Chapter 3: Human Speed Measurements

HUMAN SPEED VS. ELECTRONIC SPEED

3

"Human speed" and "electronic speed" are not really technical terms, but they can be useful for describing the differences between signals we can and cannot quantify without specialized equipment. Examples of human speed signals include street lights, musical rhythm, Morse code, and pulse rate. These signals can be quantified either intuitively, with some training, or in the case of pulse rate, with little more than a clock that displays seconds. Some signals seem to fall in both categories: audio tones for example. Different tones played by an audio speaker sound like they have different pitches. So, using just our ears, we can detect fairly subtle variations in the rate at which the speaker vibrates. On the other hand, it would be difficult to actually count the number of speaker vibrations per second to determine the exact frequency of the tone. That would require some specialized equipment.

You may have already programmed the BASIC Stamp microcontroller to send and measure signals at electronic speeds. Examples from What's a Microcontroller and Robotics with the Boe-Bot include controlling servo motors, measuring light or a dial's position, and communicating with a digital integrated circuit. The signals in those activities were quantified in terms of 2 microsecond (µs) increments. Typical human reaction times might be in tenths or even hundredths of seconds, but it takes a microcontroller or some other piece of "specialized equipment" to send or measure signals in terms of microseconds (millionths of seconds). If the electronic speed signals don't work as expected, it can also be difficult to diagnose without the aid of a signal measuring device, and that's where the PropScope proves itself to be exceedingly useful.

Before we move on to measuring electronic speed signals, this chapter provides a survey of many of the more commonly used PropScope measurement tools and techniques, applied to human speed signals with a variety of circuits. Human speed signal measurements provide a nice starting point with the PropScope. They make it possible to compare things we can manually initiate and monitor against the PropScope's graphical display and measuring tools. Since many of the measurement techniques are the same at both human and electronic speeds, taking human speed measurements first provides an opportunity to compare easily verifiable physical quantities against PropScope measurements. Having already measured similar signals at human speeds will also make it easier to proceed with confidence with electronic speed measurements in later chapters.

ACTIVITY #1: A POTENTIOMETER'S VARIABLE VOLTAGE OUTPUT

In the previous chapter, the potentiometer was wired as a voltage divider, and its W terminal output was measured as a DC voltage. With the same wiring and a simple adjustment of the Oscilloscope's time scale, you can instead use the PropScope to plot the W terminal's voltage variations against time as you turn the pot's knob back and forth. So, instead of viewing the potentiometer's W terminal as an adjustable DC voltage signal, we will use the PropScope view it as a time varying voltage signal.

Comment [AL25]:

ATTENTION READERS:

Make sure to try these instructions with PropScope software v1.0.7. You can find a link to a copy in a post at forums.parallax.com -> PropScope.

Potentiometer Voltage Parts List

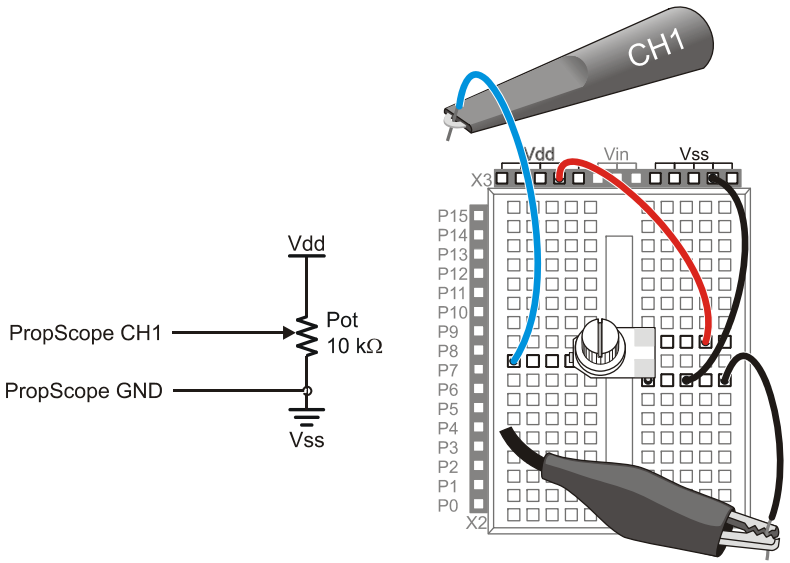
(1) Potentiometer – 10 kΩ
(misc) Jumper wires.

Potentiometer Voltage Test Schematic

Figure 3-1 is a repeat of Figure 2-23 from Chapter 2, Activity #4.

- ✓ Rebuild the circuit in Figure 3-1.

Figure 3-1: Potentiometer Voltage Divider Circuit

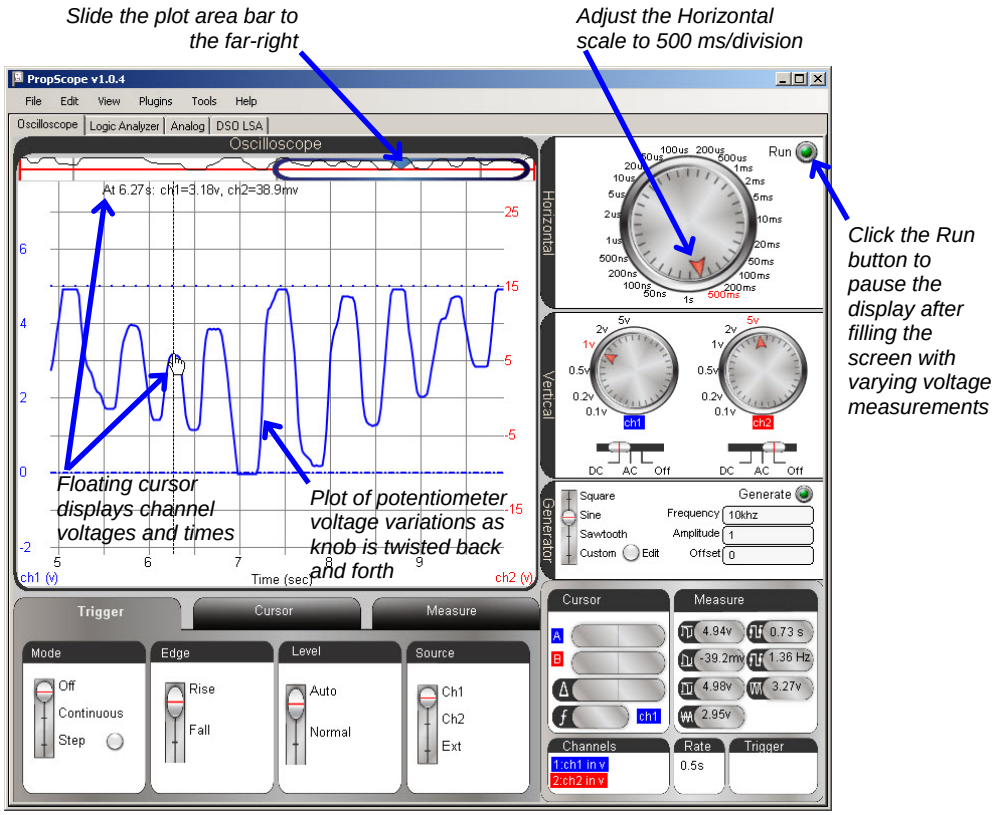


Potentiometer Voltage Test Procedure

Figure 3-2 shows an example of a running history of potentiometer W terminal voltages plotted by the PropScope. The wavy line is a plot of W terminal voltages over 5 seconds as the pot's knob is turned back and forth. Assuming your PropScope is still configured for Chapter 2 measurements, two PropScope settings have to be adjusted for this display. The first adjustment is to turn the oscilloscope's Horizontal knob to 500 ms. This sets the amount of time between two vertical lines in the oscilloscope display to 500 ms. ~~So, the~~ PropScope ~~can~~ display 500 ms worth of voltage measurements per time division. Since the Oscilloscope screen displays ten time divisions, it can display 5 seconds of plotted voltage activity because 500 ms/div × 10 div = 5000 ms = 5 s.

3

Figure 3-2: Test the Voltage Divider Output



Comment [AL26]:

To-do:

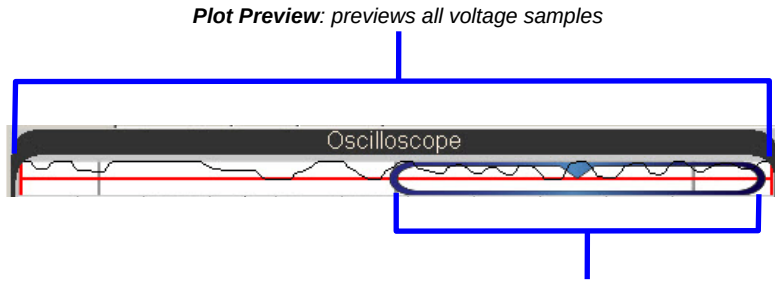
Update all screen captures with ones from final software revision.

These are 1.0.4 and 1.0.5 software vintage screen captures.

When the Horizontal knob is set ~~too~~ 200 ms, 500 ms, or 1 s/div, the PropScope displays in datalogging mode, which scrolls measurements from right to left. The most recent measurements appear on the right, and all the older measurements shift to the left. The further left on the plot you look, the older the measurements. This is similar to the way older polygraphs and seismographs draw plots, except that they feed paper under a pen that the device moves according to the property it measures. The most recent measurements are drawn by the pen, and as you look further along the paper that passed underneath the it, you are looking at older and older measurements.

To view the most recent PropScope measurements in datalogging mode, the second adjustment you'll have to make to get the display in Figure 3-2 is to move the Plot Area bar to the far right of the Plot Preview so that it matches Figure 3-3. The PropScope plots two oscilloscope screen~~'s~~ worth of measurements, and a ~~"~~preview~~"~~ of the entire plot is visible in the Plot Preview. See the squiggly line passing through the Plot Preview? That's the entire two oscilloscope screen~~'s~~ worth of plotted potentiometer voltages. By positioning the Plot Area bar in the Plot Preview, you choose the portion of the plot the Oscilloscope screen shows you. When you slide the Plot Area Bar to the far right, you will see the rightmost portion of the two screens worth of plotted measurements. At this position, the variations in the plotted voltages will be visible at the right side of the oscilloscope screen as soon as you make adjustments to the potentiometer.

Figure 3-3: Preview Bar and Plot Area Bar



- ✓ Make sure the CH1 coupling switch is positioned at DC.
- ✓ Turn channel 2 off by moving the CH2 coupling switch to Off. (The coupling switches are below the Vertical dials.)
- ✓ Set the CH1 Vertical dial to 1 V.

- ✓ Set the Horizontal dial to 500 ms.
- ✓ Click, hold, and drag the plot either up or down so that the dashed ground line and dotted maximum voltage line are positioned similarly to what's shown in Figure 3-2.
- ✓ Click, hold, and slide the plot area bar  to the far right of the Plot Preview as shown in Figure 3-3.
- ✓ Check the Run button, and make sure it is bright green, indicating the oscilloscope is running.
- ✓ Start twisting the potentiometer's adjusting knob back and forth. Remember to keep pressing it downward onto the breadboard to maintain electrical contact as you twist its knob.
- ✓ After the oscilloscope screen has filled up, click the Run button to stop the oscilloscope's plotting and freeze the waveform.
- ✓ Point at various points on the waveform. The floating cursor should display the voltage and time of the measurement you are pointing at.

Your Turn: Adjust the Visible Portion of the Plot with the Plot Area Bar

As mentioned earlier, the PropScope stores two Oscilloscope screen's worth of voltage samples. It also previews them at the top of the Oscilloscope display in the Preview Bar shown in Figure 3-3. You can use the Plot Area Bar to position the Oscilloscope screen over any portion of the Plot Preview, and then view the corresponding portion of the voltage plot in the oscilloscope screen.

- ✓ Click the Run button to resume plotting voltages.
- ✓ Keep adjusting the potentiometer's knob back and forth until the measurements span the entire Preview Bar.
- ✓ Stop the display again by clicking the Run Button.
- ✓ Try positioning the Plot Area Bar at different locations in the Preview Bar.
- ✓ Compare the small plot outlined by the Plot Area Bar in the Preview Bar to the full size one in the Oscilloscope display. The portion of the plot outlined by the Plot Preview Bar should be a miniature of the one in the oscilloscope display.

What if you want to check the voltage of the pot's W terminal once every second. While the Run button is stopped, you can use the time division lines as a guide for where to point the floating cursor to get measurements that are one second apart. Since each time division is 500 ms, two time divisions add up to 1 second's worth of plotted voltage measurements. So, use the floating cursor to point at the plot at every other vertical time division line to check the voltage at one second intervals.

Now, what if you want to check voltages once every second for 10 seconds? You can treat 0 seconds as the leftmost edge of the oscilloscope screen when the plot area bar all the way to the left, and treat 10 seconds as the rightmost edge when the plot area bar is all the way to the right. You will have to reposition the Plot Preview Bar at least once to check all ten measurements.

- ✓ The Run button should still be stopped so that you have a frozen plot of potentiometer measurements.
- ✓ Use the Floating Cursor to check the W terminal's voltages at seconds 0, 1, 2, 3, and up to second 10. Reposition the Plot Area Bar as needed.

At the far right of the plot, the measurements may end at a value like 9.97 s; you can use that as your 10 second measurement.

ACTIVITY #2: HIGH/LOW SIGNAL VOLTGES AND FREQUENCIES

Binary signals can be examined with an oscilloscope to get information about both timing and voltages. If only the timing needs to be examined, a device called a logic analyzer can be used instead. A logic analyzer doesn't plot actual voltage values, just high and low signal states. In this activity, you will use the Oscilloscope view to monitor both the voltages and the timing of two binary signals. In Activity #3, you will use the PropScope's DAC card and the PropScope software's Logic Analyzer view to monitor the high/low patterns and timing of four binary signals at once.

Parts List for Probing I/O Pins

(misc.) Jumper Wires

Circuit for Probing I/O Pins

The next example program will send high and low signals to the P14 and P15 I/O pins. Figure 3-4 shows how to connect the CH1 and CH2 probes directly to the I/O pin sockets so that you can monitor the signals with the PropScope.

- ✓ Connect the Probes as shown in Figure 3-4.

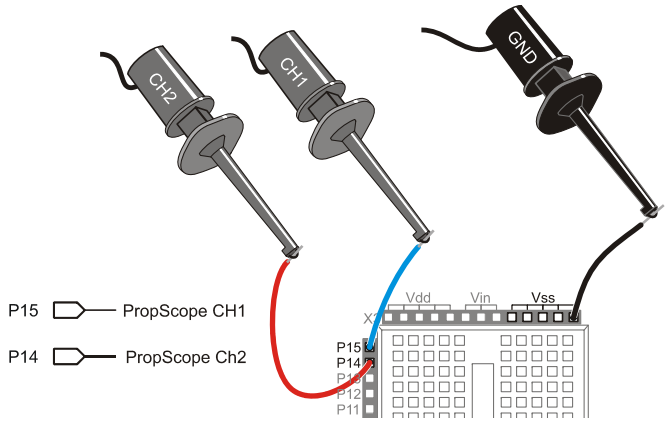


Figure 3-4: PropScope Probes Connected to Two I/O Pins

Comment [AL27]:

To-Do:

Update remaining wiring diagrams in this chapter with PropScope probes.

Example Program: Alternate High Low Signals.bs2

Alternate High Low Signals.bs2 is a program with a loop that starts by sending a high signal to P14 and a low signal to P15. After a 500 ms pause, it changes the P14 signal to low and the P15 signal to high. After another 500 ms pause, the program repeats the same signal sequence. Since the HIGH, LOW, and PAUSE commands are all in a DO…LOOP, the sequence repeats indefinitely.

✓ Enter and run Alternate High Low Signals.bs2.

```
' Alternate High Low Signals.bs2
' Turn LEDs connected to P14 and P15 on and off.  Alternate the on/off
' signals and repeat indefinitely.

' {$STAMP BS2}                                 ' Target module = BASIC Stamp 2
' {$PBASIC 2.5}                                ' Language = PBASIC 2.5

DEBUG "Program Running!"                       ' Program running message

DO                                             ' Main loop

  HIGH 14                                      ' P14 LED on
  LOW 15                                       ' P15 LED off
  PAUSE 500                                    ' Wait 0.5 seconds
  LOW 14                                       ' P14 LED off
  HIGH 15                                      ' P15 LED on
  PAUSE 500                                    ' Wait 0.5 seconds

LOOP                                           ' Repeat main loop
```

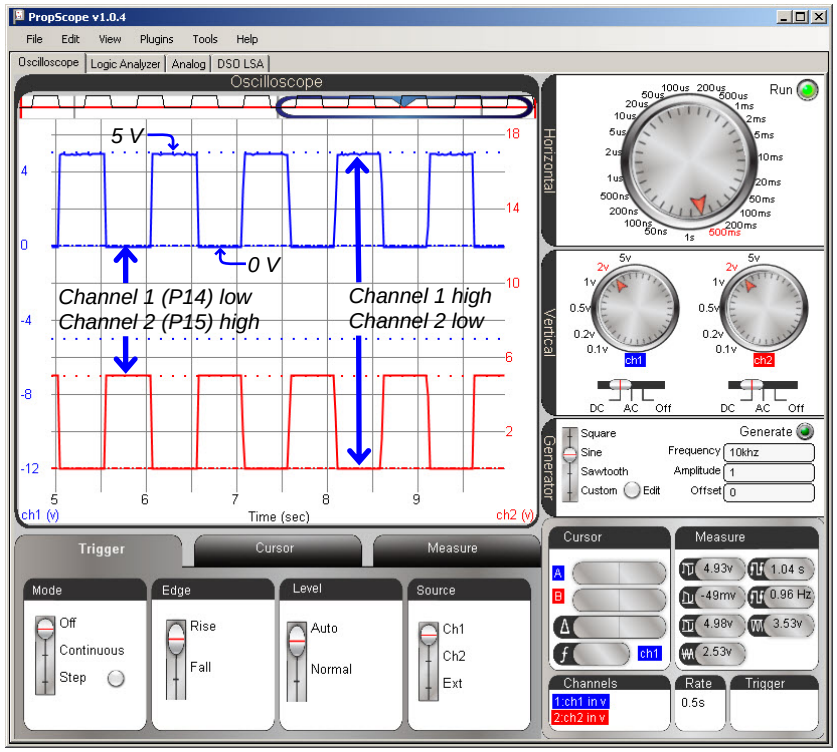
For more information about the program, go to www.parallax.com/go/WAM, and download What's a Microcontroller (.pdf or .zip). Work through Chapter 2, Activity #1 and Activity #2.

Oscilloscope Voltage Measurements

Let's examine the signals the BASIC Stamp transmits as it runs Alternate High Low Signals.bs2. Figure 3-5 shows the Oscilloscope view's representation of the high/low voltages the BASIC Stamp transmits as it executes the program. The channel 1 trace is monitoring the P15 signal, and the channel 2 trace is monitoring P14. When the channel 1 voltage is high, the voltage is 5 V. When it's low, its 0 V. The same applies to channel 2, except when the channel 2 signal is high, the channel 1 signal is low.

- ✓ If the plot is currently stopped, click the Run button to restart it.
- ✓ Restart the Channel 2 trace by moving the coupling slider below the CH2 Vertical Dial from Off to DC.
- ✓ Verify that the Horizontal dial is still set to 500 ms.
- ✓ Set both Vertical dials to 2 V.
- ✓ Click and drag the traces up/down to arrange them as shown in Figure 3-5.
- ✓ Verify that the high signal for each trace is 5 V, and the low signal for each trace is 0 V. Remember that the voltage scale for the channel 1 trace is on the left of the Oscilloscope screen and the scale for the channel 2 trace is on the right.

Figure 3-5: Binary Signals in the Oscilloscope View



Square Wave: The signals in Figure 3-6 are called square waves.

Figure 3-6 shows the Oscilloscope view's Measure tab, which now has more useful voltage information. The highest voltage (Vmax) is 4.98 V, and the lowest voltage (Vmin) is –49 mV, which is just below 0 V. The peak-to-peak voltage (Vpp), is the difference between Vhigh and Vlow, and it's 4.98 V.

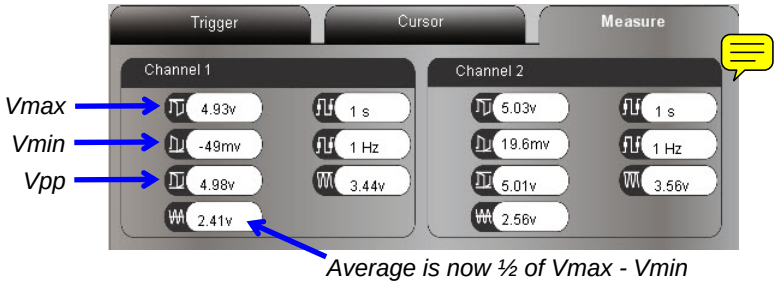
Comment [AL28]: Update values with figure after new screen capture.

The average voltage no longer represents a DC voltage. Now it represents the average of the channel's voltage over time, which is 2.41 V in the figure. Since the signal is 5 V half the time, and 0 V, the other half of the time, it stands to reason that the 'average' voltage of half way between these two values would be about 2 ½ volts.

Comment [AL29]: Update value with new screen capture that uses the latest software. It should be closer to 2.5 V.

- ✓ Click the Measure tab.
- ✓ Check the voltages for each channel in the Measure tab, and make sure they agree with what you see on the Oscilloscope screen.

Figure 3-6: Measure Tab Voltage Info for Binary Signal



Your Turn: A Closer Look at Average Voltage

What happens to the average voltage if the P14 signal is high less than half the time? Try making it high 1/5th of the time and see what happens to the average voltage measurement.

- ✓ Save a copy of Alternate High Low Signals.bs2 under a new name.
- ✓ In the copied program, change the first PAUSE command to PAUSE 100.
- ✓ Change the second PAUSE command to PAUSE 400.
- ✓ Load the modified program into the BASIC Stamp.
- ✓ Check the Measure tab. Is the Channel 1 average voltage close to 1 V?
- ✓ How about the P15 signal's average voltage?
- ✓ Compare the amount of time the Channel 1 and Channel 2 traces spend high to its cycle time. (The amount of time a signal takes before it repeats itself is called the cycle time.)
- ✓ Compare those time ratios to the average voltage measured for each channel.
- ✓ How does

  average voltage = 5 V × (high time ÷ cycle time)

  …compare with your measurements?

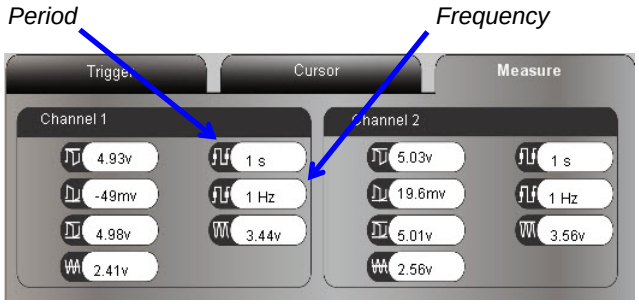
Oscilloscope Frequency Measurements

For a signal that repeats itself periodically (a periodic signal), the cycle time is also called its period. Again, that's the amount of time it takes a signal to repeat itself. ~~A signal's frequency is the number of times it repeats in one second.~~

3

The Measure tab's Period and Frequency measurements for Channel 1 are pointed out in Figure 3-7. The PBASIC code in Alternate High Low Signals.bs2 sets each I/O pin output state, pauses for 500 ms (1/2 a second), then inverts the I/O pin output states and pauses another 500 ms before repeating. Since the program makes the signal repeat after two half-second intervals, its period should be 2 × ½ second = 1 second. The measure tab provides a quick verification that the program is in fact repeating its signals with a period of one second (1 s). Another characteristic of a periodic signal is frequency, which is the number of times a signal repeats itself in one second. Since the signal takes a second to repeat itself every, the frequency is also one repetition per second or one hertz (1 Hz).

Figure 3-7: Measure Tab Frequency Info for 1 Hz Binary Signal



Comment [AL30]: Update to reflect adjustments in the previous Your Turn section. Average voltage should be 1 V.

Periodic Signal: A signal that repeats itself "periodically".

Cycle: A repetition of the signal.

Period: The time it takes for one cycle (repetition) of a periodic signal. You will see the terms period and cycle time used interchangeably in this book.

Frequency: ~~They~~ number of times in a second a signal repeats itself.

Hertz: ~~A measurement of frequency in cycles per second.~~

Adjust the Signal Frequency, Adjust the Display

Next, let's examine what happens if both PAUSE commands in Alternate High Low Signals.bs2 get reduced to PAUSE 50.

- ✓ Save another copy of Alternate High Low Signals.bs2.
- ✓ In this copy, change both PAUSE commands to PAUSE 50.
- ✓ Load the modified program into the BASIC Stamp.

Figure 3-8 shows how the higher frequency signals end up looking pretty cramped on the oscilloscope screen. With 500 ms/division, the Oscilloscope is showing the signal's activity over 5 seconds. In that amount of time, a signal that repeats every tenth of a second ends up repeating 50 times.

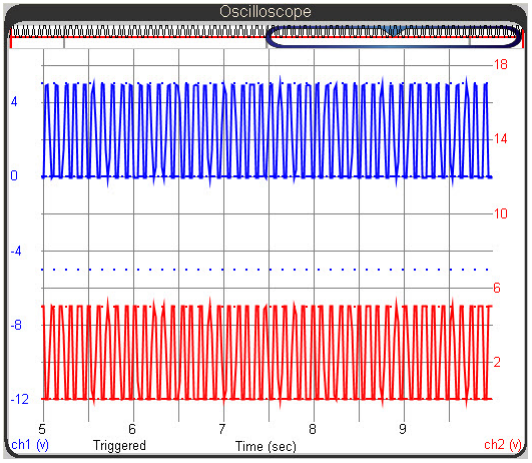


Figure 3-8
50 Cycles on the Oscilloscope Screen

Looks a little cramped, doesn't it?

When a signal repeats itself too quickly to see clearly in the Oscilloscope screen, the horizontal dial should be adjusted to a smaller time increment. It's usually best to pick a time increment that accommodates one or two cycles. Since our square wave now has a period of 1/10th of a second (100 ms), the ideal setting would be one that makes the oscilloscope screen 2/10ths of a second or 200 ms wide. Remember that the Horizontal dial selects the time per division, and remember also that one division is 1/10th of the oscilloscope screen. So, to display two cycles, the setting on the Horizontal dial should be 1/10th of 200 ms, which is 20 ms.

- ✓ Stop here and think very carefully about the previous paragraph. Here is a summary:

  Step 1: Figure out how much time a couple of signal cycles will take.
  Step 2: Divide that value by 10 to get your Horizontal dial setting. Again, that's because the Horizontal knob sets the time/division, and each time division is 1/10th the width of the oscilloscope display.
- ✓ Set the Horizontal dial to 20 ms.
- ✓ Check the time scale at the bottom of the Oscilloscope screen. Is it 200 ms wide?

The PropScope is now plotting in Oscilloscope Mode, not Datalogging mode.

The PropScope only plots in datalogging mode (from right to left) for the 200 ms/div, 500 ms/div, and 1 s/div.

For all other Horizontal knob settings, the PropScope plots measurements in oscilloscope mode, from left to right. In this mode, the display does not actually scroll. It only updates the display when all the measurements in the plot have been acquired.

That may have solved cramped display problem, but now, the signal still isn't staying still in the Oscilloscope screen! Now that the oscilloscope is at a Horizontal setting below 200 ms/div, it is functioning in oscilloscope mode, refreshing the display only after it has acquired all the measurements, and displaying those measurements from left to right. So instead of evenly scrolling, it now skips each time the display updates.

~~Taking measurements on a waveform that jumps around like that could be nerve wracking. So, oscilloscopes have a built-in feature called a trigger.~~ An oscilloscope trigger aligns the plot to an instant when the voltage either rises above or falls below a certain level. It also makes periodic signals like our square wave signal stay still in the Oscilloscope screen. Try this:

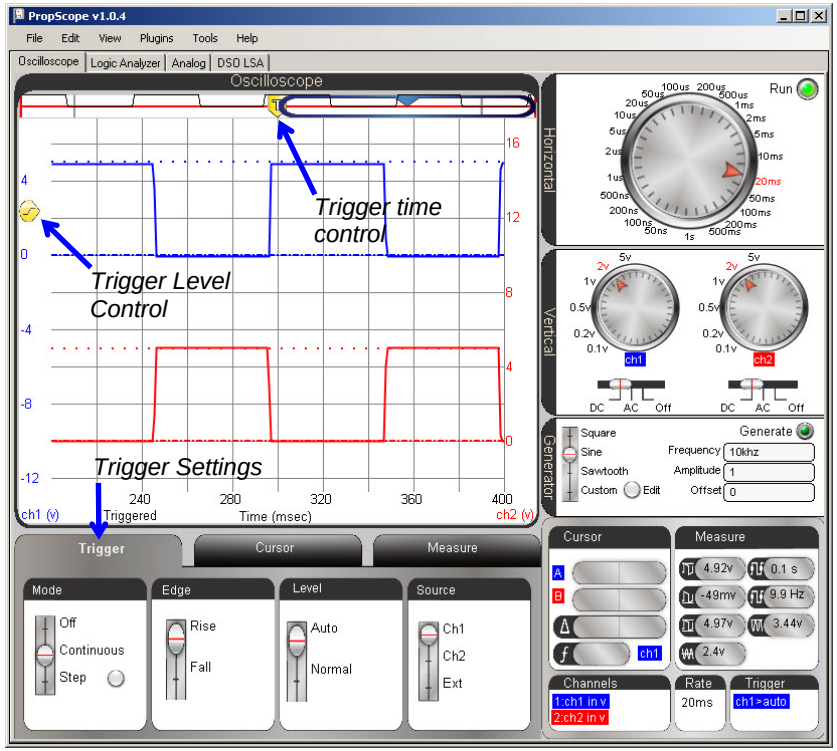
- ✓ Click the Trigger settings tab, and set the Mode to Continuous, the Edge to Rise, the Level to Auto, and the Source to CH1.

The display should now stay still in the Oscilloscope screen. Two new controls should have also appeared, the trigger voltage level control and the trigger time control. The trigger voltage control should have appeared to the left of the channel 1 trace as shown in Figure 3-9, and the trigger time control should have appeared in the Plot Preview. These controls set the time and voltage that the signal has to pass through to "trigger" a refresh of the display.

Comment [AL31]: To-do: rules for caps, then search and replace.

Comment [AL32]: To-do: rules for caps, then search and replace.

Figure 3-9: Trigger Setting Adjustments



Let's talk about what each of those Trigger tab settings and the two trigger controls do. The Source is CH1, meaning that the PropScope monitors channel 1 for a trigger event. Since Edge has been set to Rise, that trigger event happens when the voltage rises above a certain voltage. What voltage? With the Level set to automatic, the oscilloscope uses the signal's average voltage to set that voltage. (Yes, it's the same average voltage you would see in the Measure tab's Average voltage field.) As proof, notice on the left, that the trigger voltage control has been automatically positioned at the half way point between the channel 1 high and low signals. The Trigger tab's Mode is Continuous, which means as soon it's done plotting measurements, it'll start checking for another trigger event.

As soon as the PropScope is done with its current plot, it starts looking for another trigger event. Since the trigger voltage level is set to automatic, that'll be ~~at about~~ 2.5 V for the signal we are currently plotting. By adjusting the Trigger Time control, you'll be able to see where the trigger event occurs. Figure 3-10 shows an example of the crosshairs that appears as you point at the Trigger Time control. These crosshairs indicate the trigger time and voltage level. Since the Trigger tab's Edge has been set to Rise with a Source of CH1, the channel 1 voltage passes through the crosshairs as it transitions from low to high.

3

- ✓ If the trigger time control is not already in the Plot Area bar, click and hold the Trigger Time control, and drag it into the Plot Area bar. Otherwise, just point at the trigger time control with your mouse.
- ✓ A pair of crosshairs should appear in the oscilloscope screen. These crosshairs intersect at the location of the trigger event.
- ✓ As you move the Trigger Time control back and forth, the vertical crosshair should move with it. A rising edge in the channel 1 waveform plot should also move with it.

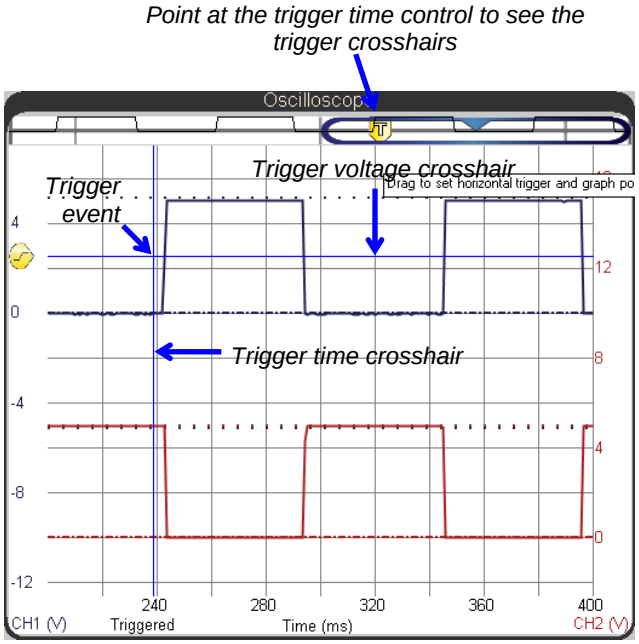


Figure 3-10
Adjusting the Trigger Time

The vertical and horizontal crosshairs that indicate the trigger time and voltage move when you adjust the Trigger Time control and the Trigger Voltage Control. Here we are only adjusting the Trigger Time Control because the trigger voltage is "automatic".

Comment [AL33]: In PropScope software v1.0.7, the signal should pass straight through the intersection of the crosshairs at any position in the plot.

When you use an oscilloscope's trigger, it's usually best to position the Oscilloscope screen to the far left in the plot by positioning the Plot Area Bar to the far left on the Plot Preview. The trigger time control should also be positioned somewhere within the Plot Area bar. This will make the event that "triggers" the plotting of the signal visible on the oscilloscope screen. ~~Since typical oscilloscope measurements examine what the signal does after the trigger event,~~ it's usually a good idea to position the trigger time control near the left side of the plot area bar.

- ✓ Slide the Plot Area bar all the way to the left in the Plot Preview.
- ✓ Slide the trigger time control to the position shown in Figure 3-11.
- ✓ As you slide the trigger time control into the Plot Area bar, the pair of crosshairs should that indicate the trigger voltage and time should reappear.
- ✓ Adjust the trigger time control's position slightly to make the vertical trigger crosshair (and the Channel 1 signal's rising edge) align with the 2nd time division line.

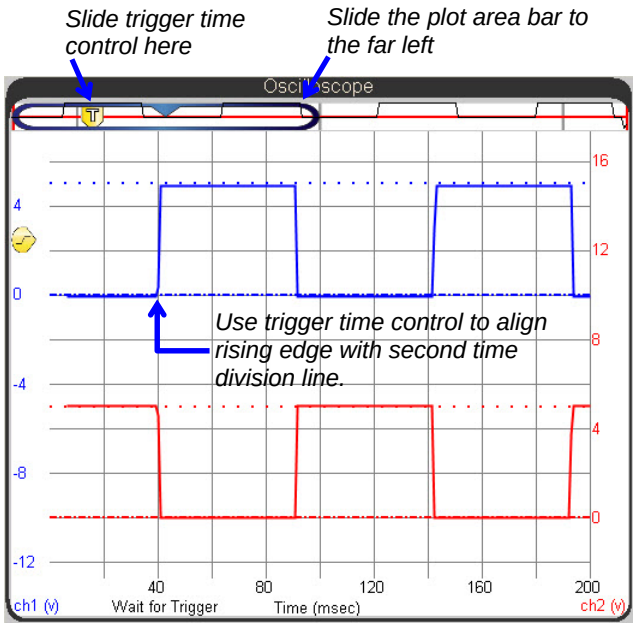


Figure 3-11
Adjusting the Trigger Time control's Position

After you slide the Plot Area Bar all the way to the left, slide the trigger time control to the approximate position shown in the preview bar. Then, slide it slowly adjust it until the rising edge of the channel 1 trace aligns with the 2nd time division line.

If you ever want to check the current trigger time or voltage, you can use your mouse to point at the Preview Bar or any of the scales (numbers around the plot). When you do that, it makes the Trigger Crosshairs appear. The vertical crosshair indicates the trigger time, and the horizontal crosshair indicated the trigger voltage level.

3

With the Trigger Level set to Automatic, the PropScope software positions the trigger voltage level control at the signal's average voltage, which is currently ~~half way~~ between the channel 1 low and high voltage levels. Inside the Trigger Level control is a line that transitions from low to high. This indicates that the Trigger Edge has been set to Rise which in turn means that the PropScope software will wait until it detects a voltage passing through the voltage crosshair's level as it's increasing. The display then aligns that rising edge with the trigger crosshair intersection and positions the rest of the plot accordingly.

- ✓ Try adjusting the Trigger edge to Fall. What happened to the Channel 1 trigger edge?
- ✓ Change the Trigger edge back to Rise. See the difference?

Figure 3-12 shows the measure tab. The period it displays is 0.1 seconds (s). This makes sense because the PAUSE command Duration arguments in the program were reduced from to 50. The signal is high for 50 ms and low for 50 ms, then high again for 50 ms, and so on.. With 50 ms high, and 50 ms low, the signal repeats itself every 100 ms, which is 0.1 seconds. This agrees with the value displayed in the Measure tab's Period field. A signal that lasts 0.1 seconds repeats 10 times in a second, so the frequency is 10 cycles per second or 10 Hz, which also agrees with the Measure tab's Frequency field.

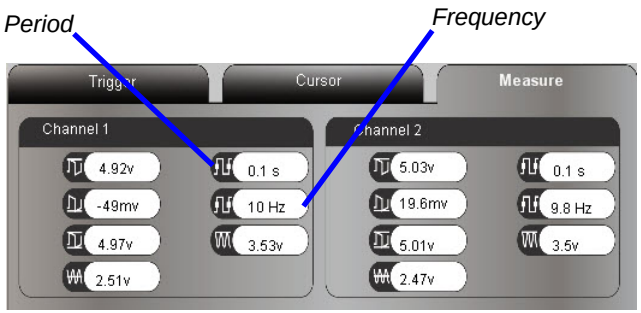


Figure 3-12
Measure Tab Frequency and Period for Channel 1

Notice that the Channel 1 frequency displays as 10 Hz while the Channel 2 frequency, which should be the same, displays at 9.8 Hz. The measure tab's automated

Comment [AL34]: Hmm, this seems difficult to follow. This explanation probably needs a diagram of an arbitrary voltage trace passing upwards through the crosshairs to indicate a rising signal trigger event.

Comment [AL35]: To-do: Get rid of the word edge here. The square wave happens to have an edge, but an edge is not necessary for a trigger event.

Comment [AL36]: Update values with new screen capture.

measurements are for getting a rough idea what the signal is doing. Tools and techniques for more precise measurements will be introduced at various points throughout the book.

Period (T) is the reciprocal of frequency (f). The time it takes a signal to repeat itself is called its period and is typically denoted by T and measured in units of seconds (s). The frequency of a signal is the number of times the signal repeats in one second, and it is denoted by f, and its measurement units are hertz (Hz).

$f = 1 \div T$ and $T = 1 \div f$

So, if you know the period is 0.1 seconds, you can use $f = 1 \div T$ to calculate the frequency $f = 1 \div 0.1\ s = 10$ Hz.

Look back at Figure 3-11. Notice that each signal has two rising edges displayed in the oscilloscope screen. These edges are also called positive edges. If the trigger is adjusted so that only one rising edge shows in the screen, the Measure tab will not update the period and frequency information for the signal. This will also happen if you reduce the time divisions (Horizontal dial) too f

Comment [AL37]: To-do: Test with 1.0.7 software. Is that still the case?

- ✓ Slowly adjust the trigger control to the right until one of the Measure tab Channel's period and frequency measurements disappears.
- ✓ Count the number of rising edges visible for that signal in the Oscilloscope screen.

Your Turn: Increase the Frequency Again

Setting the Horizontal knob incorrectly is a really easy mistake to make, and it can lead to incorrect or confusing measurements. So it's a good idea to practice the steps in this activity.

Comment [AL38]: To-do: Search knob, Replace with dial for PropScope adjustments, but not for the potentiometer. Details, details…

- ✓ Repeat the process of increasing the frequency again, this time by reducing the PAUSE commands in Alternate High Low Signals.bs2 to PAUSE 5.
- ✓ Predict the signal period based on a signal that's low for 5 ms and high for 5 ms before it repeats.
- ✓ Multiply your predicted signal period by 2, because we want to start by displaying about 2 cycles in the oscilloscope screen.
- ✓ Divide the amount of time the oscilloscope should display by 10 to get the per division value for the Horizontal dial setting.

The actual signal period will be slightly longer than predicted because of the time it takes for the BASIC Stamp to process all the commands: DO…LOOP, HIGH, LOW, PAUSE,

etc. This will also make your measured frequency slightly lower than the one you calculated. One of the Projects at the end of the chapter will challenge you to use the PropScope to fine tune your PAUSE command to make the signal timing more precise.

ACTIVITY #3: MULTIPLE HIGH/LOW SIGNALS

There are lots of situations where the signal voltages don't matter, but a group of signal states and their timing are important. The tool for these types of measurements is called a logic state analyzer, which is abbreviated LSA and often referred to as a "logic analyzer." Logic analyzers are especially useful for examining binary communication between devices. This type of communication typically involves the exchange of binary values (high/low signals) over multiple signal lines. The chapters on synchronous and asynchronous serial communication will rely heavily on the PropScope's logic analyzer features. The signals in those chapters occur at electronic speeds, which are much faster than you could observe with say, an LED.

In this activity, we'll use the logic analyzer with a human speed example that involves the pushbutton and LED circuits from What's a Microcontroller Chapter 3. The PBASIC program for the BASIC Stamp will monitor the pushbutton circuits and blink one of two indicator lights when one of two pushbuttons is pressed and held. The logic analyzer will be used to monitor and display the circuit activity as the application runs.

For more information about the circuit and program, go to www.parallax.com/go/WAM, and download What's a Microcontroller (.pdf). Work from the beginning of Chapter 3 through Activity #4.

Pushbutton and LED Parts

(2) Pushbuttons – normally open
(2) Resistors – 10 kΩ (brown-black-orange)
(2) Resistors – 220 Ω (red-red-brown)
(2) LEDs – any color
(3) Jumper wires

Pushbutton and LED Circuit

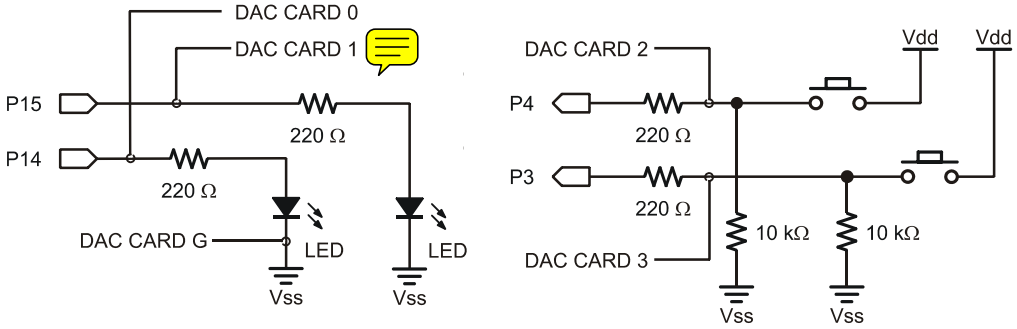
Figure 3-13 shows a schematic of four binary circuits, two LED indicator lights and two pushbuttons. It also shows connections between these circuits and the PropScope DAC CARD's four logic analyzer channel inputs, which are the sockets labeled 1 through 4. There is also a mandatory common ground connection, which is the wire between Vss on

your board and G on the DAC card. Vss is your board's ground, and G is the DAC CARD's Logic Analyzer ground connection. Figure 3-14 shows a wiring diagram example for the same circuit. These connections will make it possible to view the plots of the binary signal exchanges between the circuits and the BASIC Stamp I/O pins with the PropScope's LSA (logic state analyzer) feature.

- ✓ Build the circuit and make the DAC card connections shown in Figure 3-13 and Figure 3-14.

The DAC CARD's 1 to 4 inputs, will be displayed as logic analyzer channels 1 through 4. The socket labeled G is the logic analyzer's ground connection, and it should be connected to your boards Vss ground. This gives the systems a "common ground" so that the logic analyzer has a 0 V reference to compare the voltage signals it measures.

Figure 3-13: Pushbutton and LED Circuit with Logic State Analyzer Connections



Comment [AL39]: To-do: Swap DAC CARD 0 and DAC CARD 1 in schematic only.

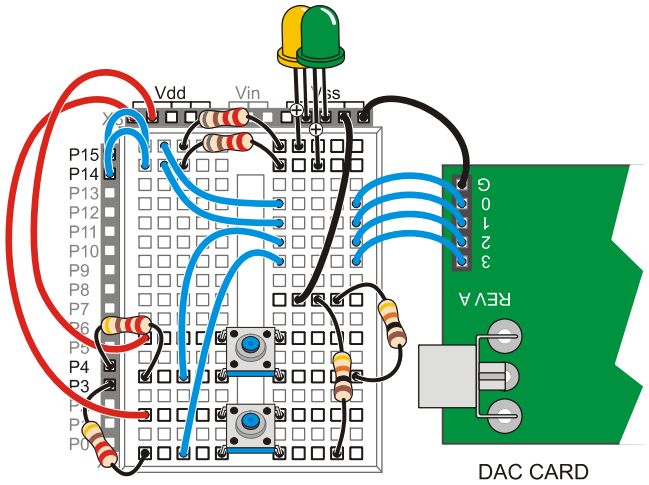


Figure 3-14
Wiring Example for the Figure 3-13 Schematic

3

The "DAC" in DAC CARD stands for Digital to Analog Conversion and refers to the card's ability to set voltages and synthesize signals. The PropScope's DAC card is really more of a multipurpose card with DAC being one of its features. Other features include: an input that can be used to trigger measurements, a signal generator that can transmit a variety of signals, a video generator signal output, and a 4-chanel logic analyzer input

When the DAC CARD is in use, PropScope CH2 is disabled. You can still take measurements with CH1, just not CH2.

Example Program: PushuttonControlOfTwoLeds.bs2

PushbuttonControlOfTwoLeds.bs2 makes the P14 LED blink on/off at 10 Hz while the P3 LED is pressed, or it makes the P15 LED blink on/off at 10 Hz while the P4 button is pressed.

- ✓ Enter PushbuttonControlOfTwoLeds.bs2 into the BASIC Stamp Editor and load it into the BASIC Stamp.
- ✓ Test and verify that the P15 LED blinks while the P4 pushbutton is pressed and held.
- ✓ Repeat for P14 LED control with the P3 pushbutton.

```
' What's a Microcontroller - PushbuttonControlOfTwoLeds.bs2
' Blink P14 LED if P3 pushbutton is pressed, or blink P15 LED if
' P4 pushbutton is pressed.

' {$STAMP BS2}                                ' Target module = BASIC Stamp 2
```

```
' {$PBASIC 2.5}                                ' Language = PBASIC 2.5

PAUSE 1000                                     ' Wait 1 second before DEBUG

DO                                             ' Main loop

  DEBUG HOME                                   ' Top-left position in terminal
  DEBUG ? IN4                                  ' Display IN4 = value
  DEBUG ? IN3                                  ' Display IN3 = value

  IF (IN3 = 1) THEN                            ' If P3 button pressed
    HIGH 14                                    ' P14 LED on
    PAUSE 50                                   ' Wait 1/20th of a second
  ELSEIF (IN4 = 1) THEN                        ' Else if P4 button pressed
    HIGH 15                                    ' P15 LED on
    PAUSE 50                                   ' Wait 1/20th of a second
  ELSE                                         ' Else no buttons pressed
    PAUSE 50                                   ' Just wait 1/20 seconds
  ENDIF                                        ' No more conditions in chain

  LOW 14                                       ' Turn P14 and P15 LEDs off
  LOW 15

  PAUSE 50                                     ' Wait another 1/20th second

LOOP                                           ' Repeat main loop
```

Logic Analyzer Measurements

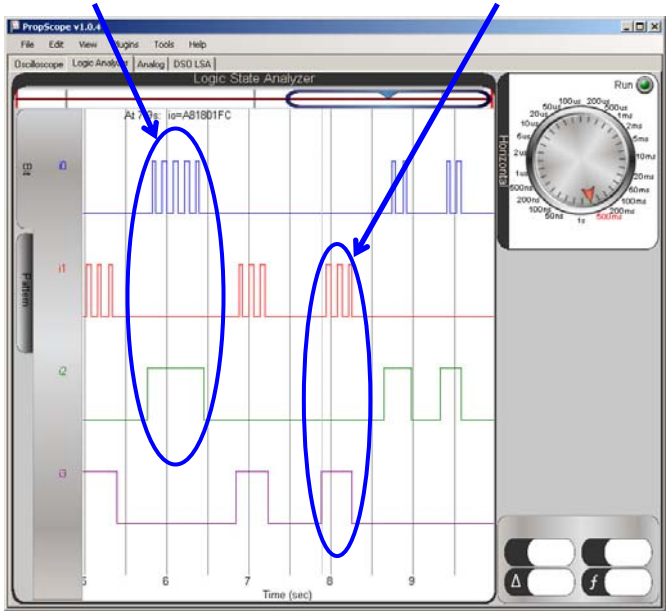
Figure 3-15 shows a plot of the binary pushbutton and LED circuit activity in the PropScope software's Logic Analyzer view.

- ✓ Click the Logic Analyzer tab.
- ✓ Set the Horizontal dial to 500 ms.
- ✓ Slide the plot area bar  to the ~~far right~~ of the Logic State Analyzer's ~~left…right~~ range.
- ✓ Try briefly pressing and holding each button, and verify that the Logic Analyzer view correctly reports the binary activity you create on your board.
- ✓ Also watch the LED activity as you press and hold a pushbutton, and make sure to check the Debug Terminal display too.

Figure 3-15: Pushbutton and LED Activity in the Logic Analyzer View

P15 LED (i0) blinks while P4 button (i2) is pressed and held

P14 LED (i1) blinks while P3 button (i3) is pressed and held



Comment [AL40]: Correct according to wiring diagram, but not according to schematic. Again, adjust schematic to wiring diagram.

3

Your Turn: Find and Fix the Bug!

PushbuttonControlOfTwoLeds.bs2 has a bug discussed and fixed in What's a Microcontroller, Chapter 3, Activity #4. The bug is that only one LED blinks when both buttons are pressed.

- ✓ Use your PropScope's Logic Analyzer to view this symptom as you press and hold both buttons. Only one of the signals monitoring LED circuits should toggle even though both i3 and i4 signals are high.
- ✓ Correct the problem in the PBASIC code, and load the modified code into the BASIC Stamp. Hint: Either break the IF…THEN…ELSE…ENDIF block into two separate IF…THEN…ELSE blocks, or look up the other solution near the end of Activity #4 in What's a Microcontroller, Chapter 3.
- ✓ Use your PropScope's Logic Analyzer view to demonstrate that the modified code corrected the problem.

ACTIVITY #4: DAC AND FUNCTION GENERATOR WAVEFORMS

The same DAC features we set DC voltages with in Chapter 2, Activity #3 can also be used to synthesize time varying waveforms.

DAC Parts List

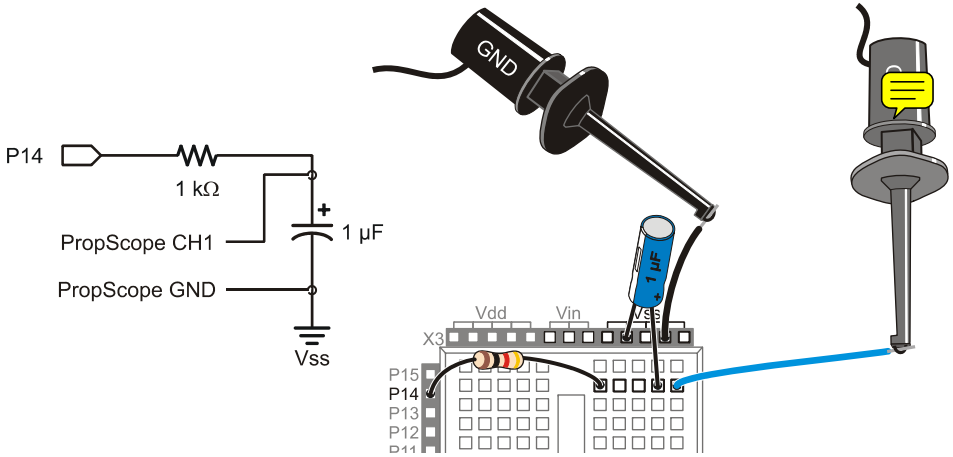
(1) Resistor – 1 kΩ (brown-black-red)
(1) Capacitor – 1 µF
(misc) Jumper Wires

DAC Circuit

Figure 3-16 shows a schematic and wiring diagram of one BASIC Stamp D/A conversion (DAC) circuit along with a PropScope probe attached for measuring the DAC output voltages.

- ✓ Build the circuit shown in Figure 3-16.
- ✓ Verify that the negative terminal of the capacitor is connected to Vss before reconnecting power to your board.

Figure 3-16: One DAC Circuit



Here is a program that repeatedly sweeps the PWM command's Duty value from 0 to 255. This repeatedly sweeps the voltage across the capacitor from 0 to 4.98 V, and it creates a voltage waveform that can be viewed with the Oscilloscope.

```
' Test Saw Tooth.bs2

' {$STAMP BS2}                                 ' Target module = BASIC Stamp 2
' {$PBASIC 2.5}                                ' Language = PBASIC 2.5

dacV VAR Byte                                  ' Byte variable declaration.

PAUSE 1000                                     ' 1 s before DEBUG
DEBUG "Program running..."                     ' Debug Terminal message

DO                                             ' Main loop

    ' dacV variable sweeps from 0 to 255, DAC output sweeps from
    ' 0/256 to 255/256 X 5 V.
    FOR dacV = 0 TO 255                        ' FOR...NEXT loop repeats 255x
      PWM 14, dacV, 1                          ' dacV sweeps from 0 to 255
    NEXT                                       ' Next repetition

LOOP                                           ' Repeat main loop
```

The oscilloscope screen in Figure 3-17 shows a 2 second time window, which means that the oscilloscope preview spans 4 seconds. This time, the Plot Area Bar is on the left with the Horizontal dial set to 200 ms/div. So, it may take a couple of seconds before the waveform appears. The signal will first be visible in the Plot Preview as it makes its way toward the oscilloscope window. Your tasks are to:

- ✓ Change the Trigger tab's Mode setting from Continuous to Off.
- ✓ Slide the Plot Area bar to the far left of the Plot Preview.
- ✓ Adjust the Horizontal dial to 200 ms/division.
- ✓ Load the modified program into the BASIC Stamp.
- ✓ Wait a couple of seconds for the waveform to make its way from the right of the plot preview and into the Plot Area bar and your oscilloscope screen.
- ✓ Click the Run button to stop the oscilloscope and freeze the display.

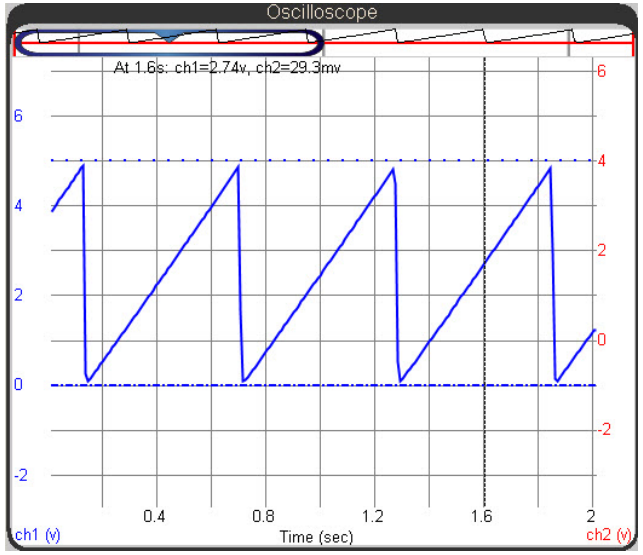


Figure 3-17
Channel 1 Trace

…connected to P14 DAC circuit output with Test Saw Tooth.bs2 running.

Your Turn: Sawtooth vs. Triangle wave

The saw tooth function in Figure 3-17 ramps slowly upward from 0 V to 4.98 V and then dives back to 0 V. In contrast, a triangle wave ramps slowly up, and then ramps slowly back down again. You can add a second FOR…NEXT loop after the first one to make the waveform ramp back down for a triangle wave.

- ✓ Save Test Saw Tooth.bs2 as Test Ramping.bs2.
- ✓ Insert this FOR…NEXT loop after the first one. In other words, this loop should occupy new lines between the NEXT and LOOP commands in the existing program.

```
FOR dacV = 254 TO 1        ' FOR...NEXT loop repeats 255x
  PWM 14, dacV, 1          ' dacV sweeps from 0 to 255
NEXT                       ' Next repetition
```

- ✓ Load the modified program into the BASIC Stamp, and remember to give the new waveform a couple of seconds to make its way across the Plot Preview and into the Oscilloscope screen.

Comment [AL41]: Insert email reply to Lisa about how the code works below this checklist instruction.

PropScope Function Generator Waveforms

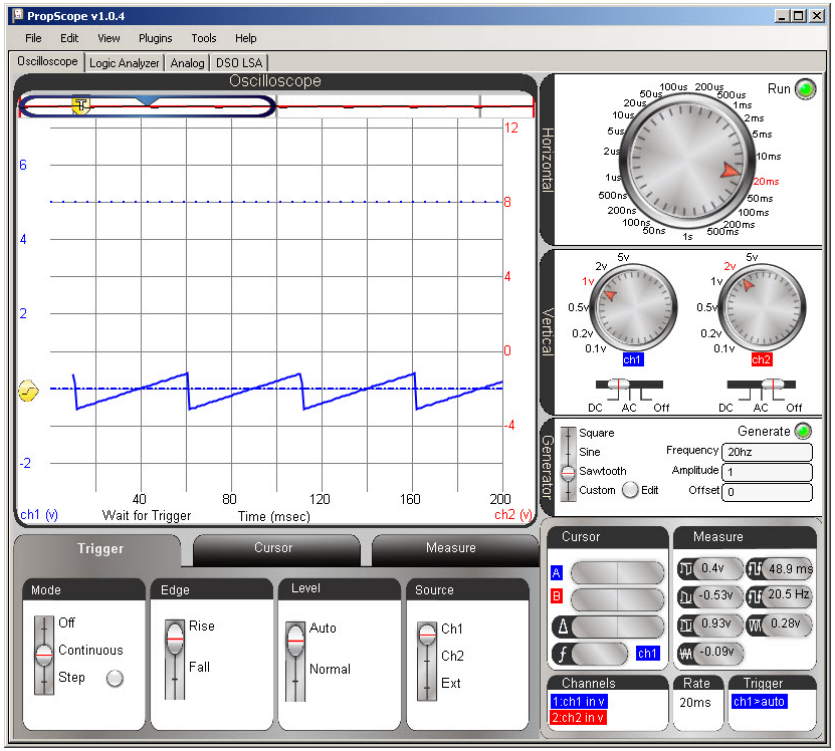
The PropScope's DAC card and Generator panel can be used to generate square, sine, and saw tooth waves. The Generator panel also has a custom setting that you can use to draw your own waveform with the Edit feature. Many circuit tests involve applying a signal and examining the affect on the output. So the PropScope's function generator feature is exceedingly useful for applying a signal to the circuit. Then, the output can be examined with the oscilloscope channel 1.

- ✓ Repeat the four checklist instructions in the Set DC Voltages with the PropScope's DAC CARD section on page 43.
- ✓ If the Run button is off, click it to restart the oscilloscope.

Figure 3-18 shows a 20 Hz sawtooth waveform generated by the DAC CARD's DAC output and monitored and displayed by Channel 1.

- ✓ In the Generator panel, set the slider to Sawtooth, the Frequency to 20 Hz, Amplitude to 1, Offset to 0, and then Click Generate.
- ✓ Set the Horizontal dial to 20 ms, and the Vertical CH1 dial to 1 V.
- ✓ Click the Trigger tab and set the Mode to Continuous, Edge to Rise, Level to Auto, and Source to CH1.
- ✓ Adjust the Trigger Time so that the vertical trigger crosshair aligns with the second time division line.

Figure 3-18: Sawtooth Waveform



The sawtooth wave in Figure 3-18 has a peak-to-peak amplitude of 1 V. True to its name, the peak-to-peak amplitude is the voltage between the top and bottom voltage peaks. This waveform swings from about ½ V above the 0 V ground line to about ½ V below, so its amplitude is a total of 2 × ½ V = 1 V.

- ✓ Try increasing the amplitude to 2 V, and observe the result.
- ✓ Try increasing the amplitude to 3 V, and observe the result again.

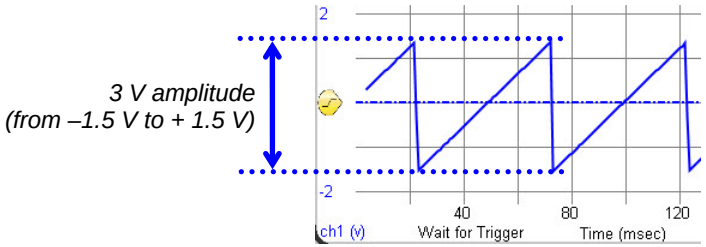


Figure 3-19
Sawtooth Wave with 3 V Amplitude

3

The DAC CARD can generate signals that fit into one of two ranges:

- –1.5 V to + 1.5 V
- 0 V to 4.7 V

If the amplitude or offset settings exceed the limits, the PropScope software will display an error message. For example, if you tried to enter a 5 V amplitude and a 3 V offset, it would display the error. A 5 V amplitude with a 3 V offset means that the signal should swing from 0.5 V to 5.5 V. That's the DC offset of 3 V + 2.5 V for the top peak and 3 V – 2.5 V for the bottom peak.

Comment [AL42]: To-do: Test this on the latest software.

The sawtooth wave's Offset setting is currently 0 V. DC offset is a DC voltage component that can be added to the signal. Let's see what happens when you add a 2 V DC offset to the signal.

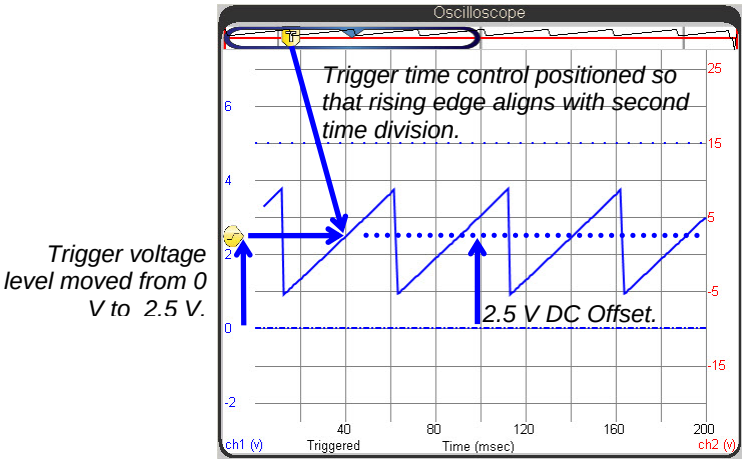
- ✓ Adjust the Generator panel's Offset to 2.5.

If the waveform does not appear to shift upward and the display stops refreshing, its because the signal no longer crosses the trigger level.

Comment [AL43]: To-do: Check if you still need this or if the trigger follows automatically now.

- ✓ In the Trigger panel, change the level switch from Auto to Normal.
- ✓ Drag the trigger level control up to the 2.5 V level as shown in Figure 3-20.
- ✓ Adjust the position ~~of the~~ of the Trigger Time control so that the middle of the upward ramp aligns with the second time division. (The vertical Trigger Crosshair should help here.)
- ✓ Change the Offset to 1.5 V and observe the effect.
- ✓ Repeat for an offset of 3 V.

Figure 3-20: Trigger Control Positions and DC Offset



The frequency adjustment is also simple.

- ✓ Try these values in the Generate panel's frequency field: 10 Hz, 15 Hz, 20 Hz, 25 Hz, 30 Hz.
- ✓ Examine the Oscilloscope display as you enter each successive value into the Generator panel's Frequency field. Make sure to press the Enter key on your keyboard after typing each new value.
- ✓ How does the appearance of the waveform change with higher and lower frequencies?

Your Turn: Other Waveforms

You can make similar adjustments to the Sine, Square, and Custom waveforms. First set the offset, amplitude and frequency values.

- ✓ Set Offset to 2.5 V, Amplitude to 4 V, and Frequency to 20 Hz.

Next, move the slider switch to the different waveforms and observe the change in the display:

- ✓ Set the Generator slider to Sine.
- ✓ Set the Generator slider to Square.

✓ Set the Generator to Custom, click Edit, and draw a waveform in the Waveform Editor. Click Update to make the DAC CARD start transmitting your custom waveform.

ACTIVITY #5: SIMULATED HEARTBEAT

What could be more "Human Speed" than a heartbeat signal? Modern heart monitors are oscilloscopes with special features, and specialized probes that filter and amplify the very small electrical signals from contact points on the body. The BASIC Stamp can be programmed to emulate an amplified heartbeat signal with its PWM command and DAC circuit. Displaying and examining a heartbeat signal like the one in Figure 3-21 with the PropScope provides an example of why it's important to understand the basic measurement techniques and not depend too heavily on the automated measurements.

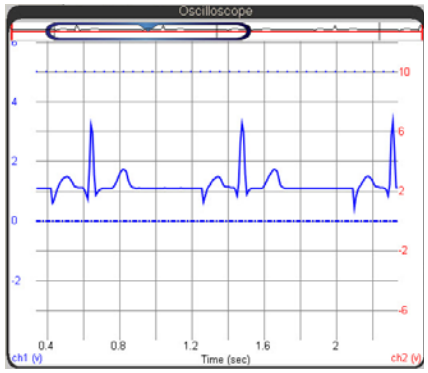


Figure 3-21
BASIC Stamp Generated Heartbeat Signal in the Oscilloscope

Probe Adjustments

This activity relies on the BASIC Stamp DAC circuit from the previous activity.

✓ If you have not already done so, reconnect the CH1 probe to the P14 DAC circuit output. (See Figure 3-16 on page 102.)

Example Program: Test Heartbeat.bs2

Test Heartbeat.bs2 emulates a heartbeat signal by fetching successive values from the heartbeat DATA directive near the bottom of the program. This DATA is stored in an unused portion of the BASIC Stamp module's EEPROM program memory and the READ command uses an address of heartbeat + index to retrieve each value. The FOR…NEXT loop increments the index variable from 0 to 99, so the READ command

fetches the next value in the list each time through the FOR…NEXT loop. The READ command also places the value it fetched from EEPROM into the dacV variable each time through the FOR…NEXT loop. Then, the PWM command uses the dacV variable to modify the voltage across the DAC circuit's capacitor. Since the entire FOR…NEXT loop is nested in a DO…LOOP, the signal repeats itself indefinitely.

- ✓ Load Test Heartbeat.bs2 into the BASIC Stamp.

```
' Test Heartbeat.bs2

' {$STAMP BS2}                              ' Target module = BASIC Stamp 2
' {$PBASIC 2.5}                             ' Language = PBASIC 2.5

index VAR Byte                             ' Variable declarations
dacV  VAR Byte

PAUSE 1000                                 ' Delay before first message
DEBUG "Program running..."                 ' Debug Terminal message

DO                                         ' Main loop
  FOR index = 0 TO 99                      ' Repeat 99x, sweep index var
    READ heartbeat + index, dacV           ' Read index value from heartbeat
    PWM 14, dacV, 1                        ' Fetched READ value sets DAC
    PAUSE 5                                ' Wait 5 ms before next iteration
  NEXT                                     ' Next iteration
LOOP                                       ' Repeat main loop

' Hand digitized values from an ECG printout.
heartbeat DATA 060, 063, 067, 069, 075, 077, 080, 082, 082, 080,
               075, 070, 063, 061, 061, 061, 061, 061, 061, 061,
               054, 044, 035, 130, 229, 183, 085, 013, 037, 060,
               060, 060, 060, 060, 060, 060, 060, 060, 060, 062,
               069, 075, 081, 086, 090, 093, 095, 092, 086, 069,
               060, 060, 060, 060, 060, 060, 060, 060, 060, 060,
               060, 060, 060, 060, 060, 060, 060, 060, 060, 060,
               060, 060, 060, 060, 060, 060, 060, 060, 060, 060,
               060, 060, 060, 060, 060, 060, 060, 060, 060, 060,
               060, 060, 060, 060, 060, 060, 060, 060, 060, 060
```

The DATA table in Test Heartbeat.bs2 was developed by treating an electrocardiograph printout as a series of values plotted on a graph. The graph used a vertical scale of 0 to 255, so that it would fit nicely into the PWM command's Duty argument for generating voltages from 0 to 255 (0/255ths to 255/255ths of 5 V). Some head and foot room was incorporated into the vertical scale so that the DAC output did not reach all the way to 0 or 255. The lowest value is 13 and the highest is 229.

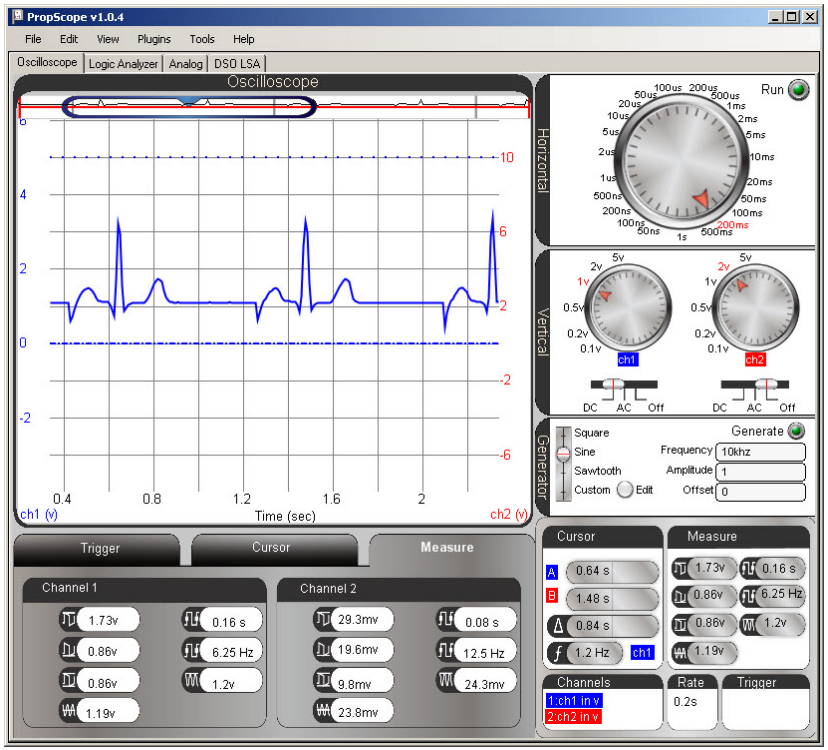
Measure the Simulated Heartbeat Signal

Figure 3-22 shows the PropScope settings to display the heartbeat. Remember that you can adjust the waveform's vertical position by sliding it up/down with your mouse. Just point at either the waveform or the dashed ground line for that channel, then click, hold, and drag.

Comment [AL44]: To-do: Test if you just click and drag the plot background for CH1, and update if needed.

- ✓ Update your PropScope settings for a similar display. Pay close attention to the Horizontal and CH1 Vertical scales, the Plot Area Bar's position, and the vertical position of the waveform.
- ✓ If the Run button is off, click it to restart the display.
- ✓ Locate the frequency for Channel 1 in the Measure tab. What's the patient's heart rate according to the automated Measure tab?

Figure 3-22: Pulse Rate in Measure Tab is not Correct



The Trigger tab's Mode setting automatically turns Off when the Horizontal dial is set to 200 ms or larger. That's because the oscilloscope goes into datalog mode with timescales of 200 ms/div or larger, and datalog mode is not compatible with triggers. A trigger aligns the rest of the plot around some trigger condition in oscilloscope mode. In datalog mode, the oscilloscope scrolls continuously, so it cannot be forced to align with a particular trigger event.

Beware of Automated Measurements

If we rely on the Measure tab for this patient's heart rate, it would be a frequency of 6.25 Hz. That's 6.25 beats per second × 60 seconds/minute = 375 beats per minute, and our patient would be in serious trouble! Fortunately for our patient, this measurement is not correct.

Comment [AL45]: To-do: Retest with PropScope v1.0.7.

From the PropScope's standpoint, ~~its~~ an understandable measurement because the PropScope calculated the average voltage of this signal to be 1.19 V, and it's counting the number of times per second the voltage crosses this level. However, that's not the ~~"~~right~~"~~ measurement for a heart rate.

Comment [AL46]: To-do: Update value with new screen capture.

Time Division and Vertical Cursor Measurement Examples

If we instead measure the time between the high points in the heart rate signal, we can get the correct pulse rate. Let's look at two different ways to take the manual measurements: by time division and with cursors.

Count Time Divisions

The period of the pulse signal can be approximated by counting the number of 200 ms divisions between the signal peaks. It looks like they are just over four 200 ms time divisions apart; let's call it 4 1/8. This value along with the fact that frequency is the reciprocal ~~if~~ period (f = 1/T) is enough information ~~for~~ the pulse rate's frequency in Hz. Then, to convert from Hz to the familiar beats per minute, just multiply by 60 seconds/minute.

Step 1: Figure out the signal period T based on the number of divisions between peaks.

$$T = 4.125 \text{ divisions} \times 200 \text{ ms/division} = 825 \text{ ms}$$

Step 2: Use the period to calculate the frequency in Hz.

$$f = 1 \div T$$

= 1 ÷ 825 ms = 1 ÷ (825/1000 s) = 1000 ÷ 825 s = 1.2121… Hz

Step 3: Multiply the Hz measurement, which is beats per second, by 60 seconds/minute get the familiar beats per minute (bpm) heart rate measurement.

f(bpm) = f(Hz) × 60 seconds/minute
= 1.2121…beats/s × 60 s/m = 72.7272…beats/m
≈ 72.7 bpm

Alright, our patient isn't so bad off after all.

Your Turn: Take a More Precise Measurement

This time division approximation of the heart rate might seem a little vague and imprecise.

- ✓ Try pointing at the top of adjacent peaks with your mouse, and record the times with the Floating cursor.
- ✓ Subtract the smaller time from the larger time and use the result to recalculate the pulse rate.

You can also improve the precision by zooming in on each feature by adjusting the Horizontal dial to smaller values and then moving the screen around to find a more precise time of each event.

- ✓ Try it.

Comment [AL47]: To-do: Test this with new software.

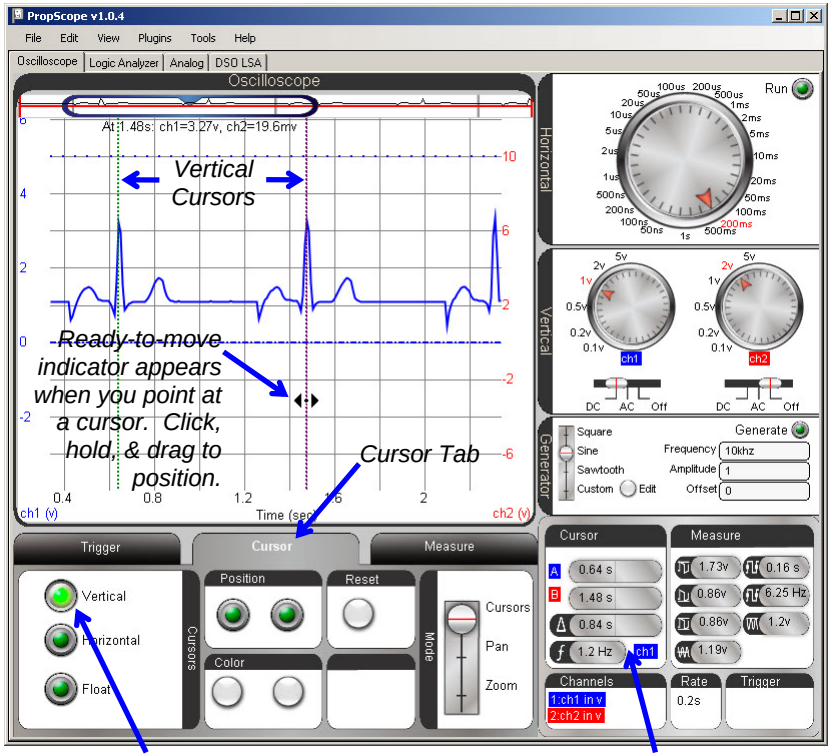
Use Vertical Cursors to Measure the Period

Oscilloscopes (including the PropScope) have tools called cursors to determine the time and voltage differences between points of interest on a waveform. Cursors are horizontal and vertical lines that can be positioned on the Oscilloscope display. A panel on the oscilloscope then displays the positions of the cursors along with voltage, time difference, and frequency calculations.

Figure 3-23 shows an example of the PropScope's vertical (time) cursors positioned over the heartbeat signal peaks. In the Cursor Display on the Oscilloscope View's lower-right, it shows both the time of each cursor's positions along with automated Δ and f measurements. The Δ measurement is pronounced delta and is shorthand for the time difference Δt "delta-t" between the vertical cursor lines. When applied to a signal that

repeats itself, Δ is also the signal's period T. The f measurement is 1 ÷ Δ and is often used to automate the f = 1 ÷ T calculation. Note that the Cursor measurements panel indicates the frequency is 1.2 Hz. Multiplied by 60 seconds/minute, we get 72 bpm.

Figure 3-23: Cursors Placed over Pulse Peaks to Measure Pulse Rate



Here is how to position cursors to take measurements like those shown in Figure 3-23:

- ✓ Navigate to the Cursor panel by clicking the Cursor tab below the Oscilloscope screen.

- ✓ In the Cursors tab, click the Vertical button to make the time cursors appear on the oscilloscope. They will appear as two vertical lines, one green and the purple.
- ✓ Click either the waveform or the CH1 ground line to select CH1 as the active channel in the Cursor Display.
- ✓ Point at one of the cursor lines with your mouse, and it will change to a double-arrow that points left and right. Click, hold, and move the cursor left or right to place it over one of the heartbeat signal peaks.
- ✓ Repeat with the other cursor and an adjacent heartbeat signal peak.
- ✓ Check the f measurement in the Cursor Display by the Oscilloscope screen's lower-right corner.

Your Turn: Measure Voltage Differences with Horizontal Cursors

Figure 3-24 shows horizontal cursors used to measure the voltage difference between the tops of the heartbeat signal's P and T wave peaks.

- ✓ Click the Horizontal button in the Cursor panel to activate the voltage cursors.
- ✓ Click the Vertical button deactivate the time cursors.
- ✓ Adjust the horizontal voltage cursors so that they measure the difference between the P and T peaks as shown in Figure 3-24.
- ✓ Check the voltage difference in the cursor display. It's near the lower-right corner of the Oscilloscope plot. It's the value that ends with V by the Δ symbol in the Cursor display.

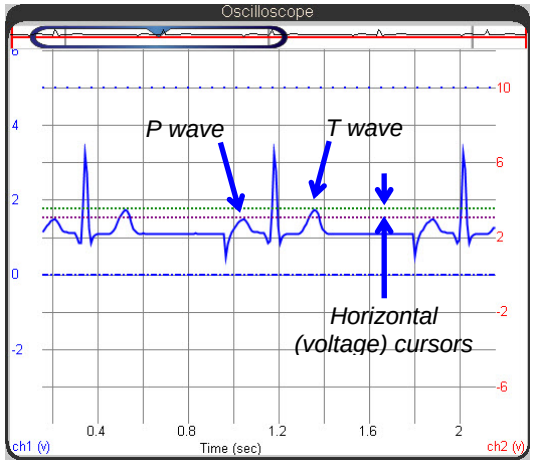


Figure 3-24
Voltage Cursors Measuring Difference between P and T waves