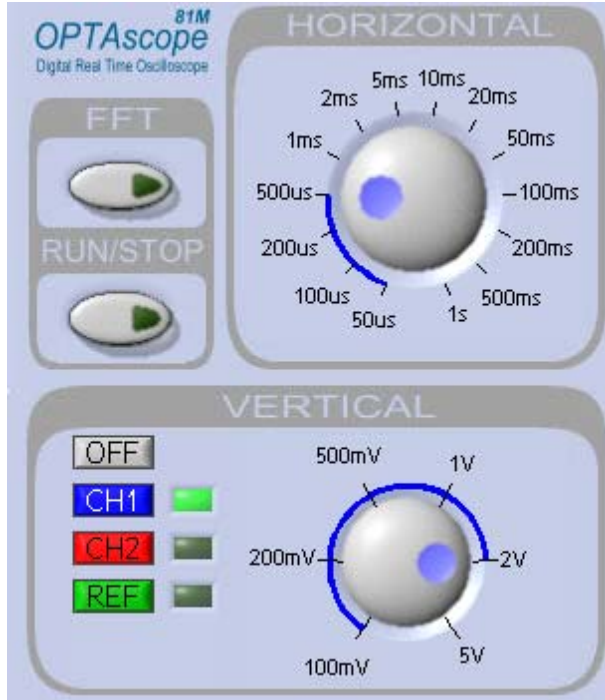


Figure 1-6:
Horizontal, Vertical, FFT
and Run/Stop controls

The Vertical dial, representing volts per division, sets the voltage scale on which the input signal will be displayed. Example: if the Vertical dial is set to 2 V (2 V/Div) and the signal displayed is 2½ divisions above the channel indicator, then the voltage of that input channel is equal to 5 Volts. It is possible to have different scales set for each signal, so that while Channel 1 is set to 2 V per division you can have Channel 2 set to 5 V per division.

The channel buttons, CH1 and CH2, select the active channel. The active channel is the last channel button clicked (with the green LED), and is also the selected channel for the Vertical dial, cursors, and automatic measurements in the display screen. The OFF button will turn off the active channel.

The Run/Stop button starts and stops the OPTAscope 81M. When the button is pressed, the OPTAscope 81M acquires data as indicated by the green LED in the button. To stop

the oscilloscope, press the Run/Stop button again. You will see it depress, indicating the oscilloscope is idle.

The FFT button opens the Fast Fourier Transformation window (Figure 1-7). The OPTAscope's Fast Fourier Transformation (FFT) function emulates a device called a spectrum analyzer by displaying the sine wave frequencies contained by a signal.

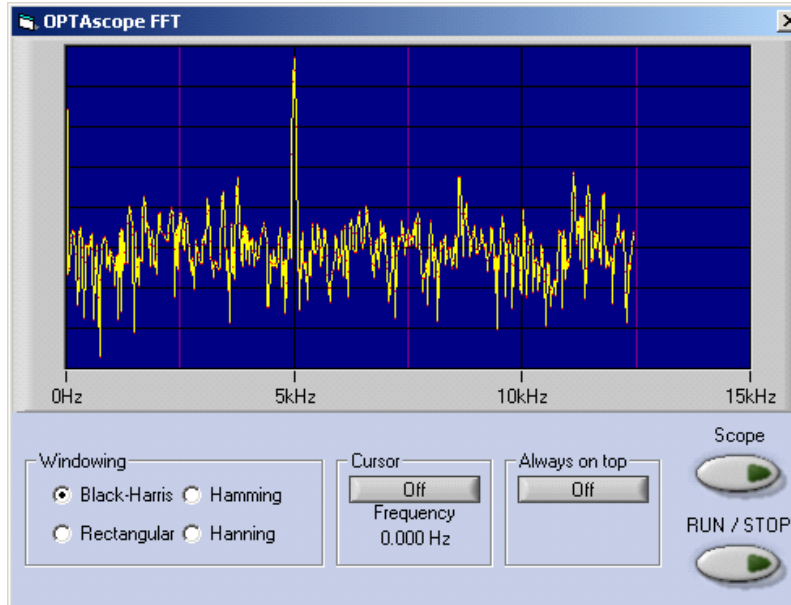


Figure 1-7:
The
OPTAscope's
FFT Window.

Plot Area Indicator

The Plot Area Indicator bar depicts the portion of the signal visible in the Plot Area. The OPTAscope 81M acquires 1,500 points every trigger event. The Plot Area automatically displays the center 500 points, represented by the blue slider bar. The red line represents the entire 1500 points captured. You may view these portions of the signal to the left or right of center by sliding the blue bar along the red line.

The orange arrow within the Plot Area Indicator bar indicates the trigger position and its relative location within the 1,500 points of data acquired. The “T” buttons on each end of the Plot Area Indicator bar move the trigger position to 10%, 50% (default) and 90%. Figure

1-8 shows the trigger position set to 50%. When the vertical line in the blue bar is lined up with the “T” in the trigger position arrow, the trigger event will line up in the center of the Plot Area.



Figure 1-8:
Plot Area
Indicator

Display Screen

The Display Screen (Figure 1-9) tabulates information regarding the OPTAscope’s settings and the measurements of the signals captured. The Channel Settings box displays the current volts per division setting for each channel. The Sample Rate box displays the number of samples taken each second. The Trigger Settings box displays the trigger channel selected and the trigger voltage level. The Automatic Measurements box displays the results of the automatic measurements taken. The Cursors box displays the statistics of the cursors positioned by the users. The Δ (delta) represents the difference in time between the two cursors. The f represents the frequency ($1/\Delta$) depicted by the relative cursor position. The cursor readouts are in reference to the active channel.

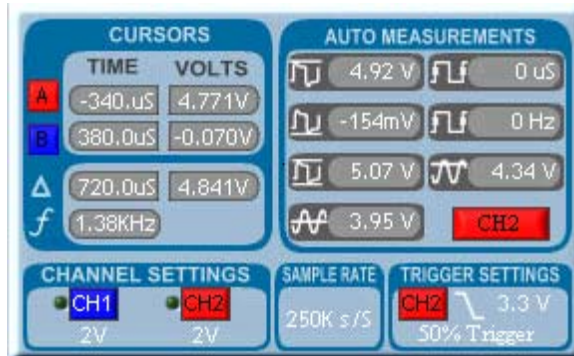


Figure 1-9:
Display screen

Files / Settings Tab

The Files/Settings tab gives you direct access to the Export Picture, Export Data, Print, Print Preview and OPTAscope Settings buttons, as shown in Figure 1-10 . For more detail on these features, review the OPTAscope help file. By clicking on the OPTAscope Settings button, a menu will appear giving you access to the Calibration, Hardware Setup, and Save/Load options.

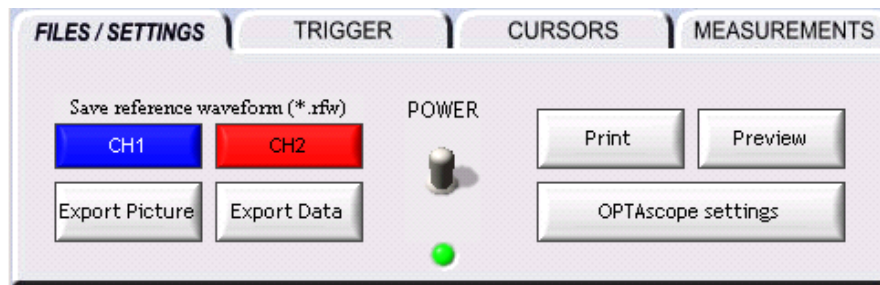


Figure 1-10:
Files /
Settings
Tab

Trigger Tab

Clicking the Trigger tab brings up a menu which allows you to set the Trigger Source, Trigger Edge, Trigger Mode, and Run/Stop Mode switches (Figure 1-11). The Trigger Source switch selects the channel monitored by the OPTAscope for a trigger event. (The external trigger is not displayed in the Plot Area and, when selected, is limited to rising-edge trigger events only.) The Trigger Edge switch configures the OPTAscope to wait for either a rising edge or a falling edge to use as a trigger event.

The Trigger Mode switch selects how long the OPTAscope will wait for a trigger event. In Normal mode the oscilloscope will wait for a trigger event until it finds one. This could take a fraction of a second, a minute, or even a week. If the trigger event never occurs the oscilloscope will never trigger, and you will never see a waveform in the Plot Area.

In Auto mode the oscilloscope will wait only a small portion of time for a trigger event. If the trigger event is not detected, the oscilloscope will trigger itself automatically. Whatever signal is being received at the time will be captured and displayed in the Plot Area.

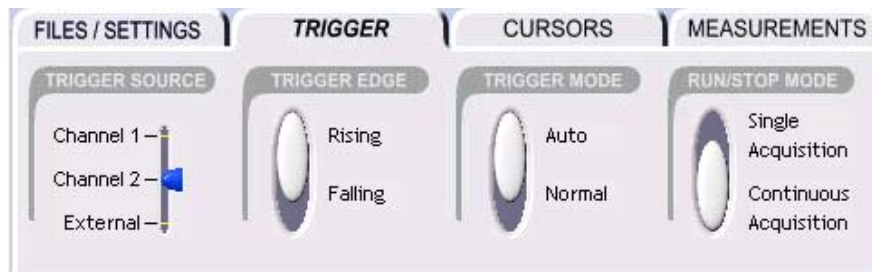


Figure 1-11:
Trigger
Tab



Auto mode is great as a first step to make sure you have the oscilloscope setup correctly. However, if the trigger event occurs infrequently, you may never see the signal you are looking for while in Auto mode. In that case, Normal mode is the better choice.

Cursors Tab

The Cursors tab shown in Figure 1-12 allows you to select the type of cursors you wish to use. The Cursor Settings switches give you several options. Snap to Plot will make the cursors snap to the closest data point on the active channel. Floating cursors are free to be placed anywhere within the Plot Area by you, the user. The other Cursor Settings switch gives you three cursor options, Horizontal Bars, Vertical Bars and Paired Bars, all of which will be used in the upcoming experiments.

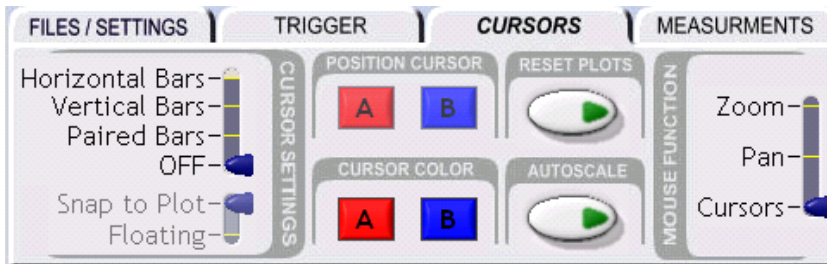


Figure 1-12:
Cursors Tab

The Position Cursor buttons will select and place the cursor(s) anywhere on the screen by clicking the associated button and then clicking in the Plot Area. Use Cursor Color to change the color of each cursor.

The Autoscale button will allow you to view all 1,500 data points in the Plot Area at once, where normally you would have to slide the Plot Area Indicator bar to see the 500 points to the left and right. Additionally, Autoscale will automatically adjust the volts per division setting such that the signal is displayed advantageously in the Plot Area.

The Mouse Function switch allows the mouse to operate in three different modes in the Plot Area. Zoom mode allows you to drag and draw a box around an area, then zoom in on that area to view detail. Pan mode lets you pan around the Plot Area while the cursors stay in place, while Cursors mode lets you move the cursors around.

The Reset Plots button will reset all plots to their default values. This button allows you to zoom out after you have zoomed in, and will also reset the Autoscale.



Cursor menu functions can also be accessed by right-clicking on the Plot Area.

Measurements Tab

The Measurements tab displays the automatic measurements for both channels. Each time a new screen shot of data is displayed the measurements are recalculated. Here is a quick description of each measurement:

MAX is the maximum the signal reached in the acquisition.

MIN is the minimum the signal reached in the acquisition.

Pk - Pk is the peak to peak voltage of the signal, calculated from MAX and MIN.

MEAN is the mean average voltage of the signal.

PERIOD is the time measured between two rising edges.

FREQ is the frequency of the signal (1/PERIOD).

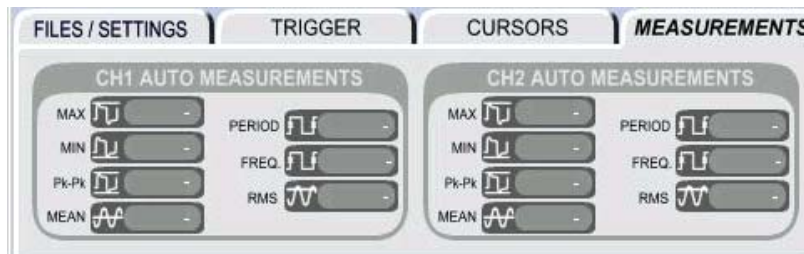


Figure 1-13:
Measurements
Tab

ACTIVITIES PREPARATION

You should already have your OPTAscope 81M connected to your PC and its software up and running. You may also check for the latest OPTAscope software at www.parallax.com

You will also need to have your Board of Education with a BASIC Stamp 2, or your HomeWork Board, connected to your PC. You will need the BASIC Stamp Editor v 2.0, available on a May 2003 or newer Parallax CD, or as a free download from www.parallax.com.

All activities in this text assume that you are using a fresh 9 V battery as your power supply. Please be aware that using worn-out batteries or alternative power supplies may alter your voltage signals somewhat from those shown in the book.

For detailed instructions on setting up your BASIC Stamp hardware and software, see *What's a Microcontroller?* under the Further Investigation section at the end of this chapter.

The Board of Education, BASIC Stamp and HomeWork Board are sold separately. For a complete list of system and equipment requirements, see Appendix A.

ACTIVITY #1: VIEWING HIGH AND LOW SIGNALS

In this activity we will verify the proper operation of your OPTAscope by viewing simple high and low signals from the BASIC Stamp.

Parts Required

- (1) OPTAscope CH1 probe and ground cable
- (2) Jumper wires



Which probe is which? The OPTAscope probes are color-coded to correspond with controls and signals graphics in the software. Connect the blue-tipped probe cable to the CH1 jack on the OPTAscope, and the red one to the CH2 jack. The green-tipped cable plugs into the External Trigger TTL jack. In the wiring diagrams, the probes are labeled for clarity. The active channel probes are colored (CH1 is blue, CH2 is red); the Ground probes are black.

Building the High and Low Signals Circuit

This circuit is built the same way for the Board of Education and the BASIC Stamp HomeWork Board.

- √ Plug the probe cable into the CH1 jack on the OPTAscope.
- √ Build the simple circuit shown in Figure 1-14 and Figure 1-15.
- √ Connect the OPTAscope CH1 probe to I/O Pin 14 of your BASIC Stamp.
- √ Connect the black probe to the BASIC Stamp's Vss connection (ground).



Figure 1-14:
High and Low signals
circuit schematic

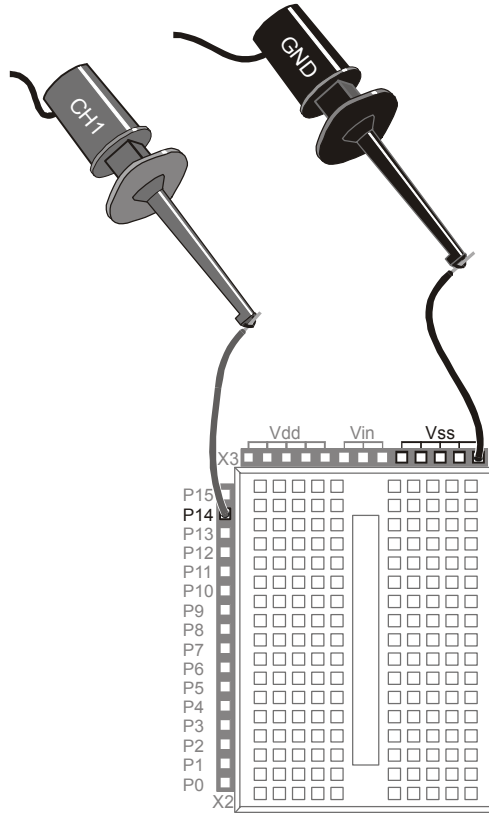


Figure 1-15:
High and Low signals
wiring diagram

Configuring the OPTAscope Software

- √ Open the OPTAscope software.

- √ Set up the OPTAscope to display the signal as shown in Figure 1-16. Remember, click on and drag the arrow to the left of the Plot Area to set the trigger voltage.

CH1	5 V / division
CH2	Off
Horizontal Dial	5 ms / division
Trigger Source	None
Trigger Edge	Falling
Trigger Mode	Auto
Run / Stop Mode	Continuous
Trigger Voltage	2 V

Figure 1-16:
Configuring the OPTAscope to view high and low signals

Measuring 5 V with PinHigh.bs2 Program

- √ Run the program PinHigh.bs2.

```
' Understanding Signals - PinHigh.bs2
' Make I/O pin high to demonstrate 5V measurement

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

DO

    HIGH 14

LOOP
```

In the OPTAscope Plot Area, you should see a line across the screen one division up from the blue arrow on the left (Figure 1-17). Your Vertical dial should be set to 5 V per division. A BASIC Stamp output pin should read about 5 V when it is “driving high”.

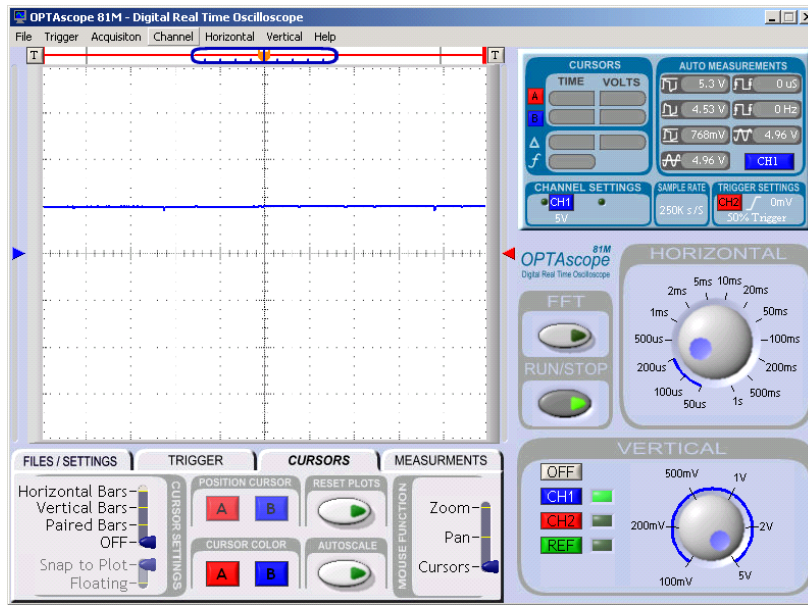


Figure 1-17:
High signal
example from a
BASIC Stamp

√ Change the **HIGH 14** command to **LOW 14**. What do you see now?

If you could watch the OPTAscope Plot Area while you programmed the BASIC Stamp, you would see the signal change from high (5 V) to low (0 V). Now you can see that the oscilloscope displays a low (0 V) as a line starting right at the blue arrow to the left. Since the BASIC Stamp is not changing the voltage level on this I/O pin during the program, you see a flat line. The next activity will generate a voltage that changes over time to make the plotted signal more interesting.

ACTIVITY #2: USING THE HORIZONTAL DIAL AND EDGE TRIGGERING

This activity will demonstrate how to use the OPTAscope's Horizontal dial to display different periods of a signal, and how to trigger on a rising or falling edge of a signal. The parts required and electronic circuit is the same as in the prior Activity.

Configuring the OPTAscope Software

- √ Set up the OPTAscope to display the signal as shown in Figure 1-18 .

CH1	2 V / division
CH2	Off
Horizontal Dial	5 ms / division
Trigger Source	Channel 1
Trigger Edge	Rising
Trigger Mode	Auto
Run / Stop Mode	Continuous
Trigger Voltage	2 V

Figure 1-18:
Configuration for OPTAscope to trigger a signal from ToggleIO.bs2 program

Toggling High and Low with the ToggleIO.bs2 Program

- √ Program the BASIC Stamp with the program ToggleIO.bs2.

```
' Understanding Signals - TOGGLEIO.bs2
' Toggle P14 to demonstrate time period measurements
' {$STAMP BS2}
' {$PBASIC 2.5}

DO

    TOGGLE 14
    PAUSE 10

LOOP
```

You should now see a square wave in the Plot Area, as shown in Figure 1-19.

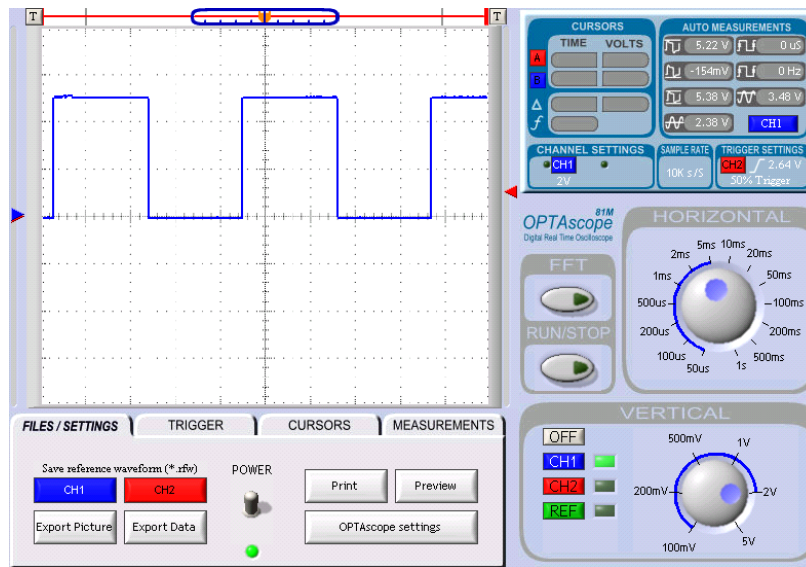


Figure 1-19: Signal generated by toggling an I/O pin

The BASIC Stamp is toggling an I/O pin, that is, switching between **HIGH** and **LOW**. Using the cursors, we can measure how long the signal is high. It will show that the high signal lasts for 10 ms, just as you programmed the BASIC Stamp to do.

- ✓ Under the Cursors tab, set the Cursor Settings switch to Vertical Bars.
- ✓ Drag the bars to either side of a high pulse.
- ✓ In the Auto Measurements box, read the delta (Δ) value in the Time column. It should be around 10 ms.
- ✓ Under the Trigger tab, set the Trigger Edge switch to Falling. Notice the signal shifts to the left. Now the oscilloscope is triggering on the falling edge of the input signal.
- ✓ Move the trigger voltage above 5 V, out of the range of the signal. Notice the rolling effect. The OPTAscope is looking for a trigger event but does not find one because you have the Trigger Mode switch set to Auto. This causes the OPTAscope to automatically trigger even though the input signal never causes a valid trigger event.

- √ Set the Trigger Mode switch back to Normal. Notice the Plot Area never updates, because the OPTAscope will now only trigger with valid trigger events. Since the input signal cannot trigger a valid event, there is nothing new to display in the Plot Area.
- √ Move the trigger voltage back to 2 V, which takes it back into the signal's range. You will now see valid trigger events updating the Plot Area.
- √ Experiment a bit: under the Cursors tab, set the Mouse Function switch to Zoom mode. Try zooming in on one pulse. Click the Reset Plots button to zoom back out. Change the Horizontal and Vertical dials, and observe the effects in the Plot Area.

Summary

An oscilloscope is a device that allows one to view graphic representations of electrical signals. Oscilloscopes are delicate pieces of electronic test equipment that must be used safely and cared for properly. Simple oscilloscopes, such as the OPTAscope, offer two modes of trigger operation with which to capture signals: Normal and Auto. Normal mode waits indefinitely for a trigger event before capturing and displaying the waveforms. Auto mode will wait a short while for a trigger event. If no trigger event occurs within a short period of time, it will automatically start the capture and display sequence anyway. Trigger events for simple oscilloscopes are of two types: rising edge and falling edge. Once a waveform is displayed, it can be measured with either automatic cursors or with user-positioned cursors.

Exercises

1. Describe the function of the Horizontal dial.
2. Describe the function of the Vertical dial.
3. Describe the difference between the OPTAscope's Normal and Auto trigger settings modes.
4. What measurement does the Δ symbol refer to?
5. What is the maximum voltage that you can safely measure on the OPTAscope?
6. Decrease and increase the **PAUSE** command in the BASIC Stamp program ToggleIO.bs2, to see what the practical limitations are with the OPTAscope. What is the smallest value that still allows you to see the signal? What about the largest value?
7. Connect the second OPTAscope probe CH2 to another I/O pin. Display both signals on the screen by selecting CH2 in the OPTAscope software. Do both channels change at the same time? Explain your answer.

Further Investigation

“What’s a Microcontroller ?”, Student Guide, Version 2.0, Parallax, Inc., 2003

Written by Andy Lindsay of Parallax, Inc., this text begins with detailed instructions for setting up and using your BASIC Stamp and Board of Education or HomeWork Board for the first time. Also introduced is digital output control with numerous examples that could be utilized with the ideas presented in this

text. It is available online from the Stamps in Class Curriculum menu on the Education page at www.parallax.com.

“XYZs of Oscilloscopes”, Tektronix, Tektronix 2003

Found at http://www.tektronix.com/Measurement/App_Notes/XYZs/, this article provides a very well documented tutorial for using oscilloscopes. The concepts demonstrated apply to oscilloscopes made by many different manufacturers.

Chapter #2: Servo Pulse Square Waves

2

PULSE WIDTH MODULATION AND HOBBY SERVOS

The focus of this chapter will be to measure and understand the pulses used to control servos. A servo is a type of tiny motor commonly used in radio-controlled hobby vehicles, and is also popular in robotics. Among the most difficult tasks for a hobbyist or amateur robotics enthusiast is to understand the timing of servo control as it relates to servo positioning.

The Parallax Boe-Bot and Toddler robots use servos for locomotion. Servos typically have a range of motion of 180°. When modified for continuous rotation, the standard hobbyist servo becomes a bi-directional, speed-controlled motor.

Servos have three leads: 6 V (red); signal (white), and ground (black). Servos are controlled by a sequence of pulses. Each pulse is 1-2 ms wide, and there must be ~20 ms of time between each pulse, as shown in Figure 2-1. The width of this 1-2 ms pulse sets the servo position (or speed/direction for modified servos) with 1.5 ms being the mid position (or stopped for modified servos). From a BASIC Stamp's perspective, this is quite convenient in that it can use the 20 ms of time between pulses to read sensors, perform calculations, and execute other code.

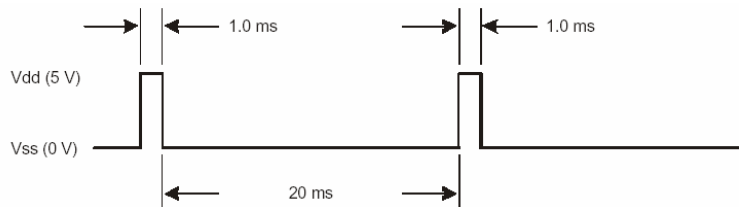


Figure 2-1:
Servo control
signal example

*A 1 ms pulse
positions the
servo in one
location.*

Hobby servo control is featured throughout Parallax Stamps in Class documentation. For more details on servo control, check out the following Stamps in Class texts:

What's a Microcontroller? Contains simple examples for controlling standard servos.

Robotics with the Boe-Bot The Boe-Bot uses continuous rotation servos to drive its wheels.

Advanced Robotics with the Toddler: The Toddler depends on precision control of standard servos to achieve its bipedal stride.

See the Further Investigation section at the end of this chapter for details.



The **PULSOUT** command is used in the following activities. It has this format:

```
PULSOUT 5, 750          ' 1.5 ms pulse on P5
```

The **PULSOUT** command's first argument, 5, states the BASIC Stamp I/O pin to be used, and the second argument, 750, is a variable or constant (0 – 65,535) that specifies the duration of the pulse in 2 µs increments. A pulse of 1.5 ms sent every 20 ms positions the servo in its centered location. Here is how this is calculated:

$$\left(\frac{\text{PulseWidth (ms)}}{2} \right) \times 1,000 = \text{Variable}$$

or

$$\left(\frac{1.5}{2} \right) \times 1,000 = 750$$

Now that we know the value for the **PULSOUT** command, we can build the experiment and program the BASIC Stamp. With an output pulse of 1.5 ms, the expected result from the servo is that it centers itself and stays put.



Servo power supply voltage is critical; applying more than ~6.5 VDC to servos will permanently damage them. If you wish to use the Board of Education Servo ports rather than the schematics and a 9 V battery in the following Activity, please note:

The Board of Education Rev B has four servo ports (P12-P15) appearing in a terminal block header on the right side of the board. The power supply for these ports is connected directly to Vin (the input voltage). If you are using a wall pack (anything more than +6 V) don't plug the servos into the terminal block – you'll put 9-15 V on the servos and they'll be destroyed. The terminal block is strictly for battery powered applications where the battery voltage is 6 VDC.

The Board of Education Rev C has a jumper selector that allows the user to select Vin or Vdd for the servo supply power. Consider your power supply and position this jumper appropriately if you choose to use the servo ports.

If you wish to use a higher input supply voltage, use the +5 V (Vdd) power supply above the breadboard, and plug the servos into the breadboard using the 3-pin male/male header pins. This will save many headaches and time lost destroying servos.

ACTIVITY #1: MEASURING PULSES FOR SERVO CONTROL

In this activity we will create and measure servo pulses using the OPTAscope's Paired Bars cursors function

Parts Required

- (1) Standard Parallax servo
- (1) 3-pin male/male header
- (1) 220 Ω resistor (for the Board of Education Revs B and C)
- (5) Jumper wires

Building the Servo Measurement Circuit

The circuits used in this chapter are built the same for the Board of Education Revisions B and C, but differently for the HomeWork Board. Schematics and wiring diagrams for both are included below.

Follow these directions for the Board of Education Rev. B and C:

- ✓ Insert the 3-pin male-male header into the servo lead. This will allow you to connect the servo directly to the breadboard.
- ✓ Build the circuit as shown in Figure 2-2 Figure 2-3:
- ✓ Connect the OPTAscope CH1 probe to P14 and the GND probe to Vss as shown in Figure 2-3:

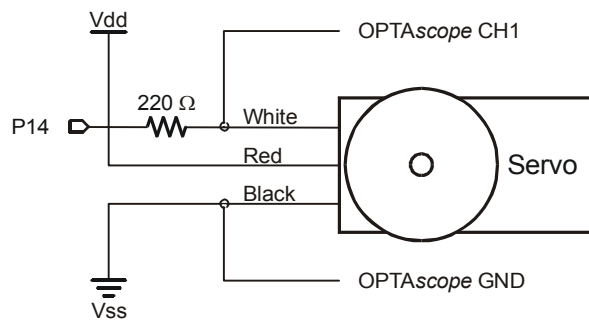


Figure 2-2:
Servo control
schematic for the
Board of Education
Revisions B and C

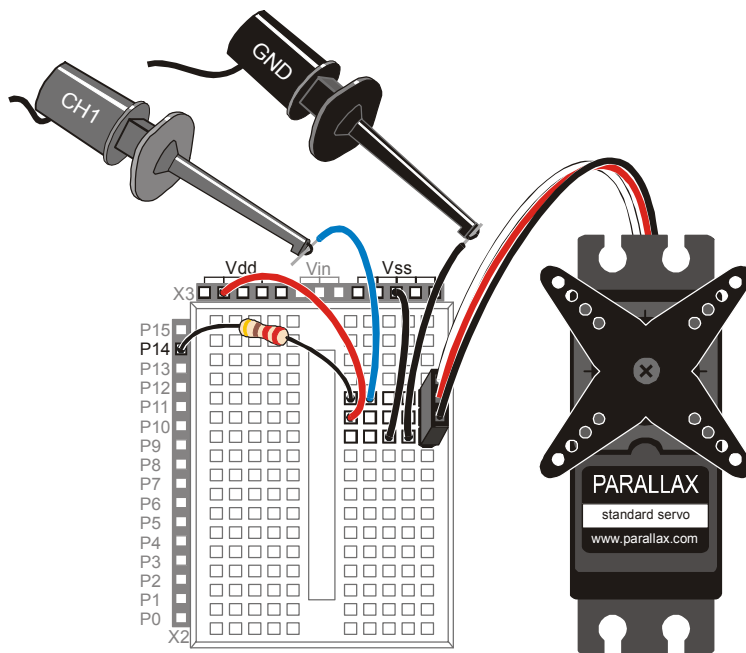


Figure 2-3:
Board of Education
Revisions B and C
servo connection
wiring diagram

Follow these instructions for the HomeWork Board:

- ✓ Insert the 3-pin male-male header into the servo lead. This will allow you to connect the servo directly to the breadboard.
- ✓ Build the circuit as shown in Figure 2-4 and Figure 2-5 below.
- ✓ Connect the OPTAscope CH1 probe to P14 and the Ground probe to Vss as shown in Figure 2-5.



Servo brand does matter! This circuit was designed to use the Parallax servo included in the Understanding Signals kit, which has a current draw of around 100 mA unloaded. If you are using a different servo that may have a higher current draw or that will be under a load, you may find that your BASIC Stamp resets due to brown-out conditions. This can be remedied by connecting a 3300 μF capacitor (not included) across V_{in} and V_{ss} .

CAUTION: be careful to observe the polarity of the capacitor! The positive terminal (longer leg) must connect to V_{in} (power source), and the negative terminal (shorter leg) must connect to V_{ss} (ground). If you insert the capacitor incorrectly, it may explode. Always disconnect the power source while building or modifying circuits. When working with a large capacitor, keep your hands away from the capacitor after the power is reconnected. Goggles are recommended.

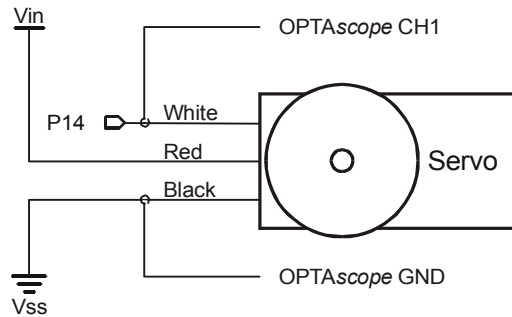


Figure 2-4:
Servo control schematic for
the HomeWork Board

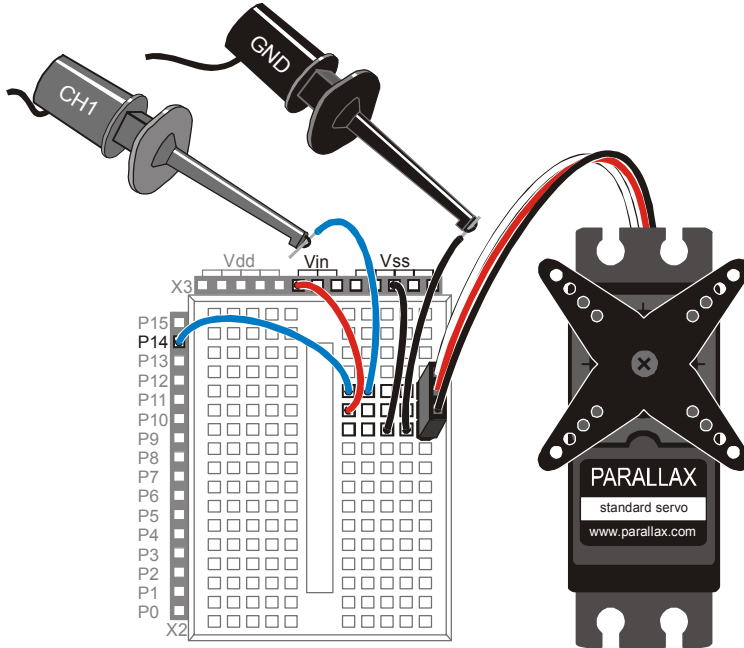


Figure 2-5:
BASIC Stamp
HomeWork Board
servo connection
wiring diagram

Configuring the OPTAscope Software

Continue the Activity in the same manner for all boards:

- ✓ Open the OPTAscope software and the BASIC Stamp Editor, then tile and position the windows to see both displays at once.
- ✓ Set up the OPTAscope as shown in Figure 2-6.

CH1	2 V / division
CH2	Off
Horizontal Dial	5 ms / division
Trigger Source	Channel 1
Trigger Edge	Falling
Trigger Mode	Auto
Run / Stop Mode	Continuous
Trigger Voltage	2 V

Figure 2-6:
Configuration for
OPTAscope to
measure servo
pulses.

Measuring the Servo Pulse Widths with ServoCentering.bs2

√ Run the program ServoCentering.bs2.

2

```
' Understanding Signals - ServoCentering.bs2
' Demonstrate a continuous pulse width for servo control

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

DO

  PULSOUT 14, 750          ' 1.5 ms pulse
  PAUSE 20                 ' 20 ms pause

LOOP
```

Each loop includes a short pulse of 1.5 ms followed by a 20 ms delay (Figure 2-7).

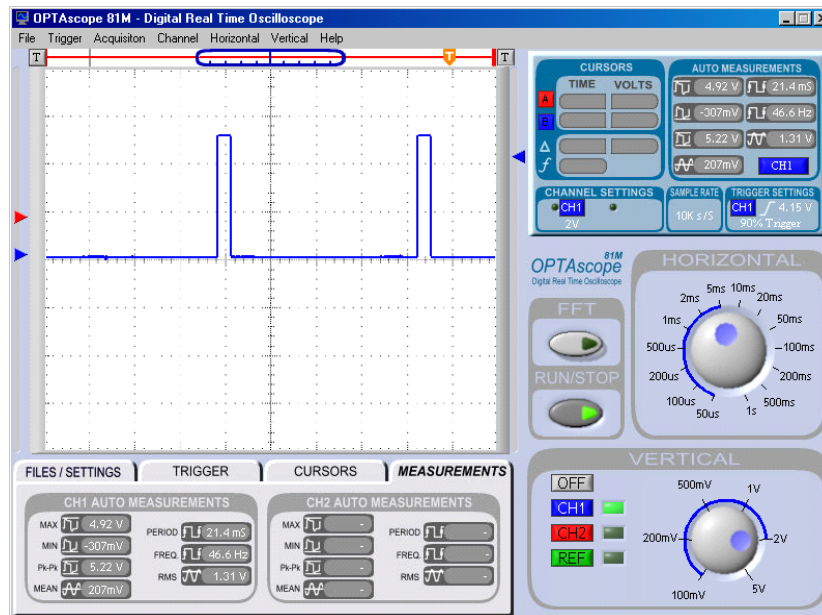


Figure 2-7:
Servo centering pulses.

√ Under the Cursors tab slide the Cursor Settings switch to Paired Bars. Place the red and blue vertical cursors on either side of the pulse to measure the width.

If all goes well, your OPTAscope screen will look like Figure 2-8 and the Auto Measurements box will show the width of the pulse.

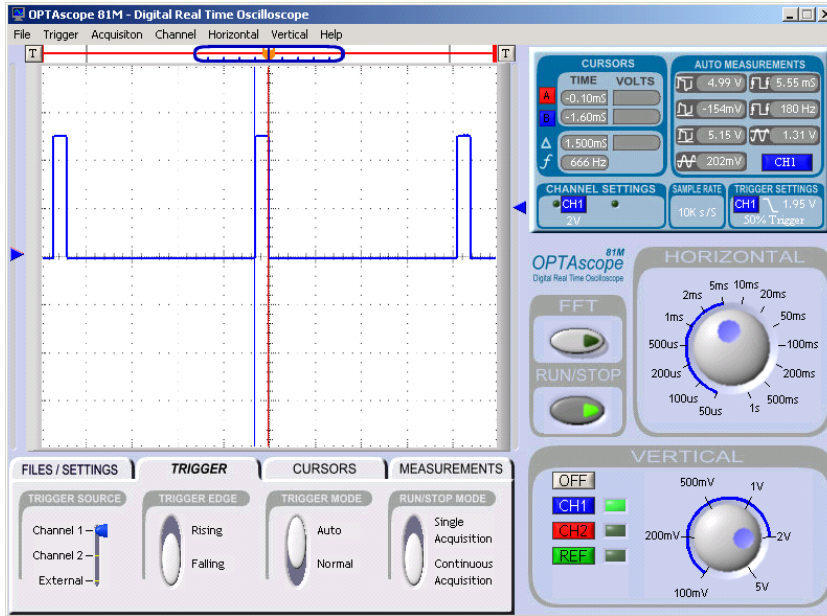


Figure 2-8: Configure the vertical pairs to measure the pulse width.

The Display Screen's Cursors box will indicate that the pulse measures 1.5 ms wide, as shown in Figure 2-9.

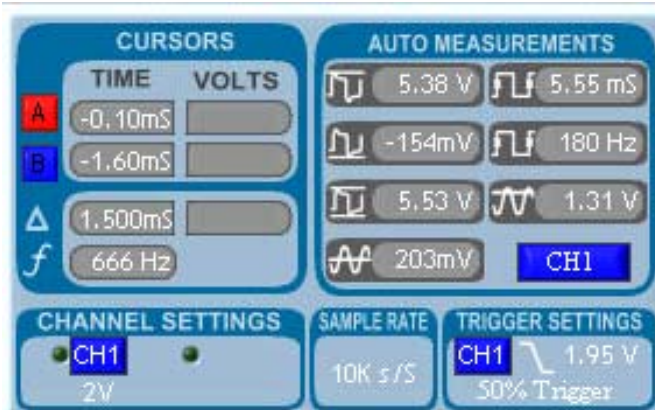


Figure 2-9:
The delta symbol Δ shows a pulse width of 1.500 ms.

2



Are your measurements different? Calibrate! If your measurements vary from those in the text, you may try calibrating your OPTAScope to see if that is the source of the discrepancy. To begin, disconnect all probes from your circuit. Then select Calibrate from the File drop-down menu on the OPTAScope task bar, and follow the directions.

ACTIVITY #2: MEASURING TIME-VARYING SERVO PULSES

Use the same parts and circuitry as the previous activity. No changes to the OPTAScope settings are required.

Time-Varying Pulse Widths with the ServoSweep.bs2 Program

- ✓ Run the program ServoSweep.bs2, which will vary the pulse width.
- ✓ Press the Run/Stop button when the pulse is at its widest.
- ✓ Measure the pulse width using the Vertical Bars cursor setting.
- ✓ Repeat this process to measure the pulse at its narrowest extreme.

```
' Understanding Signals - ServoSweep.bs2
' Demonstrate a changing servo pulse

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

PulseWidth      VAR Word           ' Pulse length
Counter         VAR Word           ' Pulse counter
```

```
DO
  FOR PulseWidth = 400 TO 1200 STEP 10      ' 100us steps 170 degrees
    DEBUG DEC PulseWidth," uS",CR          ' Display value in uS
    FOR Counter = 1 TO 10                    ' 10*20ms = 200 ms/step
      PULSOUT 14, PulseWidth
      PAUSE 20                               ' Between 20 and 50 ms
    NEXT
  NEXT
  FOR PulseWidth = 1200 TO 400 STEP 10      'Go back the other way
    DEBUG DEC PulseWidth," uS",CR
    FOR Counter = 1 TO 10
      PULSOUT 14, PulseWidth
      PAUSE 20
    NEXT
  NEXT
LOOP
```


Summary

This chapter demonstrated how to measure a servo pulse signal using the Paired Bars cursor settings option. This chapter also provided a visualization of servo control signals, which will aid in understanding timing and control in your other BASIC Stamp robotics programs.

Exercises

1. In the ServoSweep.bs2 program, a **PAUSE** command is used to pace the pulses sent to the servo. What are the smallest and largest values that you can use and still maintain useful servo control? Program the BASIC Stamp with a larger delay between pulses, observe the results and measure with the OPTAscope.
2. Explain the difference noticed in the Plot Area when you set the Trigger Edge switch from Rising to Falling and back again.
3. Explain the correlation between the pulse width variation and servo position you observed in Activity 2.
4. Set the Mouse Function switch to Zoom to take a very close look at the leading edge of a servo pulse. Is it made up of perfectly straight lines? Explain your answer.
5. While the OPTAscope is still monitoring the servo pulses, set the time base to 2 ms/div using the Horizontal dial. Set the Trigger Edge switch to Falling, and set the Trigger Mode switch to Normal. What do you see? Why is that so?

Further Investigation

“Advanced Robotics with the Toddler”, Student Guide, Version 1.2, Parallax, Inc., 2003

This text, authored by Ken Gracey of Parallax and Bill Wong, can be viewed as an in-depth application of standard servo control. Chapters 2 through 4 provide a series of experiments that program the Toddler to walk bipedally through the precise control of two standard servos. It is available online from the Stamps in Class Curriculum menu on the Education page at www.parallax.com.

“What’s a Microcontroller”, Student Guide, Version 2.0, Parallax, Inc., 2003

In this text written by Andy Lindsay, Chapter 4 provides numerous servo control examples. It is available online from the Stamps in Class Curriculum menu on the Education page at www.parallax.com.

“Robotics with the Boe-Bot”, Student Workbook, Version 1.5, Parallax, Inc., 2001

Chapter 2 provides examples with continuous rotation servos. It is available online from the Stamps in Class Curriculum menu on the Education page at www.parallax.com.

Chapter #3: Sine Waves

SINE WAVES WITH THE BASIC STAMP FREQOUT COMMAND

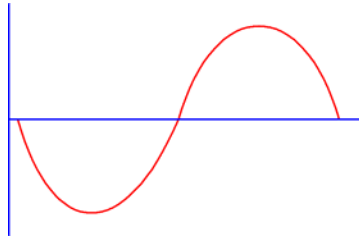


Figure 3-1:
Typical sine wave – one cycle.

3

A sine wave is a common electrical signal that can be viewed on the OPTAscope. Unlike digital signals that typically are either high or low, analog signals can be high, low, or any value in between. One of the most common types of sine waves that we can sense is the sound wave. Sound waves permeate the air in three dimensions in the same manner that ripples flow across a pond in two dimensions. A cross-section of the surface of this pond would look something like the waveform in Figure 3-1.

Despite the fact that BASIC Stamps are digital by their nature, they can generate sine waves. The interesting thing here is *how* they generate sine waves. The BASIC Stamp must approximate the sine wave with a sequence of digital square waves that is later filtered into a nice, clean sine wave. When you connect a piezoelectric speaker to a sine wave you will hear a tone (provided the sine wave is oscillating at a frequency within the audible range). This may sound confusing, but it will clear up as we work through the next lesson. First, we need a little more background information.

The pulse train used to control servos is only one type of signal in a family of signals called PWM. PWM stands for pulse width modulation. The servo pulse train is unique in that it has a fixed “off-time”. Most PWM signals have variable off-time as well as variable on-time. Consider the pulse train depicted in Figure 3-2.

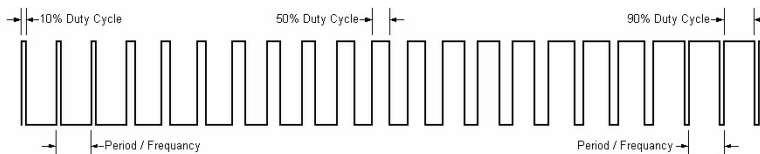


Figure 3-2:
PWM signal train

Note the ratio of on-time vs. off-time as the signal progresses from left to right. You can see that the on-time is progressively increasing while the off-time is progressively decreasing. If this signal were filtered into a smooth DC signal, we would see an analog voltage starting close to zero and growing to a voltage close to 5 VDC. So, as the duty cycle increases, the average DC value increases. This is the fundamental principle upon which the BASIC Stamp recreates sine waves.

Using the PWM output from the BASIC Stamp, a small RC filter, and a piezo speaker, you can create a sequence of square waves, sine waves, and sound waves that are viewable on the OPTAscope.

ACTIVITY #1: SINE WAVE SURFING

The goals of this activity are to:

- Create a sine wave.
- Capture and view it on the OPTAscope.
- Vary the frequency of the sine wave and note the change in pitch and volume.
- Experiment with the trigger level.

Parts Required

- (1) Piezo speaker
- (1) 220 Ω resistor
- (1) 1.0 μF capacitor
- (6) Jumper wires



WARNING: The 1.0 μF capacitor can explode if it is connected improperly!

Always disconnect the power before building or modifying circuits. Carefully check the polarity of the 1.0 μF capacitor. The positive terminal (longer leg) must be connected to a power source pin and the negative terminal (shorter leg) must be connected to Vss (ground).

Building the Sine Wave Measurement Speaker Circuit

- ✓ Build the circuit as defined by the schematic in Figure 3-3. Awiring diagram of the circuit is shown in Figure 3-4. If using the HomeWork Board, omit the 220 Ω resistor as one is already built in; replace with a jumper wire to make the necessary connection.

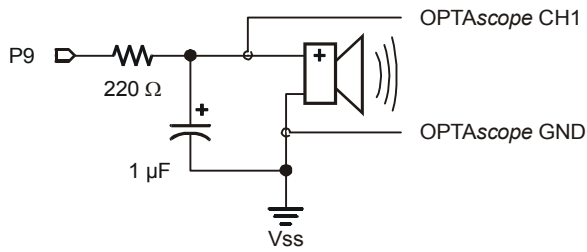


Figure 3-3:
Sine wave circuit

Note: If using the HomeWork Board, leave out the 220 Ω resistor. It is already built into the board.

3

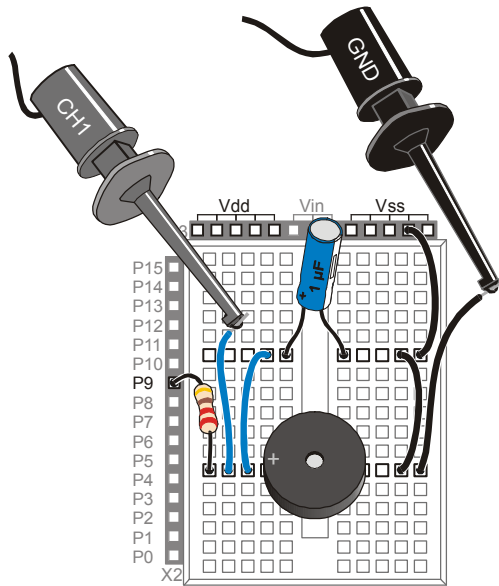


Figure 3-4:
Sine wave speaker wiring diagram

Note: If using the HomeWork Board, replace the 220 Ω resistor with a jumper wire. A 220 Ω resistor is already built into the board.

Configuring the OPTAscope Software

- √ Configure the OPTAscope with the settings shown in Figure 3-5.

CH1	2 V / division
CH2	Off
Horizontal Dial	1 ms / division
Trigger Source	Channel 1
Trigger Edge	Rising
Trigger Mode	Auto
Run / Stop Mode	Continuous
Trigger Voltage	2 V

Figure 3-5:
Configuration for
the OPTAscope to
measure sine
waves



FREQOUT *pin, duration, freq1* {*freq2*}.

FREQOUT is a BASIC Stamp command that generates a series of pulses that approximate a sine wave when filtered. This sine wave can be translated into sound with a piezo speaker, assuming the frequency is within the audible range. The **FREQOUT** command requires the following arguments: *pin* (0-15) is the I/O pin used, *duration* (1-65535) is the length of the tone in ms., *freq1* is the frequency in Hertz, *freq2* is an optional second frequency, also in Hertz. For detailed information, open the BASIC Stamp Windows Editor Help file.

Triggering a Sine Wave with TestPiezoWithFreqout.bs2

✓ Run the program TestPiezoWithFreqout.bs2.

```
' Understanding Signals - TestPiezoWithFreqout.bs2
' Send a tone to the piezo speaker using the FREQOUT command.
'{$STAMP BS2}
'{$PBASIC 2.5}
DO
  FREQOUT 9, 60000, 200
LOOP
```

With the CH1 probe connected you will see the signal shown in Figure 3-6. The **FREQOUT** command in this program generates a 200 Hz signal for 60000 ms (1 minute) from I/O pin 9. However, since the command is nested in a **DO..LOOP**, the command is immediately repeated and the signal is generated continuously.

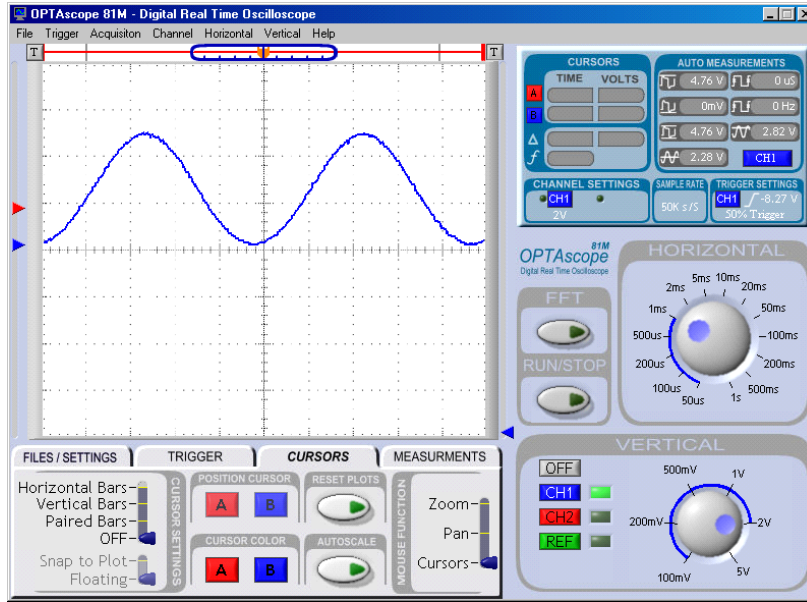
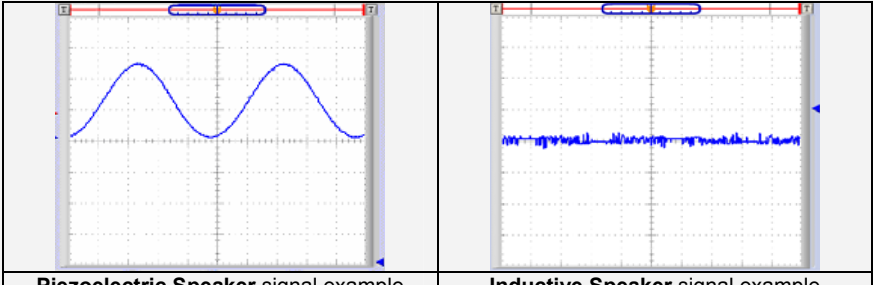


Figure 3-6:
200 Hz sine
wave

3

Signal Problems? The piezo speaker signal shown below-left resembles the one you can reasonably expect to see when using the piezo speaker included in the Understanding Signals kit. If you are using a different speaker, your OPTAscope display might more closely resemble the inductive speaker signal shown below-right. Some speakers look like piezo speakers, but they use a coil (an inductive element) instead of a piezoelectric element to vibrate the surface that generates sound. Because the properties of a coil are very different from the properties of a piezoelectric element, the circuit is changed drastically, and so is the signal that is measured and displayed by the OPTAscope.

Remedy: If you do not have a true piezoelectric speaker at your disposal, you can still view the signal by removing the speaker from the circuit shown in Figure 3-3 and Figure 3-4. When you re-run the program and capture the signal, it should more closely resemble the piezo speaker signal example.



i

The trigger is set with the blue arrow to the right of the Plot Area. As you move it up and down you will see the signal move to the right and left. This is because the trigger event lines up in the center of the screen. A trigger event happens when the input signal crosses the trigger voltage on a rising edge. Let's experiment!

- √ Adjust the trigger level up and down to see the sine wave shift left or right.
- √ Set the Trigger Edge switch to Falling, then experiment with the trigger level.
- √ Set the Trigger Edge switch back to Rising. Change the **FREQOUT** command's **freq1** argument; try several different frequencies. Note: you will have to download the revised program into your BASIC Stamp or HomeWork Board each time.
- √ For each frequency tested (at least five), adjust the trigger level to capture a nice image of the sine wave. Make a mental note regarding the volume and pitch differences.

ACTIVITY #2: SINE WAVE FREQUENCY AND AMPLITUDE MEASUREMENT

This activity uses the same circuit you have already built. This activity will measure the frequency and amplitude of a sine wave and compare it to the BASIC Stamp code that generated the signal.

Measuring Frequencies with TestPiezoWithFreqout.bs2

- ✓ Re-run the program TestPiezoWithFreqout.bs2.
- ✓ Press the Run/Stop button to view the signal, and then press again to capture it.
- ✓ Under the Cursors tab, set the Cursor Settings switches to Paired Bars and Floating.
- ✓ Place the red horizontal cursor on the bottom of the signal.
- ✓ Place the red vertical cursor at the peak at the left (Figure 3-7). Now place the blue horizontal cursor on the top of the signal and the blue vertical cursor on the peak to the right.

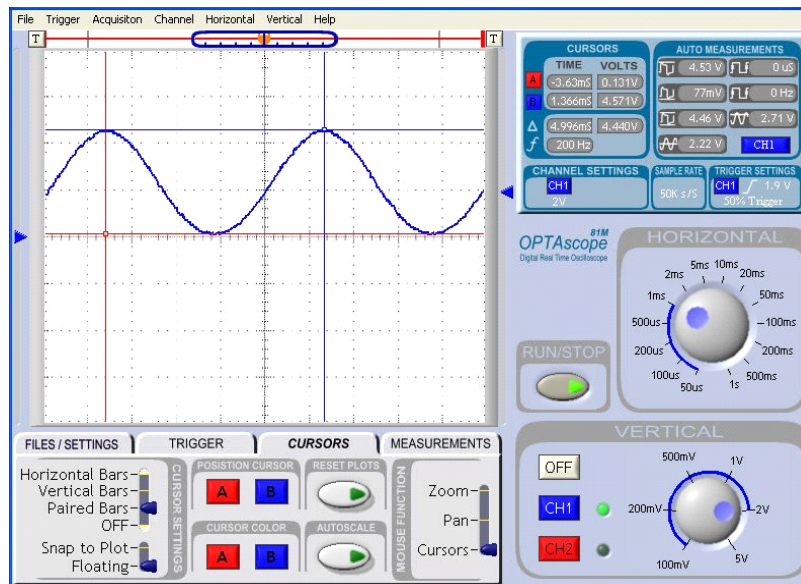


Figure 3-7: Measuring sine wave frequency and amplitude with the Paired Bars cursors

Review the data in your Cursors measurement box. Does it compare to your BASIC Stamp source code? Your Δ should measure around 5.0 V peak-to-peak, and 200 Hz for

the f (frequency) as shown in Figure 3-8. That's just what we programmed the BASIC Stamp to generate - a 200 Hz sine wave!

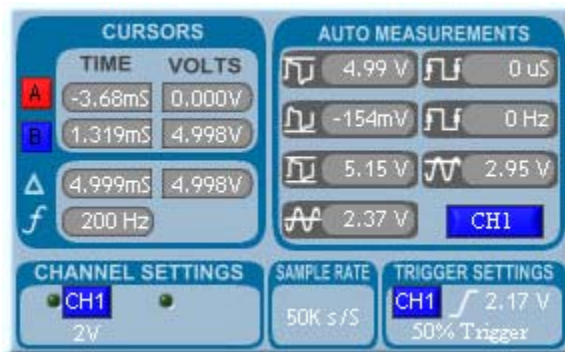


Figure 3-8: Measuring peak-to-peak voltage and frequency

- √ To generate a 500 Hz signal, change the **FREQOUT** command's *freq1* argument to read:

```
FREQOUT 9, 60000, 500
```

- √ Remember to keep the cursors in the same exact location described before. You will use those cursors to compare the differences of this higher frequency.

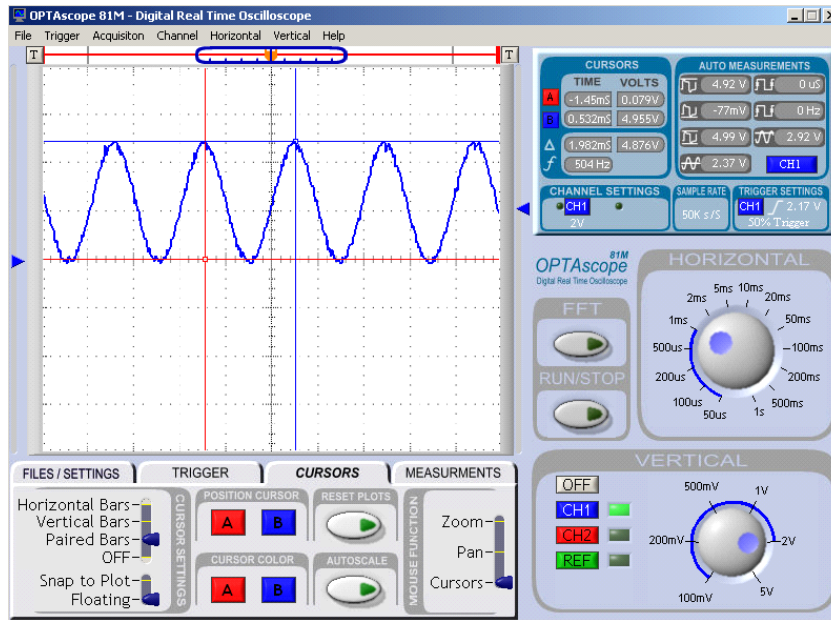


Figure 3-9:
500 Hz
frequency

3

- ✓ First move the vertical cursors to the centers of the two peaks in the middle to measure the frequency of the sine wave. You should get 500 Hz.
- ✓ Record your ΔV and f measurements, and make a note about volume and pitch.
- ✓ Repeat this process for five more frequencies in the 1000-5000 Hz range, again making a note of your ΔV and f measurements, volume and pitch. Hint: try setting the Mouse Function switch to Zoom when measuring the 5000 Hz signal.
- ✓ Make a chart of your data. What can you infer from your five amplitude measurements?

ACTIVITY #3: DUAL SINE WAVE MEASUREMENT

The BASIC Stamp's `FREQOUT` command provides the ability to generate two sine waves on the same pin. In this step, we'll add the second frequency. Pure tones are generated by single sine waves. However, most tones you hear coming from your average electronic consumer goods are actually mixed tones. The previous exercises generated a pure tone, while the next exercise combines two different frequencies. In this way you will be able to hear, and see, the difference between pure and mixed tones.

Dual Sine Waves with the DualSineWaves.bs2 Program

You will again reuse the circuit from the previous two activities.

- √ Set the Vertical dial to 1 V/division and the Horizontal dial to 200 μ s.
- √ Run the program DualSineWaves.bs2.

```
' Understanding Signals - DualSineWaves.bs2
' Send two sine waves to the piezo speaker using the FREQOUT command

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

DO

    FREQOUT 9,10000,2000,6000

LOOP
```

You should see an unusual signal like the one shown in Figure 3-10. This repetitive signal is composed of the 2 kHz and 6 kHz sine waves mixed together.

- √ Set the Cursor Settings switches to Vertical Bars and Floating.
- √ Carefully measure 2 kHz and 6 kHz signal components.

The 2 kHz component is easily discernible as you can measure from peak to peak of the mixed signal. The 6 kHz component is also visible as the smaller curves at the top and sides of each wave.



Why does my signal look different if I change the capacitor value? The RC network that converts the digital pulses sent by the BASIC Stamp's **FREQOUT** command into sine waves does two things: 1) reduces the amplitude of the sine wave, called signal attenuation, and 2) moves the sine wave to the right, which is called a phase shift. The resulting sine wave's amplitude and phase shift is a function of $R \times C$ and of the frequency of the sine wave. When **FREQOUT** sends two different frequencies at once, each component sine wave is attenuated and phase shifted separately by the RC circuit. If you change the value of C in $R \times C$, the overall shape of the signal changes because the component sine waves are attenuated and phase shifted differently. If you complete this activity with a different value capacitor and your resulting waveform differs from the one below, you can still use the FFT function to measure that the same component frequencies are present.

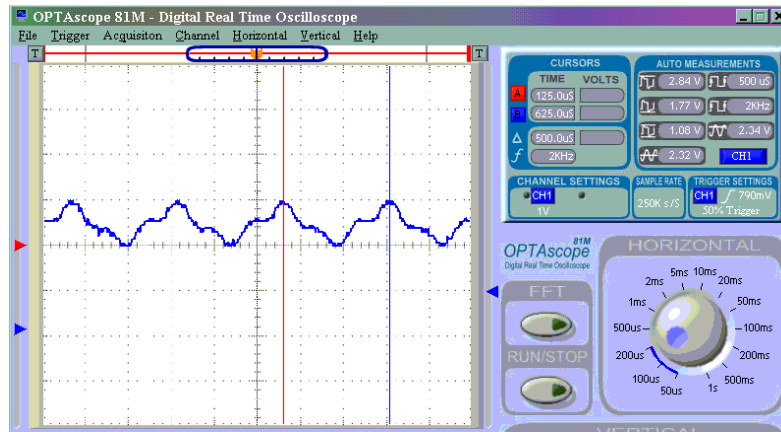
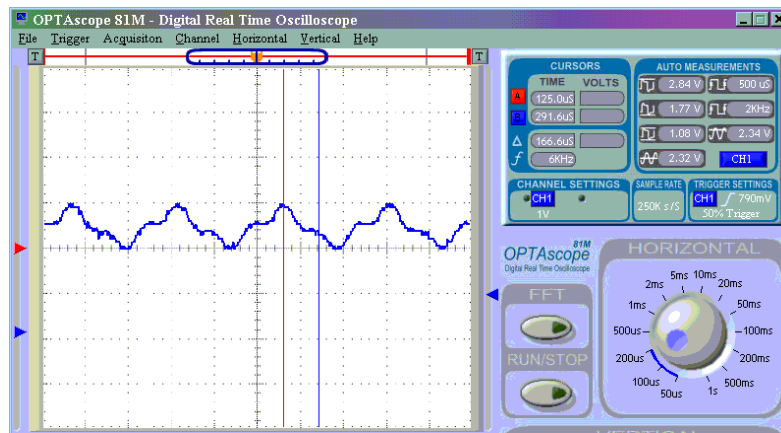


Figure 3-10:
Mixed Frequency
Signal

The two
component sine
waves can be
discerned with the
Vertical Bars
cursors.

The 2 kHz
component is
measured in the
top picture.



The 6 kHz
component is
measured in the
bottom picture.

Fast Fourier Transformation with the DualSineWaves.bs2 Program

The OPTAscope's Fast Fourier Transformation (FFT) function emulates a device called a spectrum analyzer by displaying the sine wave frequencies contained by a signal. Fourier analysis is used extensively by the radio communication industry to prevent people from accidentally broadcasting signals on other channels. This type of analysis can also be used to look at things like engine vibration, where certain frequencies of vibration can indicate particular problems. FFT is a vast subject, but this exercise will give you a brief introduction by analyzing the mixed signal generated in the previous Activity.

- √ Set the Horizontal dial to 1 ms. You should see a compressed view of the mixed signal.
- √ Press the FFT button, and the FFT Window will open (Figure 3-11).
- √ Select Black-Harris in the Windowing box.
- √ Click on the ON/OFF button in the Cursor box to turn on the FFT cursor.
- √ Locate the FFT cursor, a white vertical line with a square handle all the way to the left in the FFT Plot Area.
- √ Move the cursor to the first peak to the right.
- √ Read the Frequency measurement in the Cursor box.
- √ Repeat to measure the second peak.

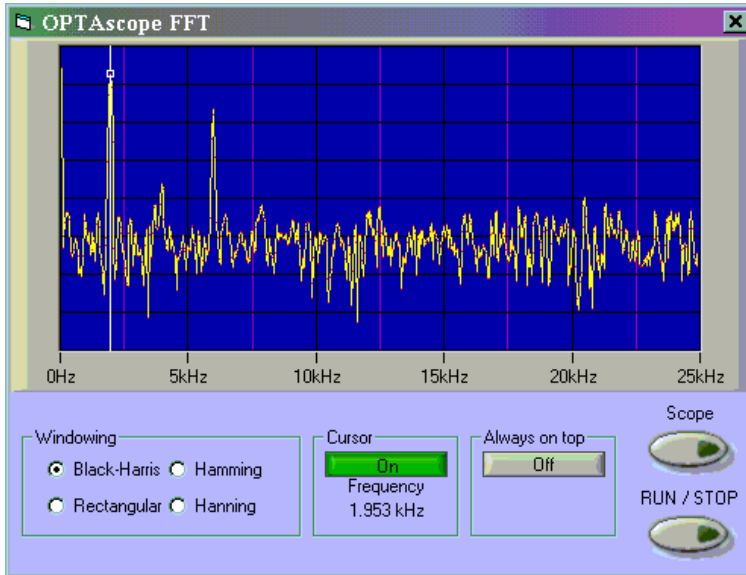


Figure 3-11:
The OPTAscope
FFT Window.

*Note that the FFT
cursor measures
near 2 kHz at the
first peak.*

Your two measurements should be very near 2 kHz and 6 kHz.

Summary

Even though the OPTAscope is a digital oscilloscope, it is well suited to view both digital and analog signals. Digital signals are usually either high or low, whereas analog signals can be high, low, or at any voltage in between. Sound waves are made up of sine waves.

The nature of sound is complex: only pure tones are made with one frequency, but most applications use tones made of mixed frequencies. Proper use of the OPTAscope can help you identify the component frequencies within a complex sine wave. You may use the Vertical Bars cursors to measure the pattern in a simple mixed signal, and use the FFT function to identify component frequencies. Since wave amplitudes can vary with frequency, adjustment of the trigger level may be required to keep them “in focus”. The Cursor Settings switch’s Paired Bars setting option allows you to measure both frequency and amplitude.

Exercises

1. Write a program to generate a signal like the one shown in Figure 3-12 below. The primary frequency is 800 Hz and the secondary is 8 kHz. You may need to adjust your Horizontal and Vertical dials to get a similar signal. Give it a try!

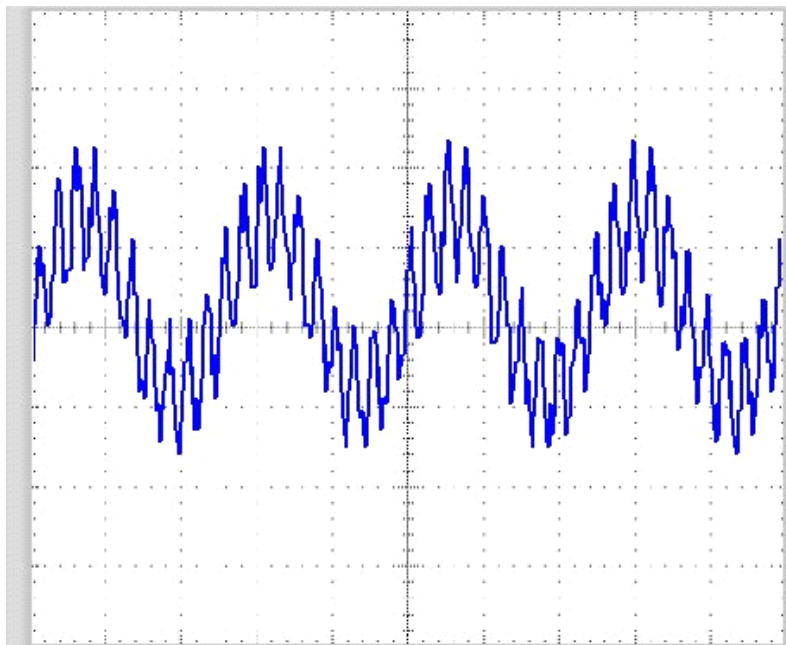


Figure 3-12:
An 8 kHz sine wave superimposed upon an 800 Hz sine wave

2. Use the FFT function to confirm the components of your signal.

Further Investigation

“What’s a Microcontroller”, Student Guide, Version 2.0, Parallax, Inc., 2003

In this text by Andy Lindsay, Chapter 8 features an overview of sound generation, providing a wide variety of frequencies to view and measure. It is available online from the Stamps in Class Curriculum menu on the Education page at www.parallax.com.

Chapter 4: R/C Circuits and Variable Resistors

Resistors and capacitors are integral to analog electronics. Anything you do with analog electronics will involve resistors and capacitors. Understanding how they react with each other is important. This chapter will demonstrate how to use the OPTAscope to view charge and discharge curves of capacitor and resistor networks.

4

WHAT ARE CAPACITORS?

Capacitors have two main behaviors in electronic circuits: charging and discharging. When a voltage is applied to a capacitor, the capacitor charges. When the charge in the capacitor equals the voltage applied, the capacitor stops charging. The charge will remain in the capacitor with or without the voltage applied to the capacitor.

The capacitor will discharge when a path for current to flow through is placed across the capacitor. The capacitor will discharge until the charge across the capacitor is completely dissipated, thereby equaling zero volts. A capacitor can repeat this charge and discharge cycle without the “wear and tear” that a battery would endure. Think of a capacitor as a temporary power source that is charged by applied voltage, and then can supply voltage back into the circuit when the applied voltage sinks or disappears.

RESISTORS AND CAPACITORS IN RC NETWORKS

A resistor will resist the flow of current. The larger the value of the resistor, the more it will resist current flow. Capacitors charge and discharge very quickly. By placing a resistor in series with a capacitor you can precisely control the rate at which a capacitor will charge or discharge. This is called an RC network or RC circuit.

By controlling the amount of capacitance and resistance, you can control the charge and discharge rate, or “curve”, of the RC network. The higher the values of the capacitor and resistor, the longer the time it will take for the capacitor to charge. We can calculate how long the charge and discharge curve will be by calculating the time constant. Here is the formula:

$$\text{Time Constant} = R \times C$$

Where R is in Ω s

Where C is in Farads ($1\mu\text{F}$ would be .000001 F, $.1\mu\text{F}$ would be .0000001 F).

Time Constant is in seconds

Let's calculate the time constant for a $220\ \Omega$ resistor and a $1\mu\text{F}$ capacitor.

$$\text{Time Constant} = 220 \times 0.000001$$
$$\text{Time Constant} = 0.00022 \text{ seconds or } .22 \text{ ms}$$

The time constant is the time it will take for the charge curve of the capacitor to reach 63% of the applied voltage or to fall to 37% of the applied voltage. This may seem odd until you consider the following images.

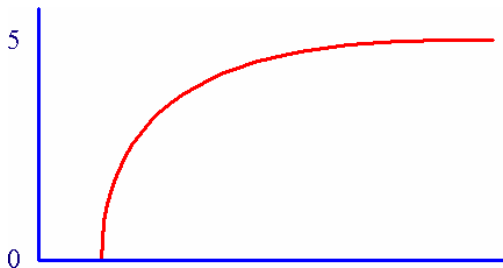


Figure 4-1:
RC network charge curve.

Note that the capacitor charges quickly at first, then it progressively takes longer and longer to completely charge.

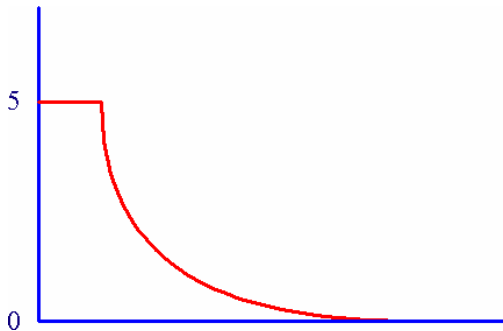


Figure 4-2:
RC network discharge curve.

Likewise, the capacitor discharges quickly at first, then it progressively takes longer and longer to completely discharge.

Consider the following example: If you applied 5 V to this RC network, it would take 2 ms for the charge curve to reach 63%, or 3.15 V. It will take approximately 5 times the time constant (10 ms) for the capacitor to charge to 5 V. Provided you allow the capacitor

to charge for at least 10 ms (to 5 V) then disconnect the power supply, it will take 2 ms for the capacitor to discharge to 37% or 1.85 V.

This use of capacitors and resistors allows you to use the BASIC Stamp to measure resistance and capacitance. Because the time constant is always the same for an RC network, we can measure the discharge time with the **RCTIME** command. The logic threshold for BASIC Stamps is 1.4 V, meaning above 1.4 V the BASIC Stamp detects logic 1 and below 1.4 V it detects logic 0. To measure the RC constant or time, set the I/O pin high for several milliseconds allowing the capacitor to charge to 5 V. Then the **RCTIME** command will make the I/O pin an input and count the time it takes for the capacitor to discharge below 1.4 V.

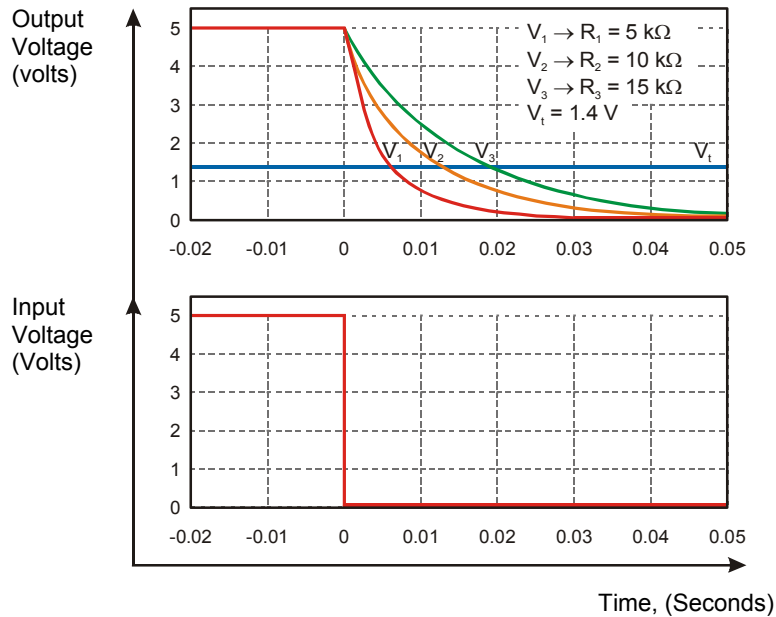


Figure 4-3: RC circuit responds differently to three different resistance values.

Note how each output takes a different amount of time to decay, or discharge, from 5 Volts to 1.4 Volts.

Different resistor sizes affect the charge or discharge time. The signal on the bottom of Figure 4-3 indicates when the I/O pin was set to an input, thereby making it high impedance at Time = 0 seconds. At this point the capacitor starts to discharge through the series resistor. The BASIC Stamp counts how long it takes the capacitor to fall to 1.4 V in 2 μ s increments.

ACTIVITY #1: VERIFYING THE CALCULATED RESISTOR/CAPACITOR NETWORK TIME CONSTANT

In this activity we will measure the resistor/capacitor time constant and verify the value with an OPTAscope measurement.

Required

- (1) 220 Ω resistor
- (1) 10 μF capacitor
- (5) Jumper wires



WARNING: The 10 μF capacitor can explode if it is connected improperly!

Always disconnect the power before building or modifying circuits. Carefully check the polarity of the 10 μF capacitor. The positive terminal (longer leg) must be connected to a power source pin and the negative terminal (shorter leg) must be connected to Vss (ground).

Building the RC Time Constant Circuit

- ✓ Build the circuit as shown in Figure 4-4 and Figure 4-5 , paying attention to the polarity of the capacitor. If you are using a BASIC Stamp HomeWork Board leave the 220 Ω resistor out. Use a jumper wire to make the necessary connection.
- ✓ Attach the CH1, CH2 and Ground probes as shown in Figure 4-4.

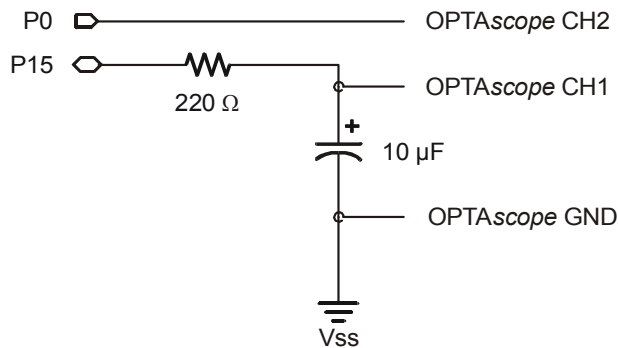


Figure 4-4:

Resistor / capacitor circuit for RC time constant measurement

Note: If you are using the BASIC Stamp HomeWork Board, do not put the 220 Ω resistor in the circuit (this resistor is already surface mounted onto the PCB).

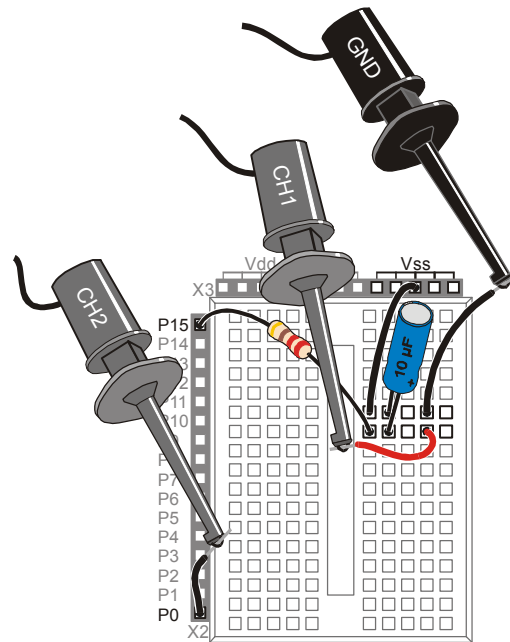


Figure 4-5:
RC circuit wiring
diagram with
OPTAscope probes

Note: If you are using the BASIC Stamp HomeWork Board do not put the 220 Ω resistor in the circuit (this resistor is already surface mounted onto the PCB).

Instead, use a jumper wire to make the necessary connection.

4

Configuring the OPTAscope Software

√ Configure the OPTAscope as shown in Figure 4-6.

CH1	2 V / division
CH2	2 V / division
Horizontal Dial	5 ms / division
Trigger Source	Channel 1
Trigger Edge	Rising
Trigger Mode	Auto
Run / Stop Mode	Continuous
Trigger Voltage	2 V

Figure 4-6:
Configuration for
OPTAscope to
measure R/C
curves

RCTimeConstant.bs2 to Demonstrate Charge/Discharge Curves

✓ Run the program RCTimeConstant.bs2.

```
' Understanding Signals - RCTimeConstant.bs2
' Read photoresistor in RC-time circuit using RCTIME command.

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

DO

HIGH 0           ' 5 V on P0
HIGH 15         ' 5 V to capacitor for charge curve
PAUSE 10        ' time for capacitor to fully charge
LOW 0           ' ground capacitor for discharge curve
LOW 15         ' same to P0 for comparison
PAUSE 10

LOOP
```

✓ Arrange your display using the red/blue arrows on the left to separate the two signals as shown in Figure 4-7.

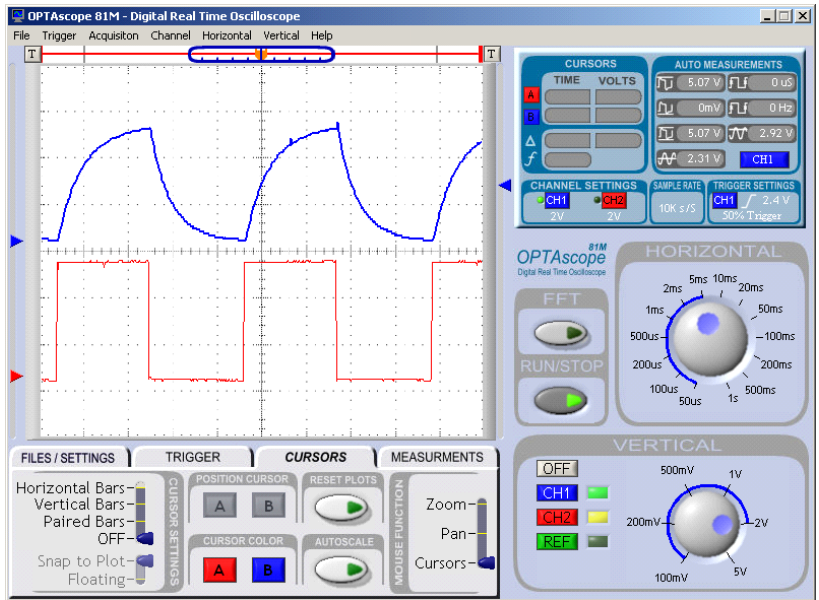


Figure 4-7: OPTAscope display of RC circuit with a 220 Ω resistor and a 10 μF capacitor

On Channel 1 you will see a charge-discharge curve. On Channel 2 you will see a square wave. Channel 2 displays what the signal would look like without the RC network. This also shows when the capacitor starts to charge and discharge each cycle.

Notice that when 5 V was applied to the RC network the capacitor charged at a specific rate. Then, when the 5 V was removed, the capacitor discharged at the same specific rate. The discharge curve is merely the inverse of the charge curve, and vice-versa.

Measuring RC Time Constants against Calculated Values

Let's verify through measurement that the RC time constant we calculated with a 220 Ω resistor and 10 μF capacitor is properly shown by the OPTAscope by taking a measurement.

- √ Click on the Run/Stop button to hold the signal in place.
- √ Under the Cursors tab, set the Cursor Settings switches to Paired Bars and Snap to Plot.
- √ Place the red vertical cursor at the point where CH1 starts to rise.
- √ Click the A and B Position Cursor buttons if the cursors are not yet tracking CH1.
- √ Move the blue horizontal cursor to 3.15 V. It should automatically track the signal. If you “lose” a vertical cursor bar, slide the Plot Area Indicator bar left or right; you should be able to find your “missing” cursor and drag it back into the view area. Your screen should look like Figure 4-8.
- √ You should measure around 2.2 ms as the delta, Δ , for the time constant. (Your measurement may vary from this, resistors and capacitor values can vary from their stated value by a given percentage range, usually marked on the device.) The Δ for the first time constant refers to the time it takes for an RC network to charge up from 0 VDC to 3.15 VDC. This is the first of 5 “TCs”, or time constants. It takes 5 TCs to fully charge (or discharge) a capacitor.

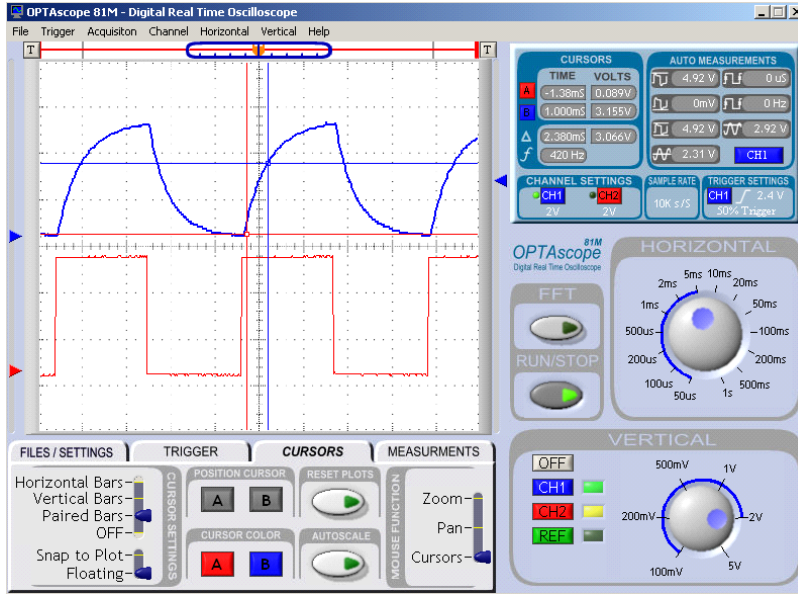


Figure 4-8: Paired Bars positioned at the start of the capacitor charge (0V) and at 3.15 V

- ✓ Replace the 220 Ω resistor with a 1 k Ω resistor. If you are using the HomeWork Board, remove the jumper wire you used instead of the 220 Ω resistor, and put the 1 k Ω resistor in its place.
- ✓ Set the OPTAscope's Horizontal dial to 20 ms.
- ✓ In your program RCTimeConstant.bs2, change both instances of the **PAUSE** command to read

PAUSE 50

- ✓ Run the modified program.
- ✓ Take the same measurements with the Paired Bars cursors that you did previously to determine the first TC of the new curve.

Now how long does it take for the capacitor to charge to 3.15 V? How long does it take to fully charge to 5 V?

By increasing the value of the resistor, the TC duration also increases. You can see that the signal rises much more slowly. If you are using the Board of Education, the TC with a new 1 k Ω resistor will be calculated as follows.

$$\text{Time Constant} = 1000 \times 0.00001$$

$$\text{Time Constant} = 0.01 \text{ seconds or } 10 \text{ ms}$$

If you are using the HomeWork Board, your calculation must account for the added resistor value:

$$\text{Time Constant} = (220 + 1000) \times 0.00001$$

$$\text{Time Constant} = 1220 \times 0.00001$$

$$\text{Time Constant} = 0.0122 \text{ seconds or } 12.2 \text{ ms}$$

4

Again, your actual measurements may vary in accordance with the tolerances of the resistor and capacitor you use.

ACTIVITY #2: VARIABLE RESISTORS IN AN RC NETWORK

In this activity we will measure the discharge time of an RC network and compare it to the BASIC Stamp **DEBUG** values. Then, we will calculate the value of a variable resistor (such as a photoresistor) with a known capacitor and resistor and measure it against the OPTAscope to see how the two values compare.

Parts Required

- (1) 220 Ω resistor
- (1) 10 μF capacitor
- (1) Photoresistor
- (5) Jumper wires

Building the Photoresistor RC Network Circuit

Add a photoresistor to amend the circuit you previously built to match the schematic (Figure 4-9) and wiring diagram (Figure 4-10). The 220 Ω resistor and 10 μF capacitor will act as a filter for the **FREQOUT** command to create a sine wave.

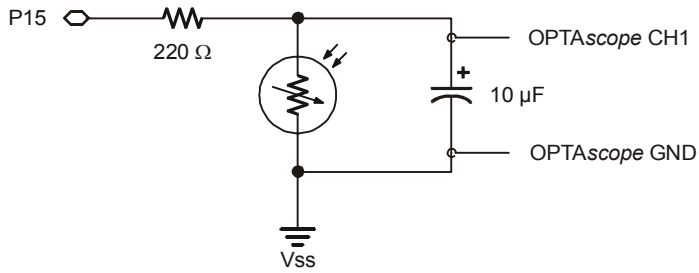


Figure 4-9:
RC Network with a photoresistor

Note: If you are using the BASIC Stamp HomeWork Board do not put the 220 Ω resistor in the circuit (this resistor is already surface mounted onto the PCB).

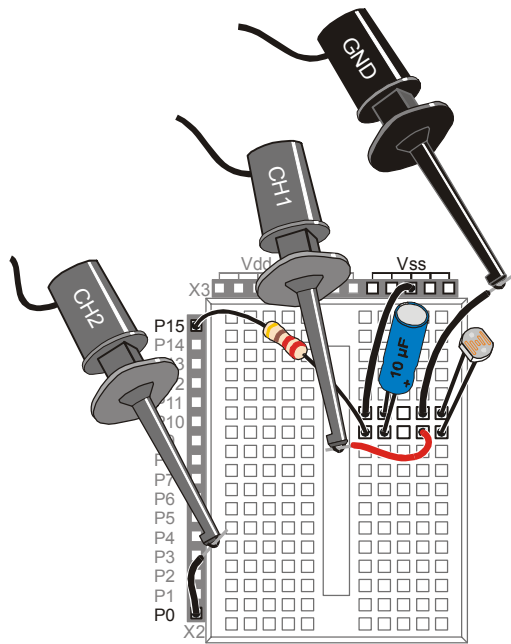


Figure 4-10:
Photoresistor RCTime wiring diagram with OPTAscope probes

Note: If you are using the BASIC Stamp HomeWork Board do not put the 220 Ω resistor in the circuit.

Instead, use a jumper wire to connect P15 to the capacitor.

Configuring the OPTAscope Software

- √ Configure the OPTAscope as shown in Figure 4-11.

CH1	2 V / division
CH2	Off
Horizontal Dial	10-20 ms / division
Trigger Source	Channel 1
Trigger Edge	Rising
Trigger Mode	Auto
Run / Stop Mode	Continuous
Trigger Voltage	2 V

Figure 4-11:
Configuration for
OPTAscope to
measure R/C
curves

4

RTimeConstantWithPhotoresistor.bs2

√ Run the program RTimeConstantWithPhotoresistor.bs2.

```
' Understanding Signals - RTimeConstantWith Photoresistor.bs2
' Read photoresistor in RC-time circuit using RCTIME command.
'{$STAMP BS2}
'{$PBASIC 2.5}

time    VAR          Word

DO

HIGH 0          ' 5 V on P0
HIGH 15         ' 5 V to capacitor for charge curve
PAUSE 10        ' time for capacitor to fully charge
RCTIME 15, 1, time
LOW 0
DEBUG HOME, "(time/1000)*2 = ", DEC3 (time/1000)*2
LOW 15

LOOP
```

√ Cover and uncover the photoresistor with your hand, and observe the discharge time extending or contracting. You may need to adjust the Horizontal dial to best display the waveform. The Debug Terminal will display the discharge time in milliseconds.



Converting RCTime units to milliseconds is done in the following PBASIC line of code:

```
DEBUG HOME, "(time/1000)*2 = ", DEC3 (time/1000)*2
```

The **RCTIME** command measures in units of 2 microseconds, but OPTAscope displays in units of milliseconds. The $(\text{time}/1000)*2$ instruction manages this conversion.



Why does my Debug Terminal display 000?

If you expose your photoresistor to very low light levels, the discharge time in this circuit may exceed 65535. This is the largest value that **RCTIME** can store in a word-size variable. Instead of allowing an overflow to occur, which would place a truncated and incorrect value in the **time** variable, the **RCTIME** command places a 0 in the time variable to let you know that a "timeout" occurred. This causes the Debug Terminal to display a 0, since $(0/1000)*2 = 0$. Simply re-run your program to clear the **time** variable, and keep the light level on your photoresistor a little brighter.

- √ Configure, resize, and reposition your PC display so the OPTAscope software is running side-by-side with the BASIC Stamp Debug display (Figure 4-12).
- √ Set the Cursor Settings switch to Paired Bars and place the red cursor at the top of the curve.
- √ Place the blue cursor at about 1.4 V.
- √ Compare your delta value measured with the OPTAscope to the data in the BASIC Stamp Debug Terminal. The values should be very close.

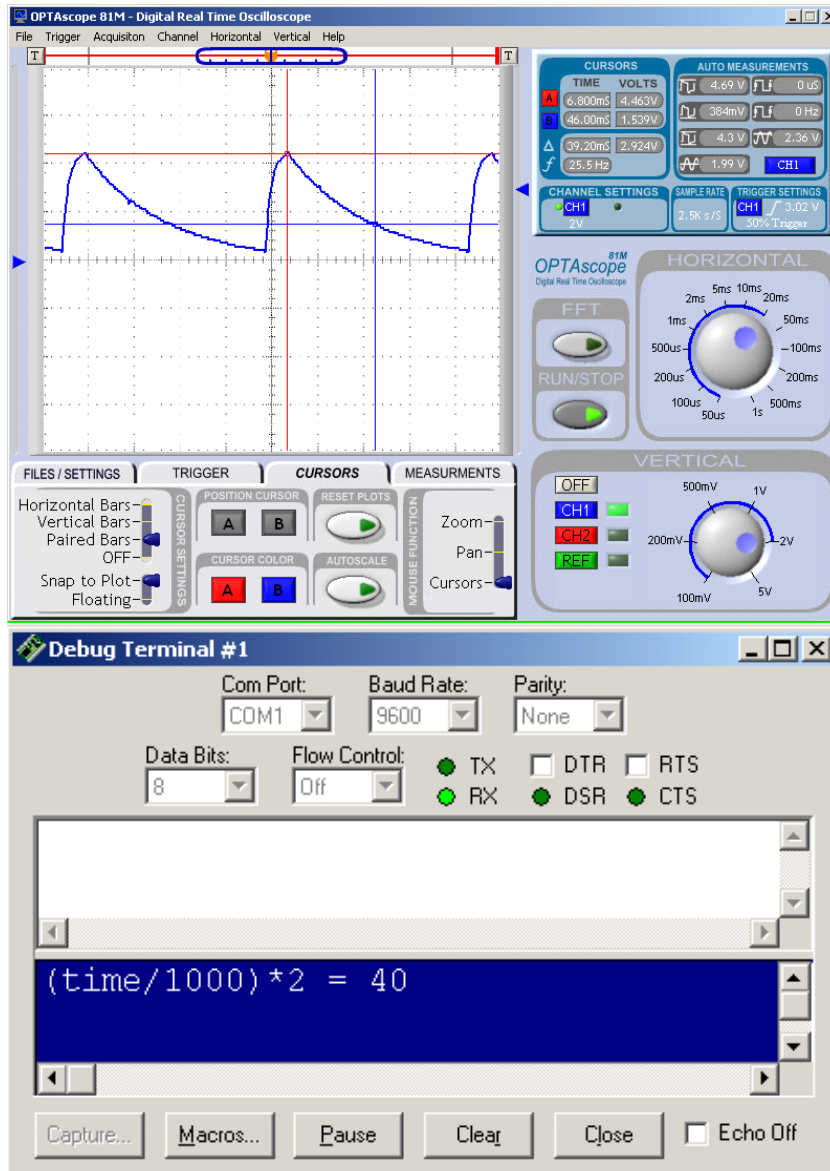


Figure 4-12: Using the Paired Bars cursors to measure capacitor discharge time

- √ Calculate the resistance of the photoresistor with the following formula:

Table 4-1: Known Values at the Instant VP15 Crosses the Threshold Voltage	
Values	Comments
Vdd = 5.0 volts	The initial condition of the voltage at P15.
VP15 = 1.4 volts	The final condition of the voltage at P15.
$e \approx 2.718$	The value of e is a constant found in many algebra books, physics texts, etc.
C = solve for this	If the value of resistance (R) is known, then the capacitance (C) can be determined.
t = known	The BASIC Stamp counts the time for us in 2 microsecond increments.
R = solve for this	If the value of capacitance (C) is known, then resistance (R) can be determined.

- √ You could also measure a capacitor and calculates its value with this formula.

$$R = \frac{t}{C \times \ln \frac{V_{dd}}{V_{P15}}}$$

Summary

This chapter demonstrated resistor/capacitor charge signals. It also showed how to use the OPTAscope to verify a calculated constant for an R/C network. Finally we demonstrated how to calculate a value of an unknown resistance, given the value of the capacitor and measuring the time constant.

Exercises

1. Build the circuit used in Activity 1, but replace the 10 μF capacitor with a 1 μF capacitor. Calculate the RC constant and verify your answer with the OPTAscope.
2. Replace the photoresistor with a potentiometer and calculate the resistance.
3. Set the thumbwheel on the potentiometer to a new position and calculate its resistance.

Further Investigation

“Applied Sensors”, Student Guide Version 1.3, Parallax, Inc. 2003.

Authored by Tracy Allen, PhD., this text provides very detailed discussions of the BASIC Stamp's RCTIME command with temperature probes and photodiodes. It is available online from the Stamps in Class Curriculum menu on the Education page at www.parallax.com.

Chapter 5: Synchronous Serial Communication

Data can be transmitted in either of two fashions: parallel or serial. Between the two, parallel data transmission is the fastest method, but it requires many I/O lines. Serial data transmission generally requires 1, 2, or 3 I/O lines. There are two prevalent modes of serial communications: synchronous and asynchronous. Synchronous is a word that means “with a clock”, whereas asynchronous is a word that means “without a clock”. This chapter introduces synchronous data communications.

Since many, many devices (i.e. chips) available today are accessed via a synchronous serial interface, it is essential to be able to “talk” to these devices with ease. In this chapter, a BASIC Stamp will be used to configure and read an A/D converter. With the help of the OPTAscope, these signals will be captured, studied, and understood. This level of understanding of synchronous data transfer will arm you with the knowledge and vocabularies you need to read a chip’s datasheet, and write a program to interface with it.

5

ACTIVITY #1: CAPTURING SYNCHRONOUS SERIAL COMMUNICATION

In this activity we will build an A/D converter circuit with an ADC0831 and capture the clock and data signals to see exactly how data is transmitted between the A/D converter and a BASIC Stamp. As the potentiometer’s tap knob is turned, you will see the 8-bit value sent from the ADC0831 change.

Parts Required

- (1) ADC0831 8-bit A/D converter
- (1) 10 k Ω potentiometer
- (14) Jumper wires



ADC0831 specifications and applications can be found on the datasheet, available online at <http://www.national.com/pf/AD/ADC0831.html>.

Building the Example ADC0831 Circuit

- √ Build the circuit shown in Figure 5-1 and Figure 5-2. It is the same for the Board of Education and HomeWork Board.
- √ Carefully connect the CH1 probe to pin 6 of the ADC0831, the CH2 probe to pin 7 of the ADC0831, and a Ground probe to Vss, as shown in Figure 5-2.

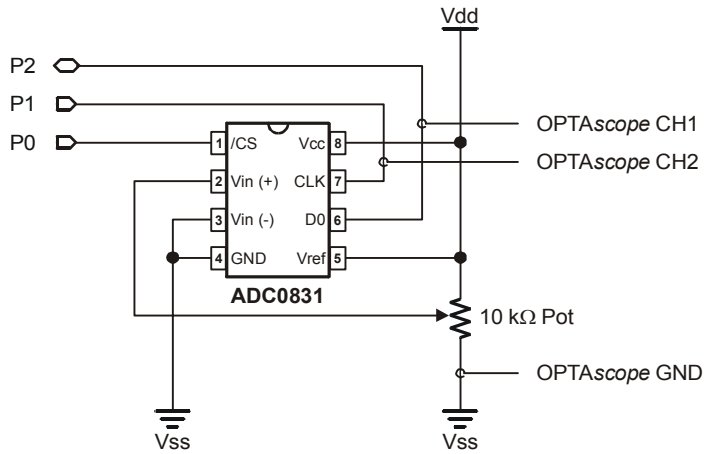


Figure 5-1:
ADC0831 circuit
schematic

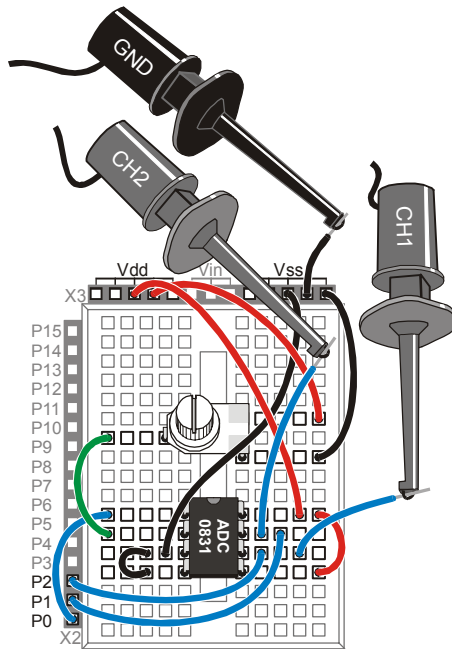


Figure 5-2:
ADC0831 wiring
diagram with
OPTAscope probes
installed

Configuring the OPTAscope Software

- √ Configure the OPTAscope as shown in Figure 5-3.

CH1	2 V / division
CH2	2 V / division
Horizontal Dial	200 μ s / division
Trigger Source	Channel 1
Trigger Edge	Rising
Trigger Mode	Normal
Run / Stop Mode	Continuous
Trigger Voltage	2 V

Figure 5-3:
Configuration for
OPTAscope to
capture
synchronous serial
data from the
ADC0831

5

Demonstrating Synchronous Serial with ShiftinA2DExample.bs2

- √ Run the program ShiftinA2DExample1.bs2.

```
' -----[ Title ]-----
' Understanding Signals - ShiftinA2DExample1.bs2
' {$STAMP BS2}
' {$PBASIC 2.5}

' -----[ Declarations ]-----

adcBits      VAR    Byte
v             VAR    Byte
r             VAR    Byte
v2           VAR    Byte
v3           VAR    Byte

' -----[ Initialization ]-----

CS           PIN    0
CLK          PIN    1
DataOutput   PIN    2

DEBUG CLS                                     'Start display.

' -----[ Main Routine ]-----

DO
  GOSUB ADC Data
  GOSUB Display
  PAUSE 100
LOOP
```

```
' -----[ Subroutines ]-----
ADC Data:
  HIGH CS
  LOW CS
  LOW CLK
  PULSOUT CLK, 210
  SHIFTFIN DataOutput,CLK,MSBPOST,[adcBits\8]
RETURN

Display:
  DEBUG HOME
  DEBUG "8-bit binary value: ", BIN8 adcBits
  DEBUG CR, CR, "Decimal Value: ", DEC3 adcBits
RETURN
```

- √ Use the red and blue arrows to separate the two signals in the display, as shown in Figure 5-5.
- √ Next, put your BASIC Stamp Debug Terminal side by side with your OPTAscope display to see the values you are receiving from the OPTAscope.
- √ Adjust the potentiometer tap by gently twisting the knob until the Debug Terminal Decimal Value reads 080. Your signals should now look like those in Figure 5-5.
- √ If you do not get a similar signal, make sure your Trigger Mode switch is set to Normal. Also, you may need to slide your Plot Area Indicator bar to the right to find the whole signal.

The blue signal at the bottom is the data signal. The data signal is what communicates a “1” or “0” to the device you are trying to talk to. A “1” is detected when the signal is above the TTL threshold, a “0” when the signal is below the TTL threshold.

The red signal on top is the clock. The clock tells the device you are talking to when to sample the data signal. This happens on the rising edge of the clock signal. At that instant in time the receiving device will look at the data line and latch in that value, a “1” or a “0”. That explains why this is synchronous serial communication; the master (the BASIC Stamp) and the slave (the ADC0831) agree to send and receive data according to the state of a second signal, the clock.



Clock Line vs. Data Line: The BASIC Stamp controls the clock line, but it is the ADC0831 that controls the data line

The lines in Figure 5-4 indicate the rising edge of the clock. Where that line meets the data signal is the value of the data signal that will be received by the BASIC Stamp.

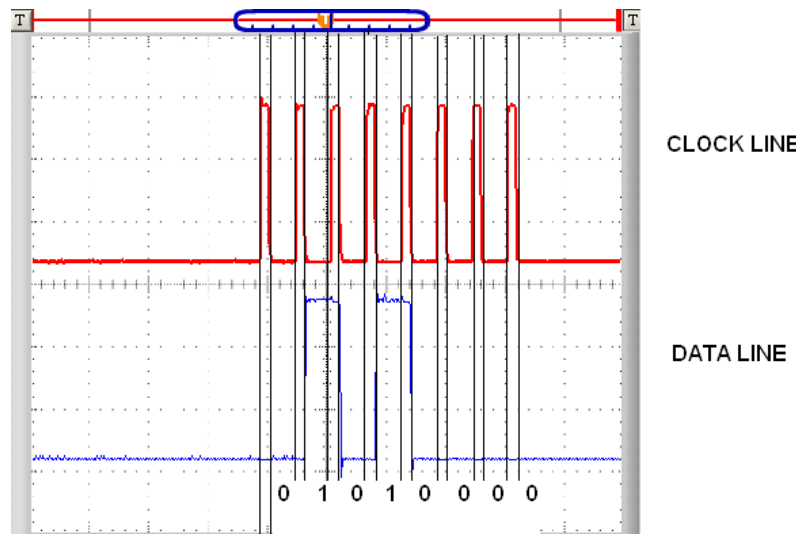


Figure 5-4:
Clock line
and data line

*Where the
clock line
meets the
data signal is
the value of
the data
signal that
will get
clocked into
the BASIC
Stamp*

5

Let's take a look at the BASIC Stamp command that generates this signal:

```
SHIFTIN DataOutput,CLK,MSBPOST,[adcBits\8]
```

The first two arguments set what pins the clock and data signals will be assigned to. The next sets the data format. **MSBPOST** specifies two things. First, the **MSB** portion means that the first value received will be placed in the most significant bit of the **adcBits** variable. The **POST** portion tells the BASIC Stamp to check for that value after the clock pulse. The next argument specifies the variable to receive the data.

- √ Compare the data signal from Channel 1 to the 8-bit binary value for number 80 as shown in the Debug Terminal. Can you recognize the binary number in signal form?
- √ Gently adjust the potentiometer tap knob to view other decimal values, and compare their binary forms to the data signal.

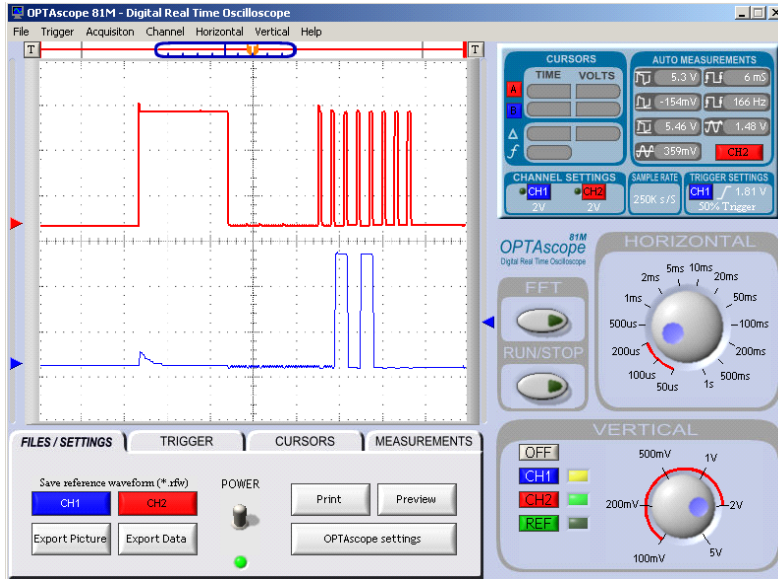
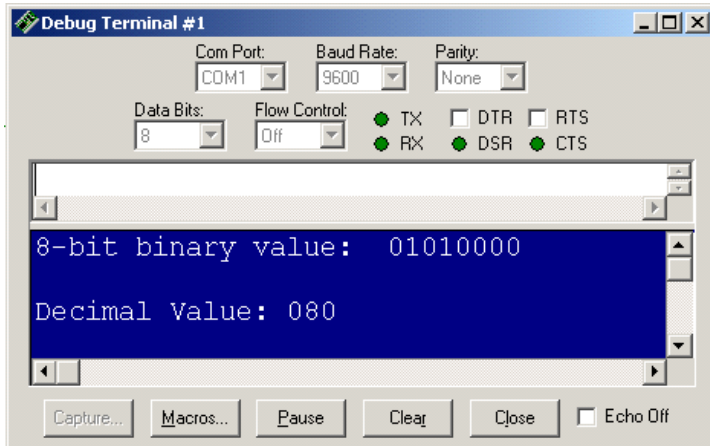


Figure 5-5: OPTAscope and Debug Terminal display binary 80

Above: note the clock line (red - top) and data from the ADC0831 (blue - bottom) which is binary 01010000, or decimal value of 80.

Below: The Debug Terminal displays each number in decimal and binary form.



Next, we will replace the clock line signal with the analog input voltage signal on Channel 2. There is only one addition to make to the circuit.

- √ Add a jumper wire to the breadboard between the ADC0831 Pin 2 and the potentiometer.
- √ Connect the CH2 probe to this new jumper wire.
- √ Click on the Measurements tab.
- √ Slowly and gently adjust the potentiometer tap to sweep through its whole range.

As you adjust the potentiometer, watch the numeric value of the Channel 2 data signal in the Debug Terminal, and the MEAN voltage in the CH2 Auto Measurements box. What relationship do you see?

5

Summary

This chapter examines the nature of the synchronous serial protocol used by the BASIC Stamp to communicate with the ADC0831. The clock signal, data signal and analog input were viewed. The activity demonstrated how to configure the OPTAscope with the Trigger Mode switched to Normal to capture signals, and made use of the Auto Measurements feature.

Exercises

1. Describe a way to estimate how long it takes for the BASIC Stamp **SHIFTIN** command to execute.
2. Look up the datasheet for the Analog Devices ADC0831 on the Internet, if possible. Does the chip have other features with different signals that you could verify with the OPTAscope?

Further Investigation

“Applied Sensors”, Student Guide, Version 1.3, Parallax, Inc. 2003

This text by Tracy Allen, PhD., has additional examples of serial communication using the DS1620 digital thermometer. It also has useful applications for signal demonstration due to the variety of communication between the DS1620 and a BASIC Stamp. It is available online from the Stamps in Class Curriculum menu on the Education page at www.parallax.com.

“Industrial Control”, Student Workbook, Version 1.1, Parallax, Inc. 2002

Authored by Martin Hebel and Will Devenport of Southern Illinois University, this text provides significant resources for the ADC0831. It is available online from the Stamps in Class Curriculum menu on the Education page at www.parallax.com.

Chapter 6: Asynchronous Serial Communication

The word asynchronous means “without a clock.” Asynchronous serial communication only uses one line, the data line, to communicate. The most commonly known type of communication that uses asynchronous communication is the RS-232 serial port on your computer. With RS-232 serial communication, the data line is used for synchronization and data. A consequence of this is that the receiver must always be watching the data line for a “start bit.”

When there is no data sent, the data line is said to be in its idle state. When the receiver sees that the data line has transitioned, it knows it’s time to initialize its timers and start receiving data. To send and receive data, both the transmitter and the receiver must set and read the data line at a very precise rate. This rate is called the baud rate. The baud rate must be the same in both the sender and the receiver before communication begins. The start bit lasts for one bit-time. After the start bit, each bit is sent, and each bit lasts one bit-time. If the bit to send is a 0, the data line is set high. If the bit to send is a 1, the data line is set low. See (Figure 6-1).

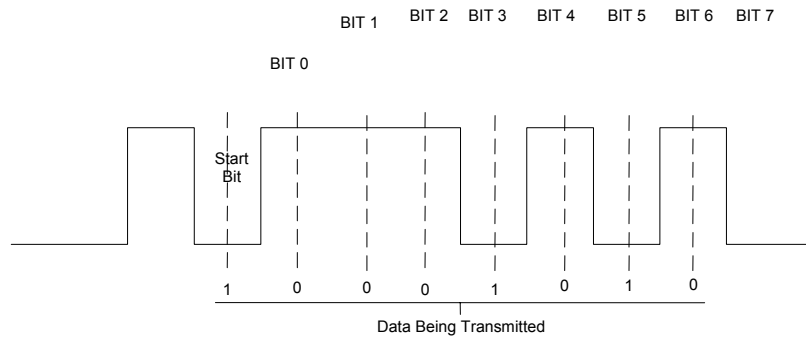


Figure 6-1:
Asynchronous
serial
communication
has no clock line

As you can see from the illustration above, the data bits are evenly spaced. Notice that the time from the rising edge of the start bit to the middle of data bit 0 is 1.5 times the bit time. When the receiver detects a start bit it will wait for 1.5 bit-times before reading in data bit 0. As a result, the receiver need only wait one bit-time to read the center of each subsequent bit. Reading the center of a bit is necessary to minimize errors.

Did you notice that when the transmitter is sending bit 0 it should be high or logic 1? This is because RS-232 is inverted. In a lot of cases it is also level shifted to -12 V for logic 1 and +12 V for logic 0. This helps transmit the data across long serial cables. Although, you can simply send a 5 V TTL signal from the BASIC Stamp and have good results with most computers. Since RS-232 signals come in both polarities, it's a good thing we have the OPTAscope to "see" this polarity rather than trying to guess it.

ACTIVITY #1: DISPLAYING 8-BIT INVERTED DATA

In this activity the BASIC Stamp will send the numbers 1 to 1,000 to the Debug Terminal and the OPTAscope will display them as they are received.

Required Parts

(2) Jumper wires

Building the Asynchronous Serial Circuit

This simple circuit is built the same way for the Board of Education and HomeWork Board.

- √ Build the circuit as shown in Figure 6-2 and Figure 6-3
- √ Connect the CH1 probe to P14 of the BASIC Stamp.
- √ Connect the Ground probe to the BASIC Stamp's Vss.



Figure 6-2:
High and low signals
circuit schematic

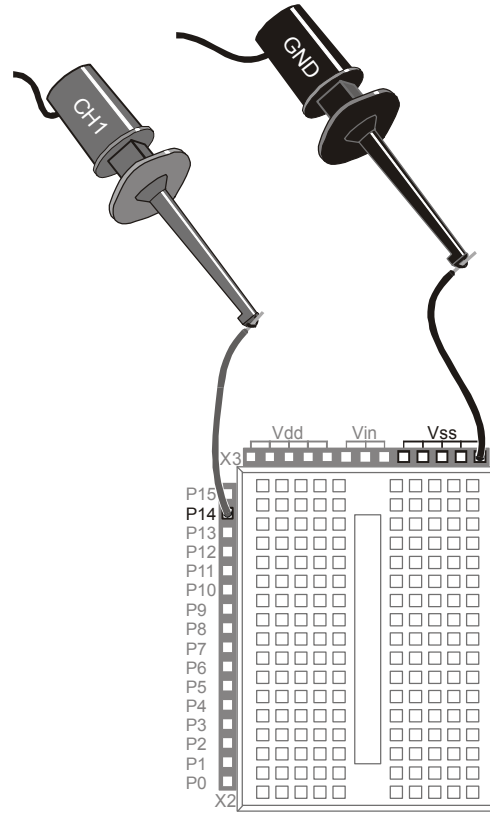


Figure 6-3:
High and low signals
wiring diagram

6

Configuring the OPTAscope Software

√ Configure the OPTAscope as shown in Figure 6-4.

CH1	2 V / division
CH2	Off
Horizontal Dial	500 μ s / division
Trigger Source	Channel 1
Trigger Edge	Rising
Trigger Mode	Normal
Run / Stop Mode	Continuous
Trigger Voltage	2 V

Figure 6-4:
Configuration for
OPTAscope to
capture
asynchronous serial
data

- √ Position the OPTAscope side-by-side with the BASIC Stamp's Debug Terminal to see the values in binary being sent by the BASIC Stamp at 9600 bps.

Running the AsynchSerial.bs2 Code

- √ Run the program AsynchSerial.bs2.

```
' Understanding Signals - AsynchSerial.bs2
' Send a single character to the DEBUG window

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

Value          VAR      Word

DO

  FOR Value = 1 TO 1000
    SEROUT 14, 16468, [Value]
    DEBUG HOME, CLS, "Decimal = ", DEC Value, TAB, "Binary = ", BIN8 Value
    PAUSE 1000
  NEXT

LOOP
```

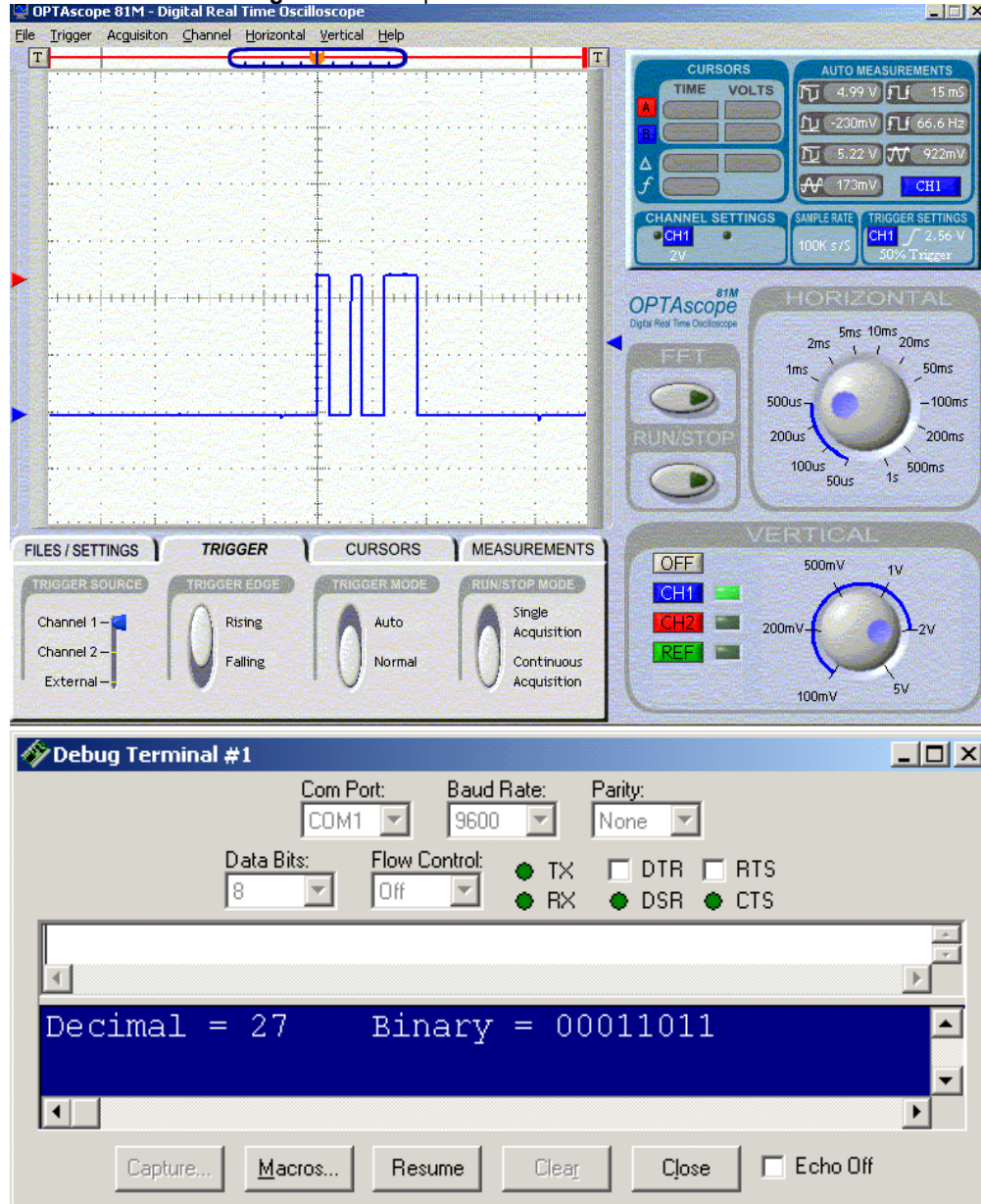
The OPTAscope will display a new waveform for each value incrementing in the Debug Terminal. What you are seeing is the binary signal of that digit sent as 8-bit no-parity inverted data at a baud rate of 9600, as determined by the **SEROUT** command's **Baudmode** argument **16486**. **Figure 6-5** captures the instance where the Debug Window and the OPTAscope are displaying number 27.

SEROUT *Tpin*, *Baudmode*, [*OutputData*]



The **SEROUT** command allows the BASIC Stamp to transmit asynchronous serial data, including RS-232 data. The **Tpin** argument specifies the BASIC Stamp I/O pin that will send the serial data. **Baudmode** is a code number that corresponds to a specific baud rate, bit number, parity and invert status. **OutputData** lists variables, constants, expressions and formatters that determine the format of the outgoing data. **SEROUT** also has other optional arguments not used in this program. For a complete description of **SEROUT**'s capabilities and tables of **Baudmode** codes, see the BASIC Stamp Manual or the Help file in your BASIC Stamp Editor 2.0.

Figure 6-5: Setup to view the 9600 inverted data



6

Now, let's change the baud rate and look at the resulting signal.

- √ Modify the `AsynchSerial.bs2` program's **SEROUT** command to read:

```
SEROUT 14, 16780, [Value]
```

- √ Run the modified program.

Now what is happening in the OPTAscope's Plot Area? Are you able to view the whole signal?

- √ Adjust the Horizontal dial until the signal appears to be similarly proportioned and viewable like the previous one.

What Horizontal dial setting did you find was necessary? Can you deduce the baud rate of this new signal? Compare the previous known baud rate and Horizontal dial setting to the new dial setting. If you set your dial to 200 ms and deduced a baud rate of 2400, you are correct. The **SEROUT Baudmode** argument **16780** produces an 8-bit no-parity inverted signal at a baud rate of 2400. In this way, the OPTAscope can be used to compare a known signal to an unfamiliar one for analysis.

However, the baud rate could have been determined directly from the signal with the cursors.

- √ Re-run the modified program.
- √ Carefully watch the Debug Terminal as it counts up, and press the Run/Stop button to capture the signal for number 27.
- √ Set the Cursor Settings switch to Vertical Bars.
- √ Set the Mouse Function switch to Zoom, and zoom in on the leftmost pulse in the signal.
- √ Use the cursors to measure the pulse width.

In the Display Area's Cursors box, look for the f measurement. You should see a number in the neighborhood of 2.4 kHz, corresponding to a baud rate of 2400 bps (bits per second).

ACTIVITY #2: DISPLAYING 8-BIT TRUE DATA

- √ Set the Horizontal dial back to 500 μ s.
- √ Modify the program AsynchSerial.bs2 to send true data by changing the **SEROUT** command's **Baudmode** argument to read:

```
SEROUT 14, 84, [Value]
```

- √ Run the modified program.

By changing the **Baudmode** argument, we can direct the BASIC Stamp to send normal data or inverted data. The **Baudmode** argument **84** specifies 8-bit no-parity true (non-inverted) data at a baud rate of 9600. Look at the code snippets below:

```
'Inverted Data
SEROUT 14, 16468, [Value] 'Idle state = 0,    logic 1 = 0,
                          'logic 0 = 1

'True Data
SEROUT 14, 84,    [Value] 'Idle state = 1,    logic 1 = 1,
                          'logic 0 = 0
```

Generally, when the signal level is RS-232, (roughly -12 to +12 VDC), inverted data is specified. When the signal level is TTL, (roughly, 0 to 5 VDC), true data polarity is specified. What complicates this issue is that RS-232 is normally inverted, so when we invert normal RS-232, we get normal data. Get it? Don't bother. Just be aware that RS-232 comes in two flavors, and it's good to have an OPTAscope to figure out which one you are dealing with.

Summary

This chapter examined asynchronous serial communication protocol. The **SEROUT** command was explained and used to generate inverted and non-inverted data signals, and signals at different baud rates. The OPTAscope's Horizontal dial, Zoom function, and Vertical Bars cursors were employed to analyze a signal and determine an unknown baud rate.

Exercises

1. What is the bit-time for one bit sent at a baud rate of 9600 bps?
2. Use the BASIC Stamp Editor's Help feature to look up the **SEROUT** command. Find the Baudmode argument to send 8-bit no-parity true data at a baud rate of 2400. Use the OPTAscope to view and verify that the signal sent is the inverse of that generated by the **Baudmode** argument 16780.
3. Using the OPTAscope, measure the duration between two characters sent with two **SEROUT** commands.

Further Investigation

“BASIC Stamp User's Manual” Version 2.0, Parallax, Inc., 2000

The **SERIN** and **SEROUT** Command Reference sections provide numerous examples that can be measured and observed with the OPTAscope. It is available online from the Stamps in Class Curriculum menu on the Education page at www.parallax.com.

Chapter 7: Pulse Width Modulation with Infrared

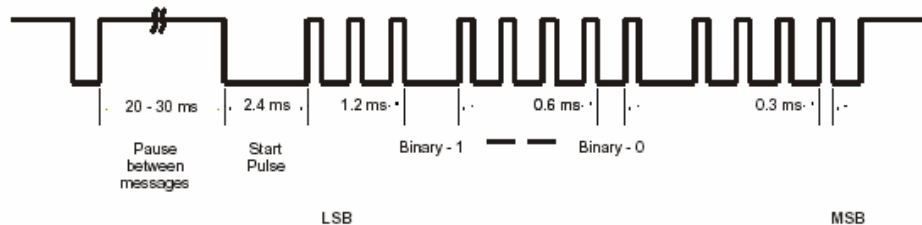
Have you ever wondered how the remote for your TV or VCR works? Infrared is what keeps you on the couch! When you press the power button on your remote control, a unique series of infrared energy bursts are emitted from the remote and radiate into the room. An infrared (IR) detector inside your TV decodes the signal and switches on your TV.

IR detectors are “tuned” for a specific frequency. The detector included the *Understanding Signals* kit is tuned for 38.5 kHz. The detector has a band pass filter that limits the input to 38.5 kHz only. This means that the IR detector will give an output only when a 38.5 kHz signal is received. The detector ignores all other inputs signals.

Data is transmitted by modulating the 38.5 kHz signal. This is done by varying the amount of time the 38.5 kHz signal is on and off. This works something like the asynchronous data signal that was covered in Chapter 6. Asynchronous communications use “high” and “low” signals at specific times to send data. The only difference in the IR protocol is that it uses a 38.5 kHz signal instead of the “high” signal. The times of the 38.5 kHz and the “low” are still controlled at specific times like the asynchronous signals we all know and love. The only other difference is that it is not the state that determines whether a bit is a one or a zero, but the duration of the bit determines its state.

The detector’s output is active-low. That means that the output is low when a 38.5 kHz IR signal is being received. Conversely, when there is no 40 kHz signal, the detector’s output is high. Figure 7-1 depicts the data bit stream from a Sony remote control. The receiver counts how long the signal is low. If it is 1.2 ms then that bit is a logic 1; if the signal is low for only 0.6 ms then it is a logic 0.

Figure 7-1: Infrared signal pulse example



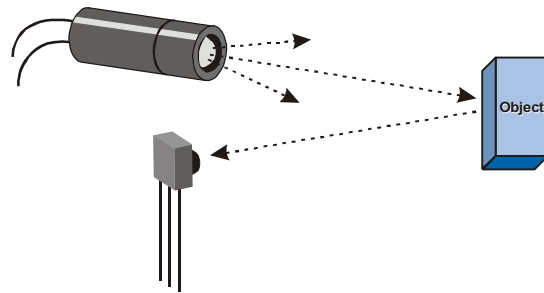


Figure 7-2:
Infrared for object
detection

Another use for IR is object detection (Figure 7-2). The IR beam reflects off objects just like light. When an IR beam is transmitted out, you can use the detector to look for an echo or reflection.

When you transmit a burst of IR light, if there is an object close enough the IR light will reflect that light back. The detector will see the light and pull its output pin low.

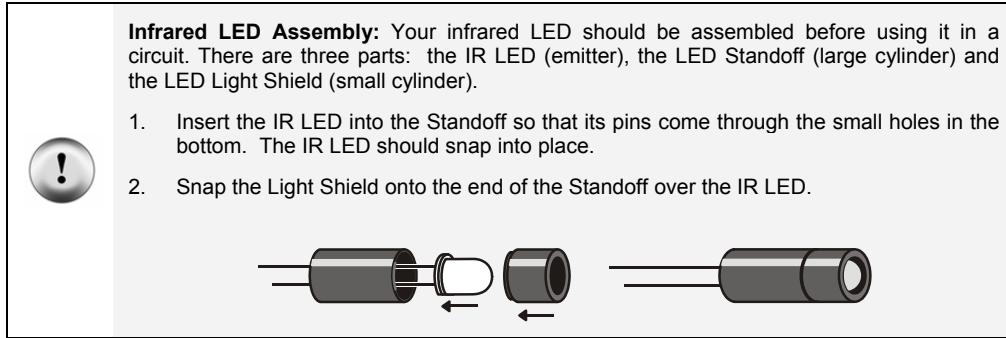
You can even tell how far away the object is by changing the sensitivity of the detector. By scanning at slightly different frequencies you can determine how far from an object is from the detector. For more information on distance detection see the article and application note references in the Further Investigation section at the end of this chapter.

ACTIVITY #1: INFRARED SIGNALS FOR OBJECT DETECTION

In this activity we will use the BASIC Stamp to generate the 38 kHz signal required to make the infrared detector's output go "low".

Required Parts

- (1) 220 Ω resistor
- (1) Infrared detector
- (1) Infrared LED
- (8) Jumper wires



Building the Infrared Object Detection Circuit

To make this circuit work properly, the infrared LED and detector should both point forward.

7

- ✓ Build the circuit shown in Figure 7-3 and Figure 7-4. If you are using a BASIC Stamp HomeWork Board omit the 220 Ω resistor and make the necessary connection with a jumper wire.
- ✓ Connect the CH1 probe to the jumper wire between the LED and resistor.
- ✓ Connect the CH2 probe to the jumper wire at the junction of P8 and the infrared receiver.
- ✓ Connect Ground probe to the jumper wire in Vss.

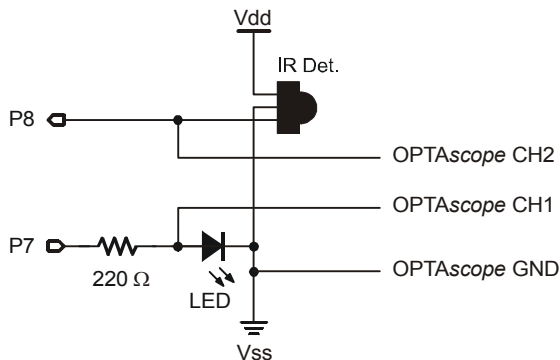


Figure 7-3:
Infrared object
detection circuit
schematic.

Note: If you are using the BASIC Stamp HomeWork Board leave the 220 Ω resistor out of the circuit – it is already installed between the I/O pin and the connection header.

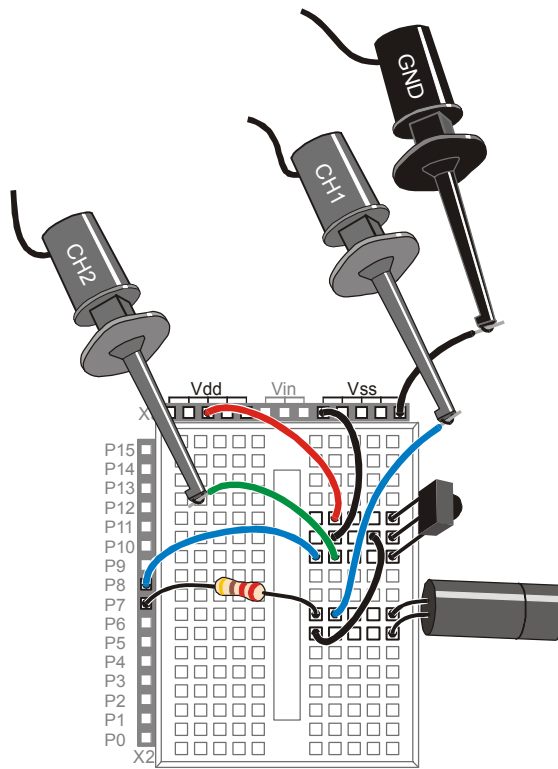


Figure 7-4:
Infrared object detection wiring diagram with OPTAscope probes installed

Note: if you are using the HomeWork Board, omit the 220 Ω resistor. Make the necessary connection with a jumper wire instead.

Configuring the OPTAscope Software

√ Configure your OPTAscope as shown in Figure 7-6.

CH1	1 V / division
CH2	2 V / division
Horizontal Dial	200 μ s / division
Trigger Source	Channel 2
Trigger Edge	Falling
Trigger Mode	Normal
Run / Stop Mode	Continuous
Trigger Voltage	2 V

Figure 7-5:
OPTAscope configuration for capturing infrared signals

Running 38kHzInfrared.bs2 Program

- √ Run the program 38kHzInfrared.bs2.

```
' Understanding Signals - 38kHzInfrared.bs2
' Send a single character to the Debug Terminal

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

DO

    FREQOUT 7,1,38500
    PAUSE 20

LOOP
```

- √ Separate the signals so that Channel 1 is in the lower half of the Plot Area and Channel 2 is in the upper half near the top.
- √ Slide the Plot Area Indicator bar one division to the right.
- √ Hold one hand in front of the IR LED and detector.
- √ Press the Run/Stop button with the other hand to freeze the signal.

7

You will see Channel 1 with a bit stream about 1ms long, and Channel 2 will have a low pulse for about the same time, except it will lag the **FREQOUT** signal. This lag occurs because it takes a little while for the IR detector to lock on to the signal, as you can see in Figure 7-6.

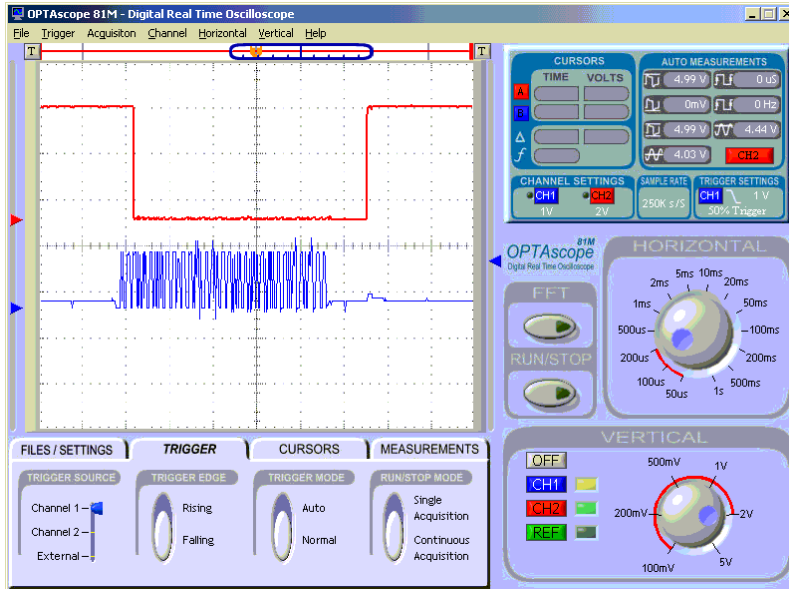


Figure 7-6: Infrared object detection

Object Detection with the 38kHzInfraredwithDetection.bs2 program.

Since the BASIC Stamp can't multitask, it cannot read the output of the detector *while* sending the **FREQOUT** signal. However, you can successfully read the detector's output immediately after you send the 40 kHz signal.

√ Run the program 38kHzInfraredwithDetect.bs2.

```
' Understanding Signals - 38kHzInfraredwithDetect.bs2
' 38 kHz infrared signal with detector feedback

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

IR_DETECT          VAR    Bit
LOW 7

START:

  PAUSE 20
  FREQOUT 7,1,38500
  IR_DETECT = IN8
```

```
IF IR_DETECT = 0 THEN DETECTED
  DEBUG HOME, "IR DETECTOR OUTPUT IS HIGH, NOT DETECTED"
GOTO START

DETECTED:
  DEBUG HOME, "IR DETECTOR OUTPUT IS LOW,      DETECTED"
GOTO START
```

If the IR signal makes it to the detector, the BASIC Stamp will report it to you in the Debug Terminal. This method can be used to detect objects. If, for whatever reason, the IR signal does not make it to the detector, the BASIC Stamp tells the Debug Terminal to display:

```
IR DETECTOR OUTPUT IS HIGH, NOT DETECTED
```

Now when you place your hand in front of the IR LED, you will see this in the Debug Terminal:

```
IR DETECTOR OUTPUT IS LOW,      DETECTED
```

The IR signal is reflecting off your hand and back to the IR detector. Note that when this message is visible in the Debug Terminal, the signals are moving in the Plot Area. When you remove your hand, the signals freeze. This is because your hand is not there to reflect the infrared light, so the Channel 2 signal is high and does not fall to the 2 V trigger level.

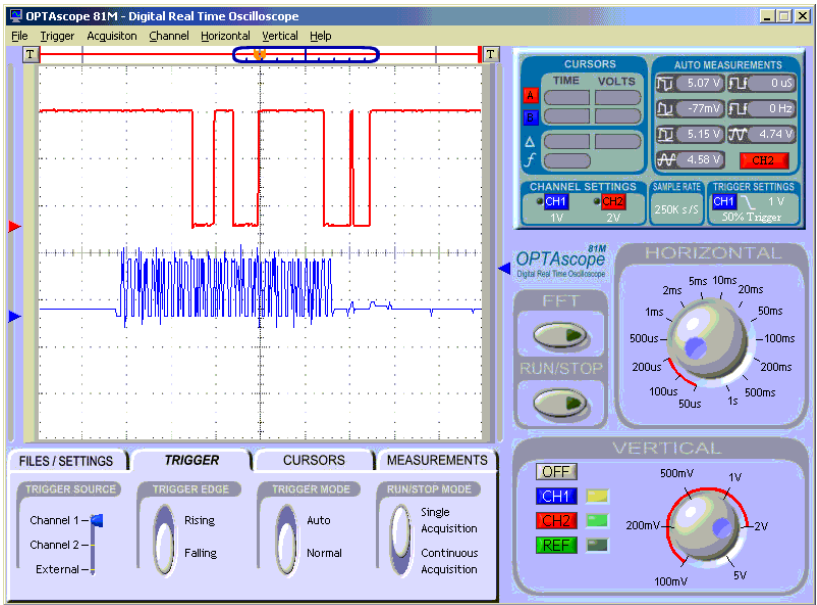


Figure 7-7: Hand waving causing sporadic detection by the receiver

If you move your hand up and down in front of the emitter and receiver, you may see a rapidly varying signal like the one shown in Figure 7-7. This happens when only a portion of the 38.5 kHz signal is being reflected off your hand.

Using the External Trigger with 38kHzInfraredwithVaryingFrequency.bs2

- √ Configure your OPTAscope with the following settings. Note: to set the external trigger at 10%, press the T button at the top left corner of the Plot Area, click the right T button to move it back.

CH1	1 V / division
CH2	2 V / division
Horizontal Dial	1 ms / division
Trigger Source	External at 10%
Trigger Edge	N/A
Trigger Mode	Normal
Run / Stop Mode	Continuous
Trigger Voltage	N/A
Autoscale	On

Figure 7-8:
OPTAscope
configuration for
capturing infrared
signals

7

- √ Point the IR LED and IR detector towards each other.
- √ Connect the External Trigger TTL probe to P0 using a jumper wire.
- √ Run the program 38kHzInfraredwithVaryingFrequency.bs2.

```
' Understanding Signals - 38kHzInfraredwithVaryingFrequency.bs2
' 38 kHz infrared signal with detector feedback

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

DO

HIGH 0
FREQOUT 7,3,38500
FREQOUT 7,1,38500
FREQOUT 7,1,38500
FREQOUT 7,2,38500
FREQOUT 7,2,38500
FREQOUT 7,1,38500
FREQOUT 7,1,38500
FREQOUT 7,1,38500
FREQOUT 7,2,38500
FREQOUT 7,1,38500
LOW 0
PAUSE 20

LOOP
```

In this example we are using the external trigger function of the OPTAscope 81M. This keeps the waveform on the screen and prevents it from jumping around, which is also called false triggering. You can see this by temporarily selecting the CH2 button as the trigger source.

If you were triggering on the IR detector you'd miss the picture because it would require a signal change (from high to low) to trigger.

√ Under the Cursors tab, turn on the Autoscale button.

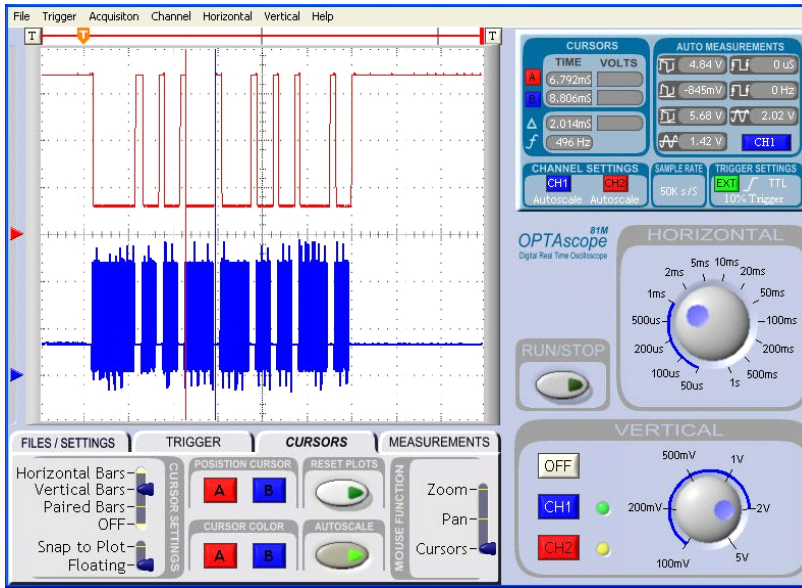


Figure 7-9: Using the external trigger feature

Turning on the Autoscale button will display all 1,500 data points of the signal in the Plot Area, as shown in Figure 7-9. Notice the Plot Area Indicator bar disappears. This gives the added benefit of better resolution at a 1ms time base, and you can still see the entire pulse train.

With the Autoscale button activated, the divisions or boxes on the screen no longer equal 1 ms per box. We have crunched the data so now each box is three times the selected time

base. In this case, the time per division is 3 ms. Normally there are 50 data points per division, now there are 150 per division.

As you can see from Figure 7-9, we have modulated the IR signal to the IR LED. This created a pulse train on the output of the IR detector.

- √ Under the Cursors tab, set the first Cursor Settings switch to Vertical Bars.
- √ Use the cursors to measure the pulse widths.

The first pulse is 3 ms; this can be used as a start bit. Every pulse after this start bit is one binary data bit. If the pulse is 1ms wide, it is a “logic 0”, if it’s 2ms long it’s a “logic 1”.

ACTIVITY #2: DECODING INFRARED REMOTE CONTROL SIGNALS

In this activity we will not use the infrared LED in the circuit, only the detector. The IR LED will in effect be disabled, because the program in this activity does not send voltage to pin 7 where it is connected. Instead, this Activity uses a handheld remote control as an infrared emitter.



Additional Required Parts

(1) Universal programmable infrared remote control, programmable for Sony TV’s.

Universal programmable infrared remote controls are widely available from major electronics and discount stores, and usually cost less than \$10.

Visualizing Handheld Remote Control Signals with DecodeSonyIRRemote.bs2

- √ Program your remote to be Sony compatible by following the manufacturer’s instructions.
- √ Point the IR LED away from the receiver so the receiver is unobstructed.

Configuring the OPTAscope Software

- √ Configure your OPTAscope as shown in Figure 7-10.
- √ Slide the Plot Area Indicator bar one division to the right.

CH1	Off
CH2	2 V / division
Horizontal Dial	2 ms / division
Trigger Source	Channel 2 at 50%
Trigger Edge	Falling
Trigger Mode	Normal
Run / Stop Mode	Continuous
Trigger Voltage	2 V
Autoscale	Off

Figure 7-10:
OPTAscope
configuration for
handheld infrared
remote control
decoding

Reading a Remote Control with DecodeSonyIRRemote.bs2

This application program is longer and more complex than the example programs used in the other Activities. The author has inserted some informational comments; for an in-depth look check out the Weekend Application Kit in the Further Investigation section at the end of this chapter.

√ Run the program DecodeSonyIRRemote.bs2

```
' -----[ Program Title and Description ]-----
' Understanding Signals - DecodeSonyIRRemote.bs2
' Decode 38 kHz Sony IR TV remote control signal.
' Author: Andy Lindsay, Parallax, Inc.
' {$STAMP BS2}
' {$PBASIC 2.5}
' -----[ I/O Definitions ]-----
IR detect      PIN    8           ' IR detector output -> P8.
' -----[ Constants ]-----
active_high    CON    1           ' Used to set PULSIN commands
active_low     CON    0           ' to detect +/- pulses.
' -----[ Variables ]-----
' This program reads all the pulses delivered by the remote, but in
' practice, only the first two to five pulses are required. This can be
' used To save seven To 9 Words in RAM (in this section) and the same
' number OF PULSIN commands in the Process IR Pulses subroutine.
```

```

IR pulse          VAR    Word(12)
counter           VAR    Nib
type              VAR    Nib
IR message        VAR    Byte

' -----[ Initialization ]-----
DEBUG CLS                    ' BOE reset clears display.

' -----[ Main Routine ]-----

DO

DO                        ' Wait for IR detector output
LOOP UNTIL IR Detect = 0  ' to go low.
GOSUB Display_Heading
GOSUB Find_and_Display_Start_Pulse
GOSUB Process_IR_Pulses
GOSUB Display_IR_Pulse_Values
GOSUB Convert_to_Binary_Number_Display

LOOP

' -----[ Subroutine - Display Heading in Debug Terminal ]-----

Display_Heading:

DEBUG HOME
DEBUG "IR Remote Messages ", CR, CR
DEBUG "Pulse Duration Value", CR
DEBUG "-----", CR

RETURN

' -----[ Subroutine - Find and Display Start Pulse ]-----

' Packets are delivered around 20 times/second while a given button on the
' remote is pressed and held. This program extracts a start pulse from
' an earlier packet. The Process IR Pulses subroutine picks up the rest
' of the pulse values a few packets later. In remote controlled
' applications, the duration of the start pulse can simply be discarded.

Find and Display Start Pulse:
FOR counter = 0 TO 15
PULSIN IR detect,active low,IR pulse(0)
IF IR pulse(0) > 900 THEN
DEBUG "Start"
DEBUG " = ", DEC5 IR_pulse(0) * 2, " us "
DEBUG " Start Bit", CR
EXIT                    ' Exit FOR...NEXT after start

```

```

ENDIF                                     ' pulse is detected.
NEXT
RETURN

' -----[ Subroutine - Process IR Pulses ]-----
Process_IR_Pulses:

DO
  PULSIN IR detect,active high,IR pulse(0)
  LOOP UNTIL (IR pulse(0) > 1400) AND (IR pulse(0) <> 0)

  ' The BASIC Stamp 2p and 2SX modules are fast enough to load these
  ' values using a FOR...NEXT loop, but all other modules should load the
  ' pulse values as a sequence of PULSIN Commands.

  PULSIN IR detect,active low,IR pulse(0)
  PULSIN IR_detect,active_low,IR_pulse(1)
  PULSIN IR_detect,active_low,IR_pulse(2)
  PULSIN IR detect,active low,IR pulse(3)
  PULSIN IR detect,active low,IR pulse(4)
  PULSIN IR detect,active low,IR pulse(5)
  PULSIN IR detect,active low,IR pulse(6)
  PULSIN IR_detect,active_low,IR_pulse(7)
  PULSIN IR_detect,active_low,IR_pulse(8)
  PULSIN IR detect,active low,IR pulse(9)
  PULSIN IR detect,active low,IR pulse(10)
  PULSIN IR detect,active low,IR pulse(11)

RETURN

' -----[ Subroutine - Display IR Pulse Values ]-----
Display IR Pulse Values:

FOR counter = 0 TO 10
  DEBUG " ", DEC2 counter
  DEBUG " = ", DEC5 IR pulse(counter) * 2, " us "
  IF IR pulse(counter) > 450 THEN
    DEBUG " Binary-1", CR
  ELSE
    DEBUG " Binary-0", CR
  ENDIF
NEXT

RETURN

' -----[ Subroutine - Convert to Binary Number Display ]-----
Convert to Binary Number Display:

```

```

FOR counter = 0 TO 10
  IF (IR pulse(counter) < 450) THEN
    IR message.LOWBIT(counter) = 0
  ELSE
    IR message.LOWBIT(counter) = 1
  ENDIF
NEXT

DEBUG CR,CR,"Binary Value: ", BIN8 IR message, CR
DEBUG "Decimal Value: ", DEC3 IR message, CR
DEBUG "Without bit-7: "
DEBUG " ",DEC3 IR_message & %01111111,CR

RETURN

```

√ Point the handheld infrared remote control at the infrared detector and press “5”.

Now you can see a similar pulse train generated from the Sony remote control protocol (Figure 7-11). As shown at the beginning of the chapter, the large pulse in the beginning is the start pulse. Any pulses that are 0.6 ms wide are logic 0s. The 1.2 ms pulses are logic 1s.

7

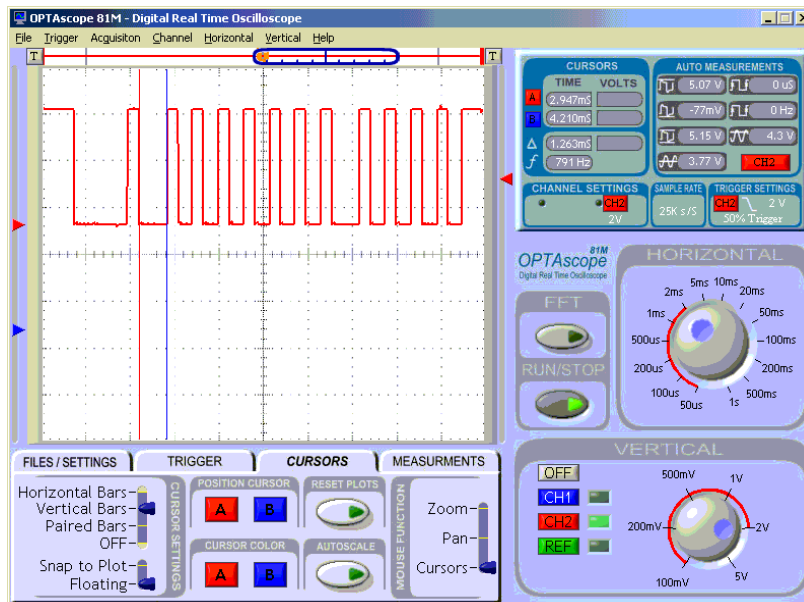


Figure 7-11:
Sony TV
remote key 5
pulse train

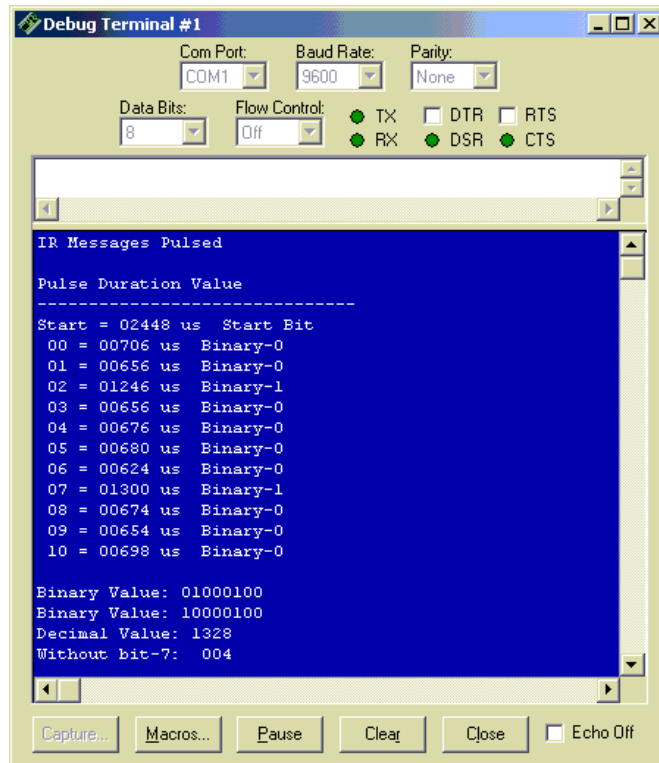


Figure 7-12:
Decoding
keypress 5 on
a Sony
remote

The pulse train generated by pressing 5 on the remote control is shown in Figure 7-11. You can see that the 10-bit wide data is transmitted LSB (least significant bit) first.

You should see the same data in your Debug Terminal (Figure 7-12). The program first looks for the start pulse then counts the 10 pulses for each bit. The value of each bit is listed.

From this we can demodulate the value of the data transmitted. The last four bits don't change so they are omitted. What we are left with is an 8-bit value of the demodulated signal. You could use this to instruct the BASIC Stamp to do different tasks by the remote control, or to send data between two BASIC Stamps.

Summary

This chapter demonstrated several uses for IR, including common circuits and methods used in robotics for object detection. The OPTAScope was triggered using an external trigger connected to a BASIC Stamp I/O pin. This ensured the signal would be captured at the proper time without relying on the receiver's pin. The Autoscale feature was introduced to automatically center the signal within the Plot Area. Additionally, the chapter provided code and a visual demonstration of decoding handheld infrared remote controls by demonstrating their waveforms.

Exercises

1. Describe how you could trigger the infrared object detection circuit without relying on the external trigger.
2. When experimenting with handheld infrared remote controls, press subsequent buttons and note how the signal changes. Describe the pattern of the binary results.

7

Further Investigation

"Infrared Emitting Diode & 40 kHz Infrared Detector" Stamp™ Weekend Application Kit Parallax, Inc., 2001

This downloadable Application Note includes detailed examples of infrared decoding and object detection. It is filed under Miscellaneous in the Accessory Docs menu on the Downloads page at www.parallax.com.

"Experiments from Optical Engineering and Robotics for a Pre-Engineering Program", Dr. S.K. Ramesh, Dr. Michael Fujita and Mr. Andrew Lindsay, of CSU Sacramento.

This article was presented at the IEEE Frontiers in Education Conference October 10-13, 2001 in Reno, Nevada. It provides examples of how these concepts can be utilized in engineering instruction. It is available from the Articles by Outside Authors menu on the Downloads page at www.parallax.com.

Chapter 8: Operational Amplifiers

Amplifiers are used in everything from car stereos to medical equipment. There are many types of amplifiers, each with different characteristics. This chapter will introduce you to simple signal conditioning with an operational amplifier, commonly called an op-amp (Figure 8-1). Op-amps have many, many potential functions in a wide variety of applications, but our introduction will be limited to a few common examples.

OP-AMPS AND THEIR USES AND LIMITATIONS

The op-amp is a very easy to use voltage amplifier. It accepts a voltage as an input and produces a mathematically related voltage on its output. Resistors can be used to configure the mathematical function that dictates the related output. By using a formula to select the input resistor (R_i) and the feedback resistor (R_f), you can set the gain of the op-amp. The gain is a numeric multiplier that refers to the relative difference between the voltage of the input signal and the voltage of the output signal. Therefore, once you know the gain of a given op-amp circuit, you can multiply the input voltage by the gain to get the output voltage. For example, an op-amp with a gain of 2 will have an output voltage twice the input voltage.

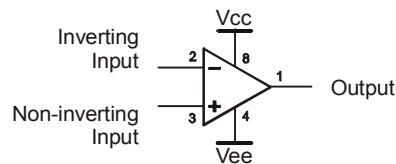


Figure 8-1:
Op-amp schematic symbol

There are limits to the op-amp that need to be considered before proceeding. The main limiting factors are:

- Supply voltage and output range
- Slew rate

Supply Voltage

Supply voltage and output range are related. An op-amp cannot create output signals any larger than the voltage applied at the V_{cc} and V_{ee} pins. This output voltage (or range) is called the dynamic range of the op-amp. Some op-amps can have a dynamic range as large as the span of the voltage supplied; they are called rail-to-rail op-amps. For

example, if $V_{cc} = 5\text{ V}$ and $V_{ee} = \text{GND}$, a rail-to-rail op-amp could create an output signal as large as 0 V to 5 Vs.

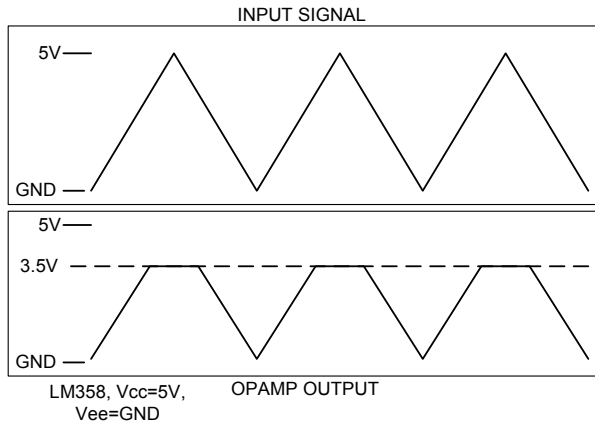


Figure 8-2:
LM358 Output ranges from 20 mV to 3.5 V

Note that the output signal is clipping at the LM358's maximum voltage of 3.5 V.

The LM358 found in the Understanding Signals kit is not a rail-to-rail op-amp. The LM358 output range is $V_{cc} - 1.5$ and $V_{ee} + 20\text{m V}$. With supply voltages of 5 V on V_{cc} and ground on V_{ee} , the output will only swing from 20m V to 3.5 V. Therefore, the output signal waveform can not exceed 3.5 V and the waveform's peaks appear to be flattened out. Such a signal is said to be clipped, as shown in Figure 8-2.



An LM358 circuit can be configured for an output swing larger than this in other applications by adjusting the V_{cc} and/or V_{ee} supply rails as needed. To check the maximum voltage you can apply to the LM358, check the datasheet, available online from <http://www.national.com/pf/LM/LM358.html>.

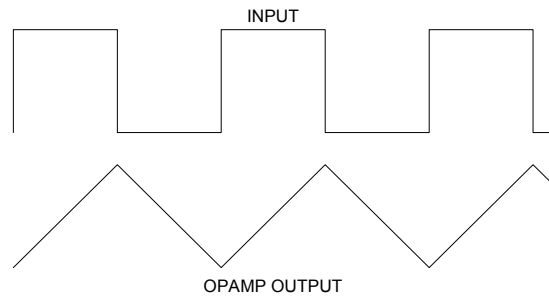
Slew Rate

Figure 8-3:
Example of op-amp output with a slow slew rate

The slew rate refers to how fast the op-amp can change its output voltage. The LM358 has a slew rate of 300 mV per microsecond. For every microsecond, the output can change as much as 300 mV. Bandwidth refers to the speed of the signal a device can process, so an op-amp is said to have high bandwidth if it has a high slew rate.

8

When the slew rate is too slow for the input signal, the op-amp will output a triangle wave (Figure 8-3). Notice the square wave on the input and the distorted triangle wave on the output. The op-amp cannot change its output as fast as the input signal. When the input suddenly changes from low to high, the op-amp can't keep up. As you can see this is a function of voltage and time. The larger the output voltage swing, the lower the bandwidth of the op-amp.

Op-amps can be used for a wide variety of applications. This chapter introduces the two most popular uses:

- Op-amps used as buffers
- Op-amps used as voltage amplifiers

AN OP-AMP USED AS A BUFFER

Sometimes, when connecting two circuits that were designed to each perform a specific function, the connection allows an interaction that makes the new combined circuit behave in unwanted ways. To prevent this unwanted interaction, a buffer circuit can be inserted between the two circuits, connecting them yet allowing them to operate as originally intended. Op-amps are commonly used to quickly and easily fashion buffers, which isolate and buffer circuits in a larger network. An example of an op-amp buffer circuit is shown in Figure 8-4.

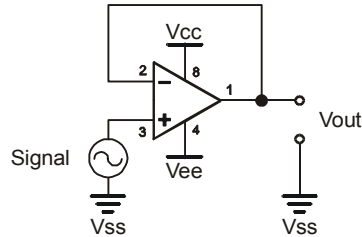


Figure 8-4:
An op-amp used
as a buffer

Functionally, a buffer circuit aims to generate an output that is identical to the input it receives. To use an op-amp as a buffer, make sure you have proper supply voltages (discussed below) and a fast enough slew rate. For all BASIC stamp applications, the LM358 has fast enough slew rate for the job.

For example, let's say we want to buffer a signal out of the BASIC Stamp that swings from 0 V to 5 V. To prevent the LM358 from clipping the output, V_{cc} should be supplied with a voltage that is 1.5 V higher than 5 V to prevent the signal from clipping. Also, the power supply may fluctuate, so to be on the safe side, increase V_{cc} by another 10-20% (depending on the precision of your power supply).

Likewise, the supply voltage at V_{ee} needs to be 20 mV below the lowest possible voltage value of the input signal. Since the input signal is expected to drop as low as 0 V, the V_{ee} supply must be at least -20 mV. Again, you may also want to adjust downward by another 10-20% in anticipation of power supply fluctuation.

AN OP-AMP USED AS A VOLTAGE AMPLIFIER

An op-amp can be used to amplify voltage. To do this, you have two types of circuits to choose from, inverting and non-inverting. As you can imagine, the inverting circuit will produce an output signal that is inverted, or the negative value of the input signal, creating a mirror image. Both inverting and non-inverting circuits use two resistors as a feedback loop to set the gain of the amplifier. However, their formulas and circuit characteristics are a little different, so let's look at each in turn.

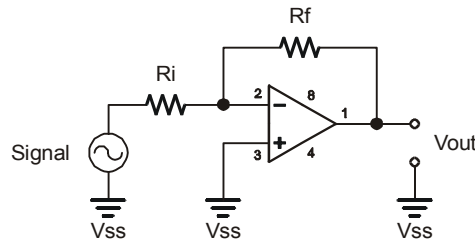


Figure 8-5:
Inverting voltage amplifier

8



Remember: R_i refers to the input resistor and R_f refers to the feedback resistor.

The inverting voltage amplifier shown in Figure 8-5 has the input signal connected to R_i . R_i is then connected to R_f and the inverting input of the op-amp. In this circuit the connection at R_i , R_f and the inverting input is called virtual ground. This means that each R_i equals the approximate input impedance. Therefore, the inverting circuit doesn't allow high input impedance. However, it does allow a gain of less than one, allowing you to take a large signal and scale it down to a smaller signal. This is called attenuation.

The formula to calculate gain is as follows:

$$\text{Gain} = R_f/R_i$$

For example, if R_i is a 10 k Ω resistor and you know you want a gain of 2, your calculation to determine R_f would look like this:

$$R_f = R_i * \text{Gain}, R_f = 10 \text{ k} * 2, R_f = 20 \text{ k}$$

Therefore, a 20 kΩ resistor is needed for R_f, given a 10 kΩ resistor for R_i and a desired gain of 2. Remember, the 10 kΩ resistor in the input impedance of the op-amp.

Two 10 kΩ resistors would present themselves as an additional load (Figure 8-6). The two 10 kΩ resistors when measured at the middle should yield 2.5 V. When you connect your op-amp circuit to a 2.5 V signal, you might see something less than 2.5 V. The output impedance of the resistor divider is high, meaning it does not supply very much current. The inverting op-amp circuit has low input impedance, hence the 10 kΩ input resistor. So, the op-amp circuit will load down the 2.5 V signal, causing it to sag.

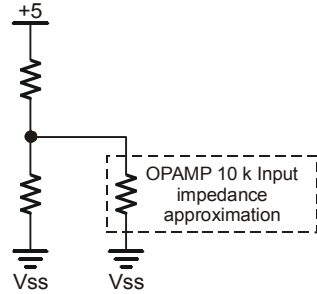


Figure 8-6:
Two 10k resistors have significant output impedance.

When using inverting op-amp circuits, keep what you have just learned in mind. You can increase the value of R_i in the op-amp circuit to increase the input impedance, but if you can, keep the value around 10 kΩ.

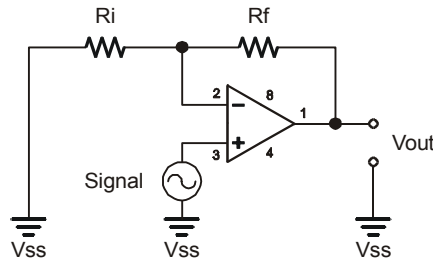


Figure 8-7:
Non-inverting Voltage Amplifier

A non-inverting voltage amplifier has its input signal applied directly to the non-inverting input of the op-amp (Figure 8-7). This gives you very high input impedance, up to 2 MΩ

depending on the op-amp used. With this configuration, you cannot attenuate the signal like you could with the inverting amplifier; it can only be used to amplify. Let's look at the formula to calculate gain for this configuration:

$$\text{Gain} = 1 + R_f/R_i$$

Using a 1 k Ω resistor for both R_f and R_i would give a gain of 2. These formulas should be good for a close approximation, but some "fine tuning" may be necessary depending on what op-amp you use.

ACTIVITY #1: SINE WAVE THROUGH A NON-INVERTING AMPLIFIER USING AN LM358 OP-AMP

In this activity, you will use a non-inverting op-amp circuit to amplify a sine wave that is generated by the BASIC Stamp using the **FREQOUT** command. You will be trying several different values for R_i and R_f to see how the gain determines how much to amplify the signal. You will also examine signal clipping, which occurs when the circuit tries to amplify the signal beyond the LM358's dynamic range.

8

Parts Required

- (2) 1 k Ω resistors
- (1) 2 k Ω resistor
- (1) LM358 op-amp
- (1) 1 μ F Capacitor
- (1) 220 Ω resistor
- (9) Jumper wires

Building and Understanding the Non-Inverting Amplifier Circuit

The schematic (Figure 8-8) and wiring diagram (Figure 8-9) for this circuit are shown below. As mentioned earlier, the resistors labeled R_i and R_f in the circuit are the input and feedback resistors. The gain of the amplifier depends on the ratio of the values selected for each of these resistors.

- √ Set aside a 2 k Ω resistor to use in place of R_i and a 1 k Ω resistor to use in place of R_f .
- √ Build the non-inverting amplifier circuit shown in Figure 8-8 and Figure 8-9, making sure to use a 2 k Ω resistor for R_i and a 1 k Ω resistor for R_f .

- ✓ Calibrate your OPTAscope 81M by selecting Calibrate under the File pull down menu and following the instructions.
- ✓ Connect the CH1 and CH2 probes as shown in Figure 8-9.



WARNING: The 1.0 μF capacitor can explode if it is connected improperly!

Always disconnect the power before building or modifying circuits. Carefully check the polarity of the 1.0 μF capacitor. The positive terminal (longer leg) must be connected to a power source pin and the negative terminal (shorter leg) must be connected to Vss (ground).

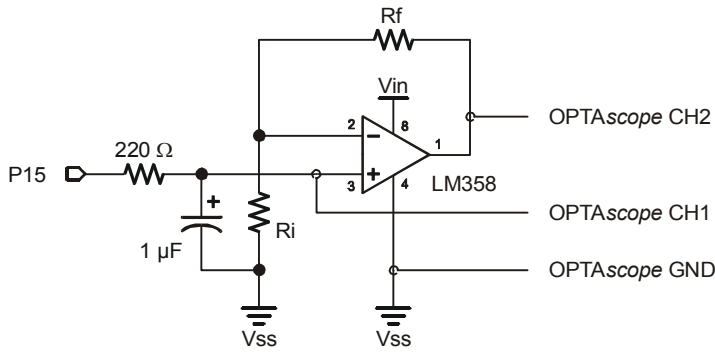


Figure 8-8:
Non-inverting op-amp circuit schematic, with Ri and Rf labels

Note: If using the HomeWork Board, leave out the 220 Ω resistor. It is already built into the board.

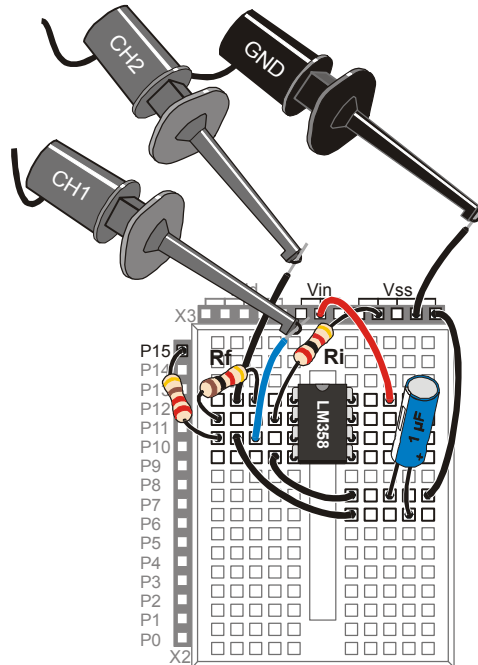


Figure 8-9:
Non-inverting op-amp circuit wiring diagram, with Ri and Rf labels

Note: If using the HomeWork Board, replace the 220 Ω resistor with a jumper wire. A 220 Ω resistor is already built into the board.

8

Recall that the gain for a non inverting amplifier is:

$$\text{Gain} = 1 + R_f/R_i$$

If Ri is 2 kΩ and Rf is 1 kΩ, the gain will be:

$$\begin{aligned} \text{Gain} &= 1 + 1000/2000 \\ &= 1 + 0.5 \\ &= 1.5 \end{aligned}$$

This means that the amplitude of the signal at the amplifier's output (measured by OPTAscope CH2) will be 1.5 times the amplitude of the signal supplied to the amplifier's input (OPTAscope CH1). If Ri is 1 kΩ and Rf is also 1 kΩ, the gain will be:

$$\text{Gain} = 1 + 1000/1000$$

$$= 1 + 1$$

$$= 2$$

In this case, the output signal should be twice the amplitude of the input signal. If $R_i = 1 \text{ k}\Omega$ and $R_f = 2 \text{ k}\Omega$, the gain will be:

$$\text{Gain} = 1 + 2000/1000$$

$$= 1 + 2$$

$$= 3$$

In this case, the output signal should be three times the amplitude of the input signal.

Configuring the OPTAscope Software

√ Configure the OPTAscope as shown Figure 8-10.

CH1	2 V / division
CH2	2 V / division
Horizontal Dial	500 μs / division
Trigger Source	Channel 1
Trigger Edge	Rising
Trigger Mode	Auto
Run / Stop Mode	Continuous
Trigger Voltage	2 V

Figure 8-10:
Configuration for
viewing non-
inverting amplifier

OPAmExamplewithFREQOUT.bs2

OPAmExamplewithFREQOUT.bs2 can be used to supply a 1 kHz sine wave signal to the amplifier's input. Whatever the amplitude of this input is, the gain will try to make the output 1.5 times greater.

√ Run the program OPAmpExamplewithFREQOUT.bs2

```
' Understanding Signals - OPAmExamplewithFREQOUT.bs2
' Generate a sine wave for the op-amp

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

DO

    FREQOUT 15, 1000, 1000

LOOP
```

Keep in mind that the amplifier gain for $R_i = 1 \text{ k}\Omega$ and $R_f = 2 \text{ k}\Omega$ should be about 1.5. The best way to measure this is with the Horizontal Bars cursors. Figure 8-11 shows the Horizontal Bars cursors measuring the amplitude of the input signal on OPTAscope CH1.

- √ Arrange the signals in the Plot Area so that their lowest peaks align as shown in Figure 8-11.
- √ Set the Cursor Settings switches to Horizontal Bars and Snap to Plot.
- √ Measure the amplitude of the input signal and make a note of the ΔV in the Cursors box.
- √ Repeat the measurement for the output signal, again noting the ΔV .
- √ Divide the ΔV measurement for the output by the ΔV measurement for the input. This will give you the signal gain, and it should be fairly close to 1.5.

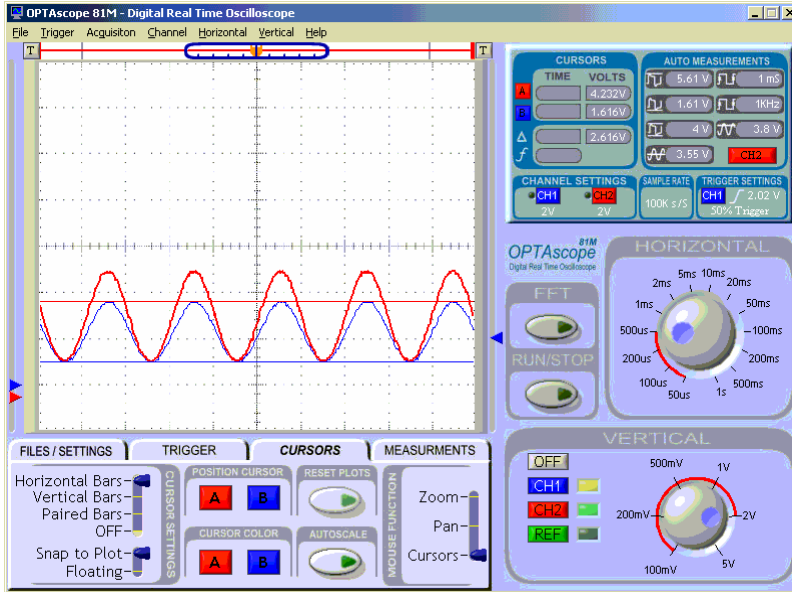


Figure 8-11: Input and output signals for a non-inverting amplifier with a gain of 2

You will repeat this exercise using different resistor values for R_i and R_f to create a gain of 2 and a gain of 3. When you repeat this exercise for a gain of 2, where $R_i = 1\text{ k}\Omega$ and $R_f = 1\text{ k}\Omega$, your signal may or may not be clipped. This will depend on your supply voltage, which determines the dynamic range of the amplifier. Whenever the op-amp tries to send a voltage that is beyond its dynamic range, the actual output will stop at the limit of its dynamic range. If you are using a fresh 9 V battery, your signal might not be clipped, but if it is close to worn out, your signal might be clipped. Figure 8-12 shows what the signal with a gain of 2 might look like if it is clipped or not clipped.

- ✓ Repeat this exercise using a $1\text{ k}\Omega$ resistor for R_i and a $1\text{ k}\Omega$ resistor for R_f in your circuit.
- ✓ Set your Cursor Settings switches to Horizontal Bars and Floating.
- ✓ Measure the amplitude of the new output signal, noting the ΔV value. If your signal is clipped, you may have to visually estimate the placement of the upper cursor as shown in the bottom picture of Figure 8-12.

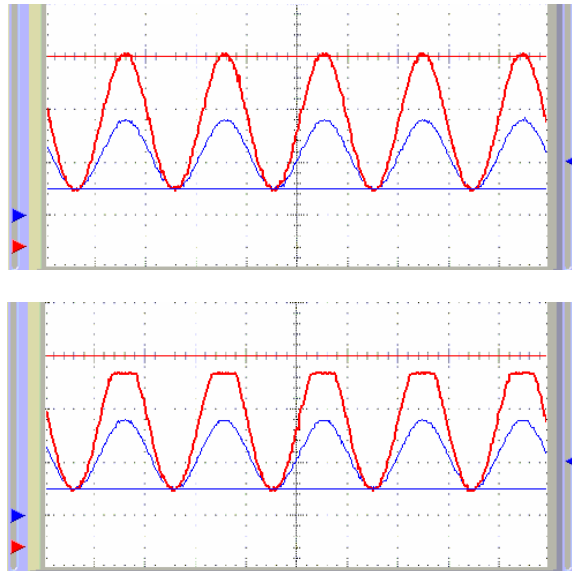


Figure 8-12:
Op-am input and
output signals

*Above: Output
signal with a non-
inverting amplifier
gain of 2 (not
clipped) using a
fully-charged 9 V
battery.*

*Below: Same
output signal
clipped because a
9 V battery was
used that is near
the end of its
useful life.*

8

When you up the gain to 3 ($R_i = 1\text{ k}\Omega$ and $R_f = 2\text{ k}\Omega$) your signal will most likely be heavily clipped, as shown in Figure 8-13. We won't calculate the gain, but the signal is worth examining anyway since it illustrates a common phenomenon in amplifier design. Clipping happens when the gain tries to make an output signal that is outside the dynamic range the amplifier. This type of distortion is undesirable if the goal is accurate sound reproduction, for example. However, this type of signal distortion is frequently found in rock music.

- √ Again repeat this exercise, using a $1\text{ k}\Omega$ resistor for R_i and a $2\text{ k}\Omega$ resistor for R_f .
- √ Observe your heavily clipped signal, as shown in Figure 8-13.
- √ Set the Horizontal Bars cursors at the top and bottom of the clipped signal.
- √ Under the Measurements tab, look for the MAX measurement in the CH2 Auto Measurements box.

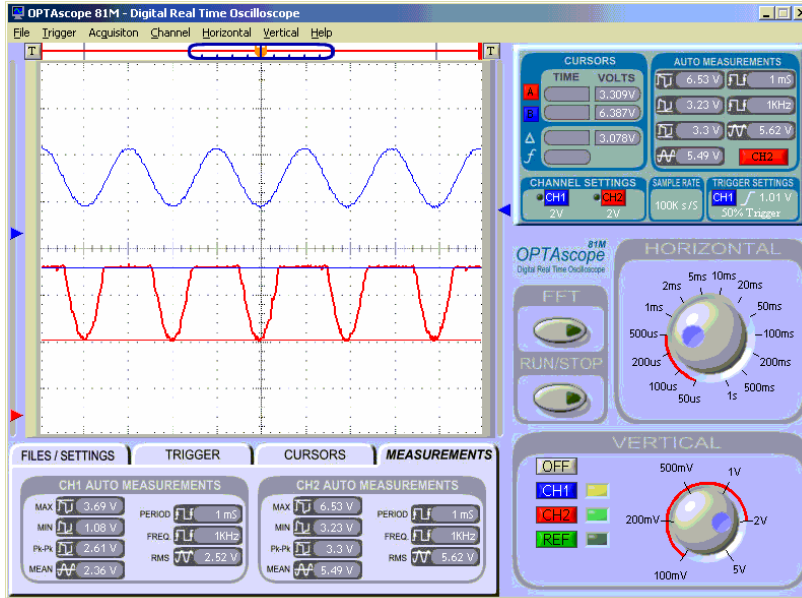


Figure 8-13: Output signal with a gain of 3, heavily clipped

Note that the MAX voltage measurement shows 6.53 V, evidence of a weakening 9 V battery.

How healthy is your battery? We measured the clipped signals for 3 different 9 V batteries, and had readings of 8.01 V, 7.29 V and a low of 6.53 V, shown in Figure 8-13.

ACTIVITY #2: INVERTING AMPLIFIER WITH ADJUSTABLE DC OFFSET

The op-amp circuit in this Activity performs two functions at once. The first function is that of an inverting amplifier; the second supplies a DC offset to the output signal.

Recalling the formula Chapter 4, using a 10 k Ω resistor as R_i and a 20 k Ω resistor as R_f will cause a gain of 2. Recall also that in the case of an inverting amplifier, the voltage output will be *negative* 2 times the voltage input. Therefore, an input signal that swings from 1 to 3 V put through an inverting amplifier with a gain of 2 would output a signal that swings from -2 to -6 V. Further, recall that an op-amp cannot output a signal with a voltage that exceeds the voltages applied to the supply rails (V_{cc} and V_{ee}). Notice that in the schematic (Figure 8-14) that V_{ee} is connected to 0 V. Since the whole range of the output signal is below 0 V, the entire signal would be clipped, and would only be visible on the OPTAscope as a 20 mV flat line!



Negative voltages can be viewed by applying a negative 9 V to Vee with a separate power supply. This is most easily done with a second 9 V battery. Disconnect the op-amp's Vee from the Vss on your Board of Education or HomeWork Board. Connect the battery's positive terminal to Vss on your Board of Education or HomeWork Board. Then, connect the battery's negative terminal to Vee.

In order to view this signal without a negative supply rail voltage, we can add DC voltage to move the entire output waveform above 0 V so it will no longer be clipped. This process is called DC offset. This is accomplished with the 10 k Ω potentiometer connected to the non-inverting input. This means that the op-amp circuit adds a DC voltage to whatever signal is supplied to the circuit's inverting input. By twisting the potentiometer's knob, the output signal will move into the viewable voltage range.

Parts Required

- (1) 20 k Ω resistor
- (1) 220 Ω resistor
- (1) 10 k Ω resistor
- (1) 1.0 μ F capacitor
- (1) 10 k Ω potentiometer
- (1) LM358 op-amp
- (9) Jumper Wires

8



WARNING: The 1.0 μ F capacitor can explode if it is connected improperly!

Always disconnect the power before building or modifying circuits. Carefully check the polarity of the 1.0 μ F capacitor. The positive terminal (longer leg) must be connected to a power source pin and the negative terminal (shorter leg) must be connected to Vss (ground).

Building the Variable Resistor Op-Amp Circuit

- √ Build the circuit shown in Figure 8-14 and Figure 8-15. If you are using the HomeWork Board, omit the 220 Ω resistor and make the necessary connection with a jumper wire.

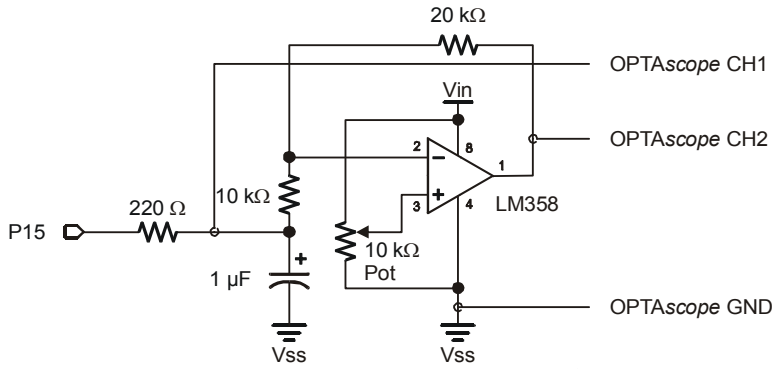


Figure 8-14:
Op-amp with
potentiometer

*Note: If using
the HomeWork
Board, leave
out the 220 Ω
resistor. It is
already built
into the board.*

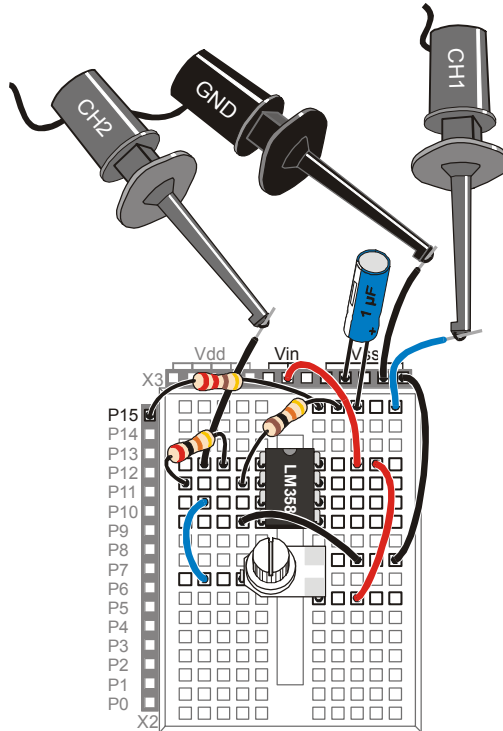


Figure 8-15:
Photoresistor
RC-Time circuit
wiring diagram
with
OPTAscope
probes

*Note: If using
the HomeWork
Board, replace
the 220 Ω
resistor with a
jumper wire. A
220 Ω resistor
is already built
into the board.*

- √ Once the circuit is built, connect the probes of the OPTAscope as shown Figure 8-15.

- ✓ Re-run the program OPAmplifierExamplewithFREQOUT.bs2.
- ✓ Adjust the tap on the potentiometer by gently twisting the knob until the entire signal is visible and no longer clipped, as in Figure 8-16 .

Notice that output signal is a mirror image of the input signal (inverted), except that the output signal has twice the amplitude (a gain of 2).

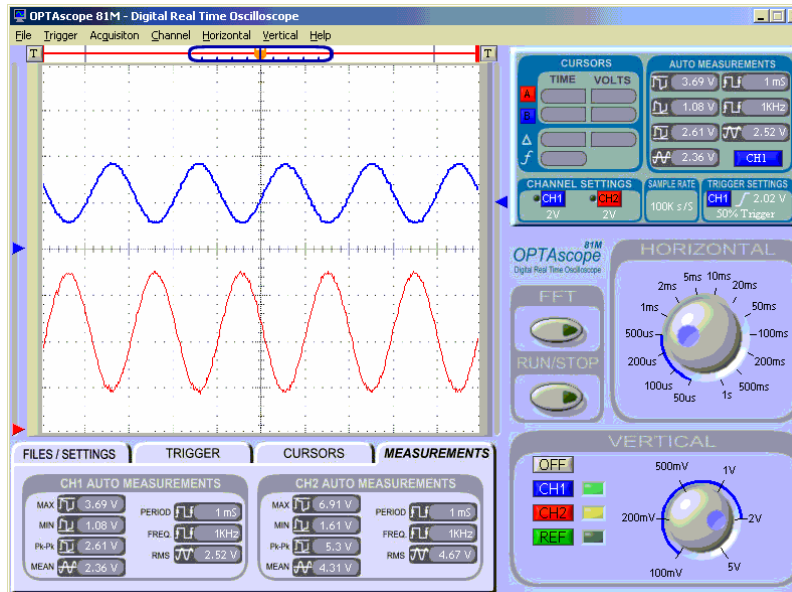


Figure 8-16:
Potentiometer
input sine wave

8

- ✓ Under the Cursors tab, set the Cursor Settings switches to Horizontal Bars and Snap to Plot.
- ✓ Measure the voltage difference between the high and low peaks of the input signal.
- ✓ Repeat this measurement with the output signal, and then compare the difference in amplitude.

Is the actual gain close to 2? You can also click on the Measurements tab to view and compare the statistics for both signals.

Summary

This chapter introduced just a few basic examples of the many uses of operational amplifiers: buffers, and inverting and non-inverting voltage amplifiers. The concepts of gain, slew rate, and signal clipping were introduced. A variable resistor was used to accomplish the DC offset necessary to view a signal whose output was below the range determined by the op-amp's power supply rails.

Exercises

1. Design and implement a non-inverting amplifier with a gain of 4.
 - a. What is the maximum voltage swing you can have without clipping the output?
 - b. What *freq1* argument would you use with the **FREQOUT** command so the RC circuit attenuates the input signal to prevent clipping the output? Explain why adjusting the frequency affects the amplitude of the RC network.
2. Design and implement an inverting amplifier with a gain of 0.25.

Further Investigation

“Basic Analog and Digital”, Student Guide, Version 1.2, Parallax, Inc. 2003.

This Stamps in Class text by Andy Lindsay of Parallax provides additional exercises using the LM358 op-amp. It is available online from the Stamps in Class Curriculum menu on the Education page at www.parallax.com.

Appendix A: System and Equipment Requirements

SYSTEM REQUIREMENTS

The software operating the OPTAscope must run on an IBM style PC. Additionally, the following list of minimum requirements must be met:

- √ Pentium (or equivalent) running @ 233MHz
- √ Microsoft Windows 98, 98SE, ME, 2K, or XP
- √ 64MB of RAM
- √ CD-ROM drive
- √ SVGA (800x600) or higher with 16-bit color
- √ USB
- √ 20MB available hard drive space

In order to assure image fluidity, the manufacturer recommends that you meet or exceed the following requirements. (This way the images viewed will not be herky-jerky.)

- √ Pentium II (or equivalent) running @ 450MHz
- √ Microsoft Windows 98, 98SE, ME, 2K, or XP
- √ 128MB of RAM
- √ CD-ROM drive
- √ SVGA (800x600) or higher with 32-bit color
- √ USB
- √ 20MB available hard drive space

EQUIPMENT REQUIREMENTS

To perform the various experiments referred to within this text, several items are required to be in your possession. Here's a list of everything you need:

- √ Understanding Signals Kit (#28119), see complete parts list table below.
- √ One of the following Board of Education platforms, sold separately:
 - Board of Education Full Kit (#28102, or #28103 without power supply)
 -OR-

- BASIC Stamp HomeWork Board (sold in 10-packs only, #28158) and Serial Cable (#800-00003)
- √ BASIC Stamp Editor 2.0: May 2003 or newer CD (#27000) or download at www.parallax.com.
- √ Universal infrared remote controller programmable for Sony TV's, needed for Chapter 7, Activity #2 only. (Not included, readily available at discount or electronics stores.)
- √ Fresh 9 V battery (not included).

UNDERSTANDING SIGNALS KIT

Understanding Signals Bill of Materials		
Parallax Part #	Description	Qty/Kit
70009	Understanding Signals Student Guide, Version 1.0	1
28014	OPTAScope, three probes, USB cable, and CD-ROM	1
900-00005	Parallax Standard Servo	1
451-00303	3-pin headers m/m	1
350-00009	Photoresistor	1
900-00001	Piezo speaker	1
800-00016	Jumper Wires, 10-pack	2
150-02210	220 Ω resistor ¼ watt 5% tolerance	3
150-01020	1 kΩ resistor ¼ watt 5% tolerance	3
150-01030	10 kΩ resistor ¼ watt 5% tolerance	3
150-02030	20 kΩ resistor ¼ watt 5% tolerance	3
150-02020	2 kΩ resistor ¼ watt 5% tolerance	3
602-00015	LM358 op-amp	1
201-01050	1.0 μF capacitor	2
201-01062	10.0 μF capacitor	2
152-01031	10 kΩ potentiometer	1
ADC0831	ADC0831 8-bit A/D converter	1
350-00003	Infrared LED emitter	1
350-90000	LED standoff	1
350-90001	LED light shield	1
350-00014	Infrared detector	1



150-02030
(3) 20 k ohm 1/4 watt 5% resistor
(red, black, orange)

150-01020
(3) 1 k 1/4 watt 5% resistors
(brown, black, red)

150-01030
(3) 10 k 1/4 watt 5% resistor
(brown, black, orange)

150-02020
(3) 2 k 1/4 watt 5% resistors
(red, black, red)

150-02210
(3) 220 ohm 1/4 watt 5% resistors
(red, red, brown)

800-00016
(20) 3" pluggable jumper wires

28014
(1) OPTAscope, three probes,
USB cable, and CD-ROM

451-00303
(1) 3-pin single row header

602-00015
(1) LM358 DIP op-amp

ADC0831
(1) 8-bit A/D converter

152-01031
(1) 10 k potentiometer

900-00001
(1) Piezo speaker

350-00009
(1) photoresistor

350-00014
(1) Infrared detector

900-00005
(1) Parallax
Standard Servo

350-00003
(1) LED Infrared T1 3/4

350-90000
(1) LED Standoff

350-90001
(1) LED shield for 350-90000

201-01050
(2) 1 uF electrolytic capacitor

201-01062
(2) 10 uF electrolytic capacitor

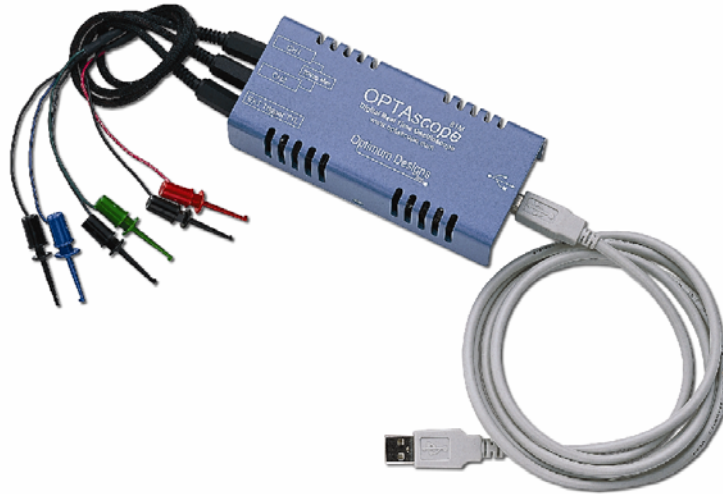
**Understanding Signals Parts Kit
#28119** - Components and quantities
are subject to change without notice.

Appendix B: OPTAscope 81M Specifications

B

OPTASCOPE 81M SPECIFICATIONS

- 2 Channels
- 1 Ms/s max. sample rate with one channel, 500 Ks/s with two channels
- View up to 60 kHz sine wave, up to 100 kHz square wave
- FFT function for signal analysis
- 20 Vpp maximum input for Channel 1 and Channel 2
- 200 kHz bandwidth
- 8-Bit vertical resolution
- External trigger source: TTL rising edge, 5 V TTL maximum input
- Trigger on rising or falling edge at any voltage
- Variable trigger voltage on both channels
- Horizontal trigger position settings at 10%, 50%, and 90%.
- Auto and Normal trigger modes
- 3 cursor options for measurements, and zoom capability
- 3 1X Probes included
- USB 1.1 for data and power supply; no separate power supply needed
- Size: 5" x 2.25" x 1.5" (excluding cable and probes)
- Weight: 8 ounces



Index

- A -

A/D converter, 65
 active channel, 8
 amplitude, 41, 43, 47, 107, 108, 109
 asynchronous data, 65
 asynchronous serial communication, 73
 attenuation, 44, 103, 105
 automatic measurements, 13
 Automatic Measurements display, 10
 Autoscale, 90
 Autoscale button, 12, 90

- B -

band pass filter, 81
 bandwidth, 101
 baud rate, 73, 76, 78
 Baudmode argument, 76
 binary signal, 76
 buffer, 102

- C -

calibration, 10
 capacitor, 27, 49, 52, 106
 polarity, 27, 52, 106
 channel buttons, 8
 clipping, 100, 105
 cursor, 10
 Cursor Settings switches, 12
 Cursors display, 10
 Cursors mode, 12
 Cursors tab, 12

delta measurement, 10

Floating, 12

Position Cursor buttons, 12

Snap to Plot, 12

Cursor Settings

 Paired Bars, 25

Cursor Settings switches, 12

 Floating, 12

 Horizontal Bars, 12, 115

 Paired Bars, 12, 29, 47, 56

 Snap to Plot, 12, 115

 Vertical Bars, 12, 91

Cursors mode, 12

Cursors tab, 12

- D -

DC offset, 113

delta, 10, 19

Display Screen, 10

distortion, 111

duty cycle, 36

dynamic range, 99, 105, 110

- E -

Export Data button, 10

Export Picture button, 10

external trigger, 14, 89, 90

- F -

f , 10, 78
false triggering, 90
Fast Fourier Transformation, 9, 45
FFT, 9, 45
Files/Settings tab, 10
Floating cursors, 12
Fourier analysis, 45
FREQOUT, 38, 40, 43, 86, 105, 108
frequency, 2, 10, 13, 35, 38, 43

- G -

gain, 99
Ground probes, 14

- H -

Horizontal Bars, 109
Horizontal dial, 7, 18, 78

- I -

infrared, 81
 detector, 81
 distance detection, 82
 IR protocol, 81
 object detection, 82
 remote control, 81
inverted data, 79
inverting voltage amplifier, 103

- M -

Measurements tab, 13
mixed signal, 44
mixed tone, 43
Mouse Function switch, 12
 Pan mode, 12

Zoom mode, 12

MSBPOST, 69

- N -

non inverting amplifier, 107
non-inverting voltage amplifier, 104

- O -

object detection, 82
OFF button, 8
op-amp, 99
 bandwidth, 101
 buffer circuit, 102
 dynamic range, 99
 gain, 99
 inverting voltage amplifier, 103
 non-inverting voltage amplifier, 104
 rail-to-rail, 99
 slew rate, 101
operational amplifier. *See* op-amp
OPTAscope 81M, 1
 hardware installation, 5
 trigger event ability, 3
OPTAscope Settings button, 10
oscilloscope, 1
 analog, 1
 digital, 1
 digital storage, 1
 safety guidelines, 4
output range, 99

- P -

Pan mode, 12
 parallel data transmission, 65
 phase shift, 44
 Plot Area, 6, 9
 Plot Area Indicator bar, 9
 Position Cursor buttons, 12
 Print button, 10
 Print Preview button, 10
 pulse train, 35, 90, 95
 pulse width modulation, 35
 and infrared, 81
 and servo control, 35
 PULSOUT, 24
 pure tone, 43
 PWM. *See* pulse width modulation

- R -

rail-to-rail op-amps, 99
 RC network, 49–51
 remote control, 91
 Reset Plots button, 13
 resistor, 49
 RS-232, 73, 79
 Run/Stop button, 8
 Run/Stop Mode, 11

- S -

sample rate, 2
 Sample Rate display, 10
 serial data transmission, 65
 SEROUT, 76
 servo, 23–24, 27
 and 3300 μ F capacitor, 27
 current draw, 27
 power supply voltage, 24

servo ports, 24
 signal

 attenuate, 105
 attenuation, 103
 clipping, 105
 distortion, 111
 inverted, 103

sine wave, 35, 57, *See* waveforms
 dual, 43

slew rate, 101
 Snap to Plot, 12
 software

 BASIC Stamp Editor, 13
 OPTAscope 81M, 5
 OPTAscope software, 13

spectrum analyzer, 9, 45
 square wave, 19, 35, 36, 55, 101
 supply rails, 112
 supply voltage, 99
 synchronous data, 65
 synchronous data transfer, 65

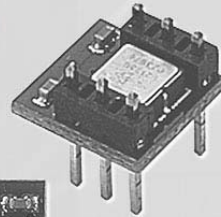
- T -

toggling an I/O pin, 19
 triangle wave, 101
 trigger, 2–3
 external trigger, 90
 falling edge, 18
 false triggering, 90
 position, 9
 rising edge, 18
 Run/Stop mode switch, 11

- Trigger Edge switch, 11
- Trigger Mode switch, 11
- trigger position, 9
- Trigger Settings display, 10
- Trigger Source switch, 11
- Trigger tab, 11
- Trigger Edge switch, 11
- trigger event
 - falling edge, 18
 - rising edge, 18
- trigger events, 3
 - falling edge, 3
 - post trigger, 3
 - pre-trigger, 3
 - rising edge, 3
- Trigger Mode switch, 11
 - Auto mode, 11
 - Normal mode, 11
- Trigger Settings display, 10
- Trigger Source switch, 11
- Trigger tab, 11
- true data, 79
- TTL threshold, 68
- V -
- Vertical dial, 8
- virtual ground, 103
- Vss, 14
- W -
- waveform
 - clipping, 100, 105
 - sine wave, 57
 - square wave, 19, 35, 36, 55, 101
 - triangle, 101
- Z -
- Zoom mode, 2, 12, 20
- Δ -
- Δ. *See* delta

Measure Acceleration

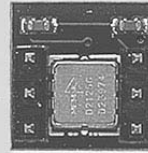
The Memsic 2125 is a low cost, dual-axis thermal accelerometer that is capable of measuring dynamic acceleration (vibration) and static acceleration (gravity) within a range of $\pm 2g$. For integration into existing applications, the Memsic 2125 is electrically compatible with other popular accelerometers and is easy to interface to the BASIC Stamp. Great to use in your next robotic, R/C airplane or alarm application.



Memsic 2125
Dual-Axis Accelerometer
#28017

PARALLAX

Order online at
www.parallax.com



Learn about **Basic Logic**
from the newest release in our
Stamps in Class Curriculum

This course is an introduction to logic in technology that bridges the gap between hardware and software. Students are directed to design solutions for real world applications first using hardware, then again using software. Along the way becoming familiar with schematic symbols, DC circuit theory, problem solving, and critical thinking.

Learn more about Stamps in Class by visiting www.parallax.com/sic

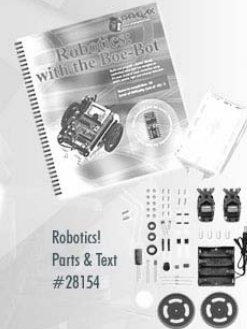
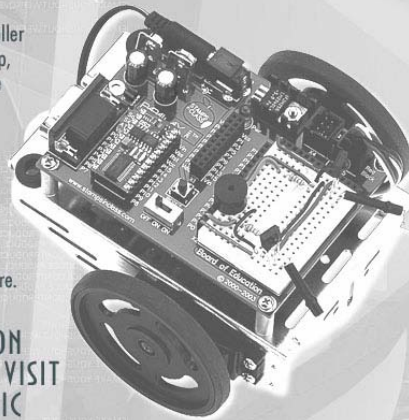
PARALLAX

LEARN ROBOTICS WITH THE BOE-BOT

If you enjoyed learning microcontroller programming with the BASIC Stamp, why not continue the learning curve by building a robot?

The *Robotics!* curriculum uses your BASIC Stamp 2 module and Board of Education to create a rolling robot that can follow or avoid light, detect and avoid objects with infrared, and much more.

FOR MORE INFORMATION
OR TO ORDER ONLINE VISIT
WWW.PARALLAX.COM/SIC



Robotics!
Parts & Text
#28154

PARALLAX

sensors!

Expand the capabilities of your next BASIC Stamp project with a sensor from Parallax. We stock an entire line of sensors that are compatible with the BASIC Stamp microcontroller.

Clockwise from top:

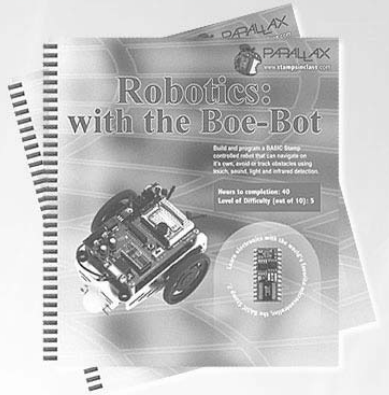
- TAOS TCS230 Color Sensor (#30054)
- Memsic 2125 Dual-Axis Accelerometer (#28017)
- Sensirion SHTX Humidity Sensor (#28018)
- FlexiForce Pressure Sensor (#30056)

For these and other sensors visit the
Component Shop at www.parallax.com

PARALLAX



More Educational Texts Available from Parallax:



www.parallax.com/sic

If you enjoyed this set of experiments, consider these other curriculums from the Parallax Stamps In Class program. All of our educational texts are available as free downloads online in .pdf format. Visit www.parallax.com/sic for details.

- What's a Microcontroller? (#28123)
- Basic Analog and Digital (#28129)
- Robotics: with the Boe-Bot (#28125)
- Applied Sensors (#28127)
- Industrial Control (#28156)
- Advanced Robotics: with the Toddler (#122-00001)

Look for these Stamps In Class curriculums coming soon!

- Basic Logic
- Understanding Signals
- Energy
- Microcontrollers for Artists and Engineers