

- ✓ Measure the duration.
- ✓ Compare it to the duration of the FREQOUT signal on CH1.

### **Your Turn: Change the Pulse Duration**

What does infrared object detection with infrared have to do with pulse width modulation? The answer is that you can control the duration of the low pulse the IR detector sends by adjusting the FREQOUT command's duration. In other words, in FREQOUT Pin, Duration, Freq1, you can modulate the pulse width by changing the FREQOUT command's Duration argument. Try this:

4

- ✓ Change the FREQOUT command in TestLeftIrPair.bs2 to FREQOUT 8, 2, 38500.
- ✓ Place an object that will cause a detection in front of the IR LED and detector.
- ✓ Repeat the pulse width measurements, on the PropScope.
- ✓ See how changing the FREQOUT command's Duration argument results in a longer FREQOUT signal in CH1, which in turn causes the IR detector to send a 2 ms low pulse?

### **ACTIVITY #4: ADVANCED TOPIC: PULSES FOR TV REMOTE COMMUNICATION**

Figure 4-23 shows how a TV remote sends bursts of signals in the 38 to 40 kHz neighborhood to an IR detector. The durations of a series of these bursts are coded to communicate which button on the remote is pressed. The IR detector on your board converts these brief IR signal durations to brief low pulses, which are also called negative pulses. The BASIC Stamp can decode these pulses to figure out which button on the remote is pressed in a manner similar to the microcontrollers inside TVs and other entertainment system components. In this activity, you will use the both the PropScope and BASIC Stamp to examine these signals.

Figure 4-23: IR Remote Signals to IR Detector

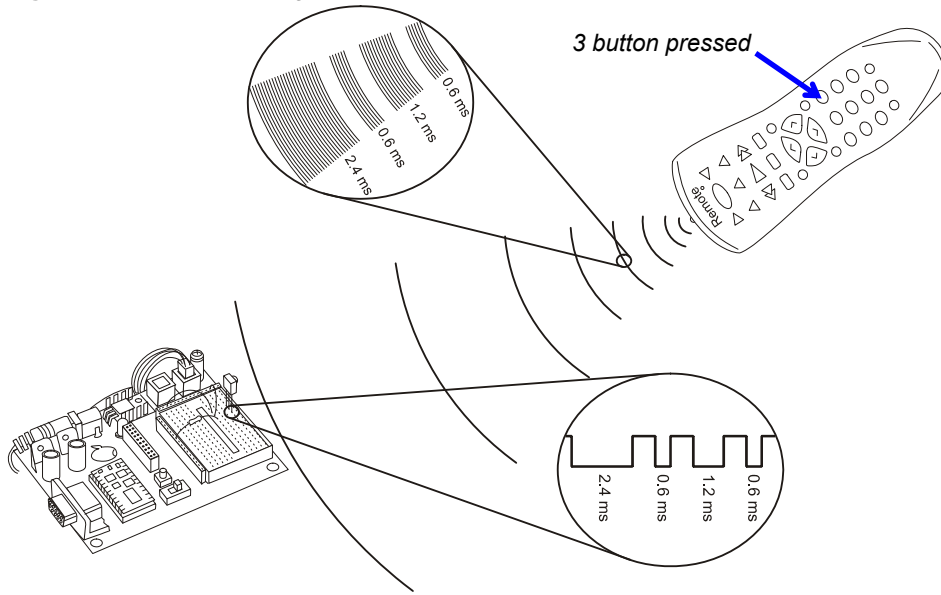
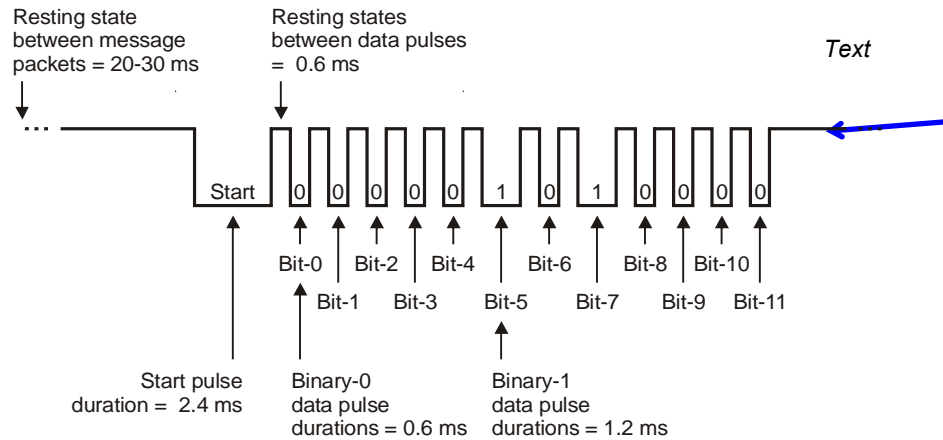


Figure 4-24 shows a timing diagram of the SONY TV remote protocol from the standpoint of the IR detector's output. After a 2.4 ms start pulse, a series of 12 low pulses follow. Each low pulse lasts for either 0.6 ms to indicate a binary 0, or 1.2 ms to indicate a binary 1. Like an 8 bit binary number can contain a number in the 0 to 255 range, a 12-bit binary number can contain a number in the 0 to 4095 range. So, the pulse width modulated signal in Figure 4-24 can be used to transmit 4096 different values.

**Figure 4-24: SONY TV Remote Protocol**

4

**Extra Parts, Equipment, and Setup**

You will use the IR detector and CH2 probe shown in Figure 4-18 and Figure 4-19, on pages 148 and 149, so there is no need to modify your circuit or probe connections. The “extra equipment” you will need is a universal remote.

- (1) Universal remote
- (1) Instruction sheet/booklet for the universal remote
- Compatible batteries

Although a universal TV remote is not included in the Understanding Signals kit, most households have one that can be configured to control a SONY TV. For those that don't, inexpensive universal remotes can also be purchased from local department stores. Check the package before you purchase a particular remote to make sure it can be configured to control a SONY TV. If the packaging says something like, "compatible with most/all major brands...", it should work with SONY TVs. These remotes are usually packaged with either a piece of paper or a booklet that explains how to configure it to be a SONY TV remote. The instructions typically involve:

- Looking up a number code for SONY TV in the remote's sheet/booklet
- Pressing and holding a certain remote button until an indicator light comes on
- Entering the number code into the remote's keypad



An **IR Remote Parts Kit** is available at [www.parallax.com](http://www.parallax.com). Just type 29122 into Search field and click Go. The remote in this kit is an example of one that costs under \$15 US.

The **IR Remote for the Boe-Bot Parts + Book** page (Item code 28139) has a link to a free .pdf tutorial and another to a zip file with lots of PBASIC code examples for BASIC Stamp IR Remote communication and control. These links are in the Downloads & Resources section near the bottom of the web page.

Here are example instructions for the TV remote in the IR Remote AppKit:

- ✓ Remove the battery compartment cover and determine how many and what kind of batteries to use (AA, AAA, etc).
- ✓ Load the battery compartment with new (or freshly charged rechargeable) batteries. Do not mix battery types.
- ✓ Find the TV setup codes section in the instruction sheet/booklet. Here are some examples of titles for that section: "Setup Codes for TV", "Setup Codes for Television", "TV Code List".
- ✓ Find the code for SONY from the TV code list, and make a note of it.
- ✓ Find the section that explains how to manually program a TV code into your remote. Here also, are examples of titles for that section: "Programming Your Remote", "To Manually Program Your Remote Control", "Programming for TV".
- ✓ Follow the instructions in the manual programming section for entering the SONY TV code into your remote.

### **IR Remote Test Code**

RecordAndDisplayPwm.bs2 measures the twelve pulses that follow the start pulse in Figure 4-24 and displays their durations in the Debug Terminal. The program uses a command called PULSIN, which is the inverse of PULSOUT. Instead of transmitting a pulse to another device, it measures a pulse transmitted by another device and stores the result in a variable. This result is a pulse measurement in terms of 2  $\mu$ s units. The Debug Terminal displays these results, and you can multiply each one by 2 to convert it from a 2  $\mu$ s unit measurement to a number of microseconds.

- ✓ Enter RecordAndDisplayPwm.bs2 and run it.



**Remove Sources of IR Interference.** Depending on the IR detector, sunlight from a nearby window could cause a lot of spurious signals. Certain fluorescent lights generate 38.5 kHz range inference too. For best results:

- ✓ Close the blinds.
- ✓ Turn off nearby fluorescent lights.

```
' IR Remote for the Boe-Bot - RecordAndDisplayPwm.bs2
' Measure all data pulses from a SONY IR remote set to control a TV.

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

time          VAR      Word(12)          ' SONY TV remote variables.
index         VAR      Nib                ' Display heading.

DEBUG "time ARRAY", CR,
      "PWM MEASUREMENTS", CR,
      "Element  Duration, 2-us", CR,
      "-----  -----"

DO                                                    ' Beginning of main loop.

  DO                                                  ' Wait for rest between messages.
    PULSIN 9, 1, time(0)
  LOOP UNTIL time(0) > 1000

  PULSIN 9, 0, time(0)                               ' Measure/store data pulses.
  PULSIN 9, 0, time(1)
  PULSIN 9, 0, time(2)
  PULSIN 9, 0, time(3)
  PULSIN 9, 0, time(4)
  PULSIN 9, 0, time(5)
  PULSIN 9, 0, time(6)
  PULSIN 9, 0, time(7)
  PULSIN 9, 0, time(8)
  PULSIN 9, 0, time(9)
  PULSIN 9, 0, time(10)
  PULSIN 9, 0, time(11)

  FOR index = 0 TO 11                               ' Display 12 pulse measurements.
    DEBUG CR$RXY, 0, 4 + index, "time(", DEC index, ")",
          CR$RXY, 9, 4 + index, DEC time(index), CLREOL
  NEXT

LOOP                                                  ' Repeat main loop.
```

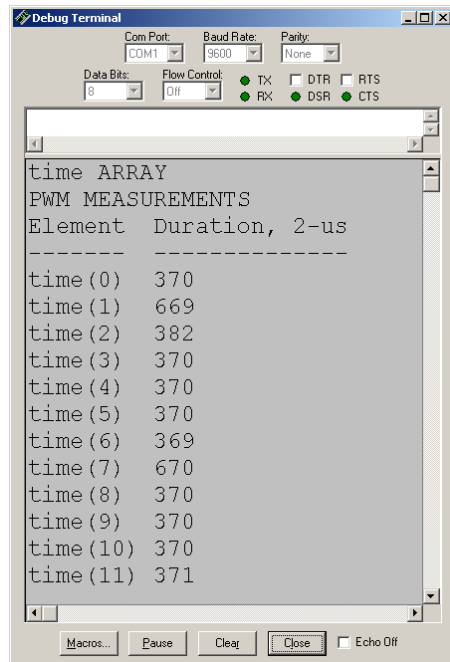


For more information on how the example programs in this activity work, download and consult the IR Remote for the Boe-Bot pdf tutorial from [www.parallax.com](http://www.parallax.com).

### IR Remote Test Measurements with the BASIC Stamp

Figure 4-25 shows the BASIC Stamp's pulse measurements while the remote's 3 button is pressed.

- ✓ Point your remote at the IR detector and press and release the 3 button.



**Figure 4-25**  
BASIC Stamp Displays  
Pulse Measurements  
from an IR Detector

*...that's receiving infrared  
signals from a SONY TV  
remote with the 3 button  
pressed.*

Table 4-1 shows approximate measurements of the first four pulses as buttons 1 through 9 are pressed.


Pulse Variable	Button								
	1	2	3	4	5	6	7	8	9
time(0)	370	670	370	670	370	670	370	670	370
time(1)	370	370	670	670	370	370	670	670	370
time(2)	370	370	370	370	670	670	670	670	370
time(3)	370	370	370	370	370	370	370	370	670

4

For this remote, the 670 values are binary-1s and the 370 values are binary-0s, and time(3) through time(0) store pulse durations that represent the lowest four binary digits in the 12-digit binary number the IR remote transmits. Table 4-2 shows a sequence of the first five remote buttons, their pulse duration patterns, the binary values they represent, and the decimal equivalent values. We'll call these values remote codes.

- ✓ Test your remote buttons 6, 7, 8, 9, and 0, and fill in the rest of the table.
- ✓ Your values may differ, but there should be a pattern of values that are closer to 600 that are binary-1s and values that are closer to 300 that are binary-0s.

Button	time(3)	time(2)	time(1)	time(0)	Binary Value	Decimal value
1	370	370	370	370	0000	0
2	370	370	370	670	0001	1
3	370	370	670	370	0010	2
4	370	370	670	670	0011	3
5	370	670	370	370	0100	4
6						
7						
8						
9						
0						



**Decimal Digits vs. Binary Digits**

Four decimal digits, each of which could be a value from 0 to 9, represent the number of:

Thousands Hundreds Tens Ones

Four binary digits, each of which could be a value from 0 to 1, represent the number of:

Eights Fours Twos Ones

Knowing this, you can convert values from binary to decimal. For example you could convert binary-1010 to decimal knowing that there's an 8 and a 2, which adds up to 10.

$$(1 \times 8) + (0 \times 4) + (1 \times 2) + (0 \times 1) = 10$$

### Variations in SONY TV Remote Communication Pulses

Servo control pulses have to be fairly precise because even small variations in pulse durations will be visible as different servo horn positions. In contrast, TV remote communication protocols tend to leave more room for variations in the pulse durations that represent binary-1s and binary-0s. The microcontroller inside the TV that receives and deciphers pulses from the IR detector can decide whether a given pulse falls into a range of durations that represents a binary-1, or a range of durations that represents a binary-0. Since universal remotes made by different manufacturers may vary somewhat in their IR signaling times (that get converted to pulses by the IR detector), leaving room in the protocol for variations in the pulse durations that represents binary-1s and 0s is an important design feature.

With possible variations in mind, your remote's pulse duration values that represent binary 1s and 0s might differ from the values in Figure 4-25. For that matter, those values differ considerably from the values in the timing diagram in Figure 4-24 on page 155. That timing diagram indicates that binary zeros last 0.6 ms, and binary ones last 1.2 ms. However, in Figure 4-25, the binary-1s appear last for  $670 \times 2\mu\text{s} = 1340 \mu\text{s}$ . That's 1.34 ms, not 1.2 ms. Likewise,  $370 \times 2 \mu\text{s} = 740 \mu\text{s}$ , which is 0.74 ms, not 0.6 ms. How will you be able to tell the difference between your remote's binary-1 and binary-0 pulses? Look for longer duration, binary-1 pulses that are closer to 1.2 ms, and shorter duration binary-0 pulses that are closer to 0.6 ms.

### IR Remote Test Measurements with the PropScope

Figure 4-26 shows a first view of the IR detector's output when the 3 button on the remote is pressed. The Horizontal dial selection was chosen using the rule of thumb that we want see two cycles of a signal in the oscilloscope screen for a first look. In this case,



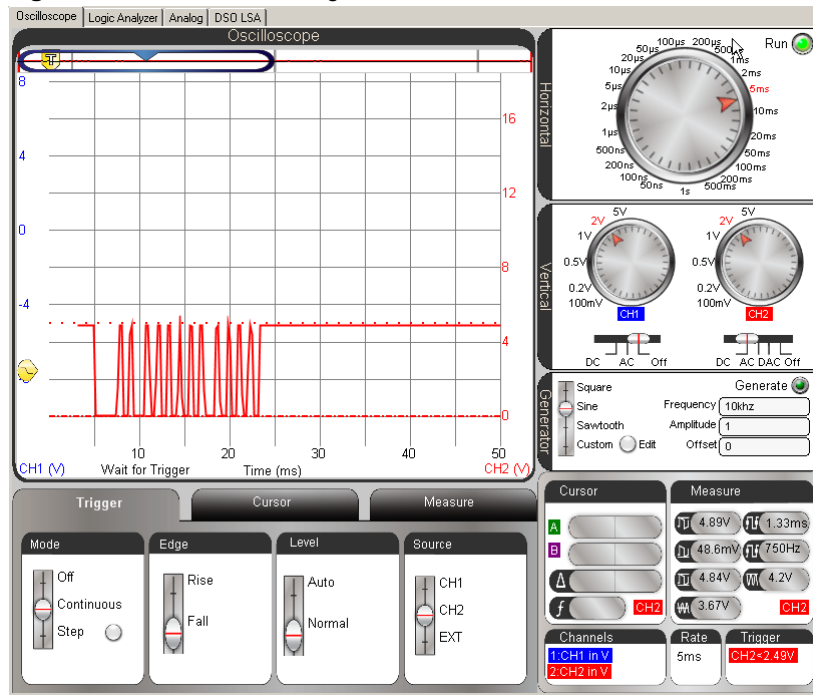
we'll call a "cycle" one infrared message. Looking at the timing diagram in Figure 4-24 on page 155, a rough approximation of the time an IR message lasts would be:

$$\begin{aligned}
 &2.4\text{ ms} = 1 \times 2.4\text{ ms} \text{ (start pulse)} \\
 &+ 7.2\text{ ms} = 12 \times 0.6\text{ ms} \text{ (highs between low pulses)} \\
 &+ 6.0\text{ ms} = 10 \times 0.6\text{ ms} \text{ (binary } \underline{0} \text{ pulses)} \\
 &+ 2.4\text{ ms} = 2 \times 1.2\text{ ms} \text{ (binary } \underline{1} \text{ pulses)} \\
 &= 18\text{ ms}
 \end{aligned}$$

4

Assuming a message fits in a 20 ms window, multiply by 2 for enough room for two cycles, which is 40 ms. Then, divide by 10 for Horizontal dial setting in units with a result of 4 ms/div. The closest value is 5 ms/div.

Figure 4-26: SONY IR Message – Initial View




**IMPORTANT:** The timing diagram also showed that the Start pulse begins with a negative edge, so it will be important to set the Trigger tab's Edge switch to Fall. There are many negative (falling) edges in the signal, but this will increase the likelihood that the first negative edge triggers the display, like it did in Figure 4-26.

Also, the Trigger tab's Level switch should be set to Normal instead of Auto. Normal means you will manually adjust the trigger voltage level. Auto triggering uses the average voltage of the signal in the oscilloscope screen, but there may be long periods of time between TV remote messages where the signal will stay at 5 V. So, the average voltage will be 5 V, as will the automatic trigger level. As the IR detector sends a series of negative pulses to the BASIC Stamp I/O pin, there will be lots of 0 V low signals. So it would be better to manually set the trigger voltage level to 2.5 V instead of letting it hover around 5 V and hoping it catches the signal transitions that we want to trigger the display.

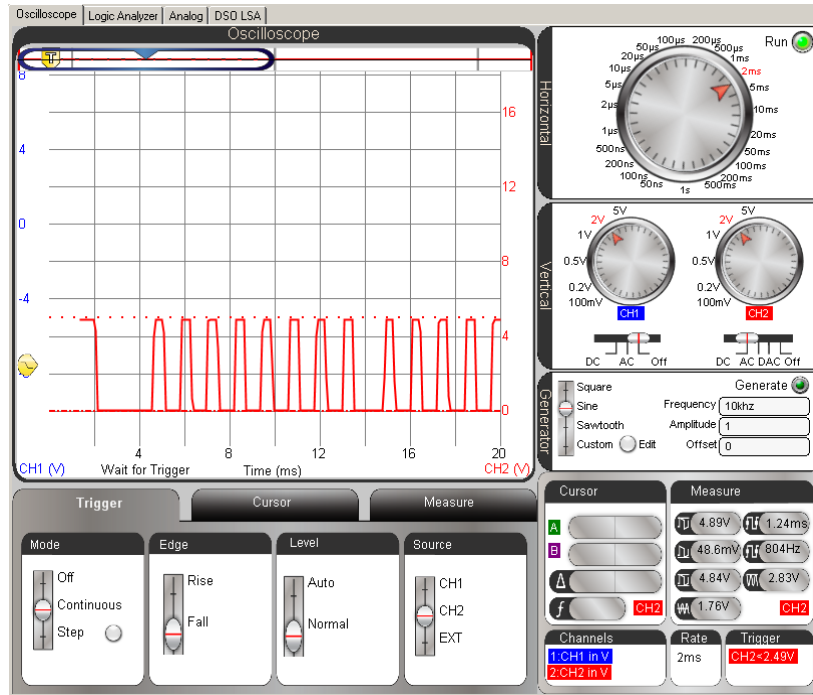
**Comment [AL55]:** To-do: Change settings to switches elsewhere in book.

- ✓ Configure your PropScope's Horizontal dial, Vertical dials and Coupling switches, and Trigger tab settings as shown in Figure 4-26.
- ✓ Make sure to set the Trigger Edge switch to Fall.
- ✓ Also, make sure to set the trigger Level switch to Normal.
- ✓ Then, set the Trigger Voltage control along the left side of the Oscilloscope screen to about 2.5 V.
- ✓ Make sure the Trigger Time control is set to the first time division line. (Careful here, in previous activities, we always used the second time division line.)
- ✓ Press/release a number button on your remote while pointing it at the IR detector.
- ✓ Verify that a signal similar to the one in the Figure 4-26 Oscilloscope screen appears.
- ✓ Verify that other number buttons result in similar signals.

Figure 4-27 shows the a closer view of the timing with the remote's 1 number button pressed/released. You may have to press/release the button a couple of times to get a display that includes the wider start pulse at the far left. 

- ✓ Adjust your PropScope's Horizontal dial to 2 ms/div, and press the 1 button a few times until you get a display similar to Figure 4-27.

Figure 4-27: IR Message (1 button) with Decreased Horizontal Scale



4

With the Horizontal dial set to 2 ms, you'll be able to see the low pulse durations change as you press/release different numbers on the remote. Figure 4-28 shows patterns that represent the 2, 3, and 4 buttons.

- ✓ Remember to point the remote at the IR detector before you press its buttons.
- ✓ Press/release the 2 button on your remote a few times, until the start pulse's negative edge, lines up with the 1<sup>st</sup> time division. The pattern of low pulses should resemble the signal labeled 2 Button in Figure 4-28.
- ✓ Repeat for the 3 and 4 buttons, and compare to Figure 4-28.
- ✓ Keep an eye on the three negative pulses that follow the start pulse. They are the ones that carry the binary values 001, 010, and 011, which are the binary remote codes for the 2, 3, and 4 buttons.

Figure 4-28: Pulse Patterns for the 2, 3, and 4 Buttons

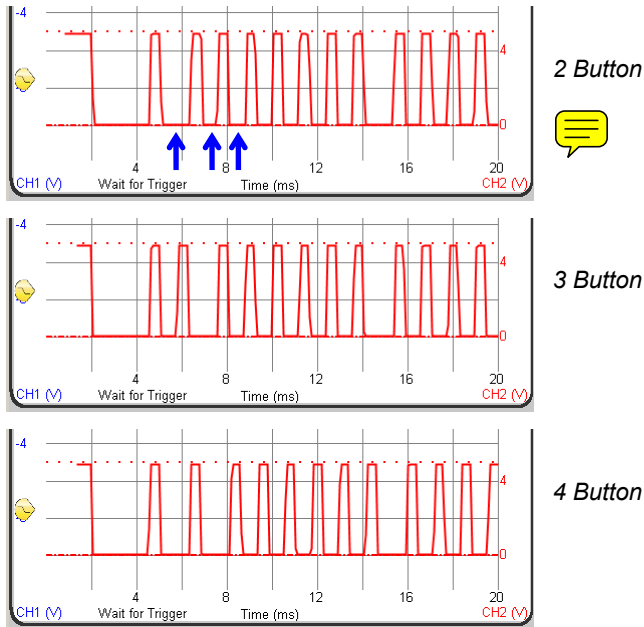
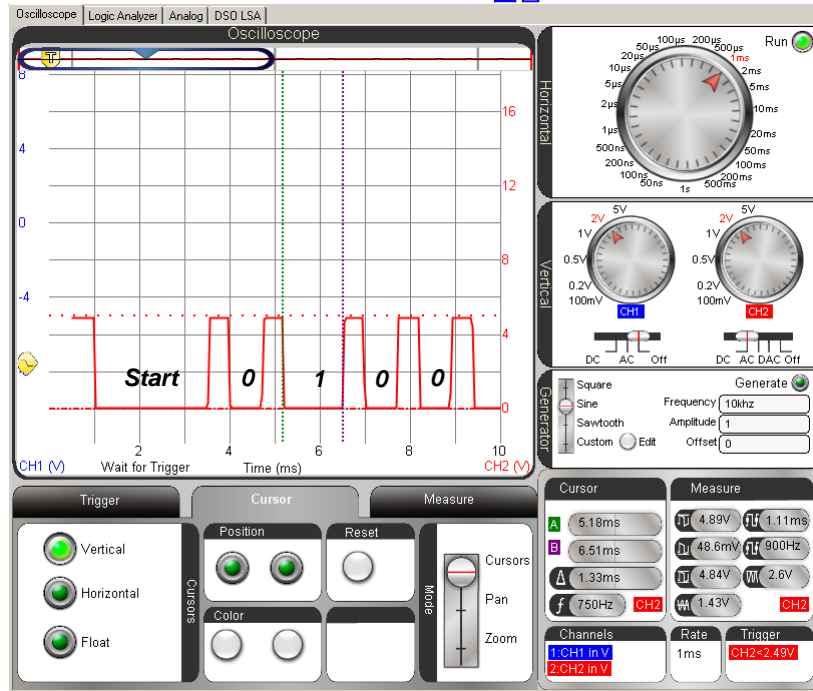


Figure 4-29 shows the Oscilloscope zoomed in another time division setting, with the Horizontal dial set to 1 ms/div. This is a good time setting for measuring low pulse times. The Vertical time cursors in the figure are measuring a binary 1 pulse.

- ✓ Adjust the Horizontal dial to 1 ms/div.
- ✓ Press the 3 button a few times until the Oscilloscope view resembles Figure 4-29.
- ✓ Use the Vertical time cursors to measure the Start, 0, and 1 pulse durations for your remote.
- ✓ How do they compare to your BASIC Stamp measurements?

Figure 4-29: Cursor Measurement of a Binary 1 Pulse

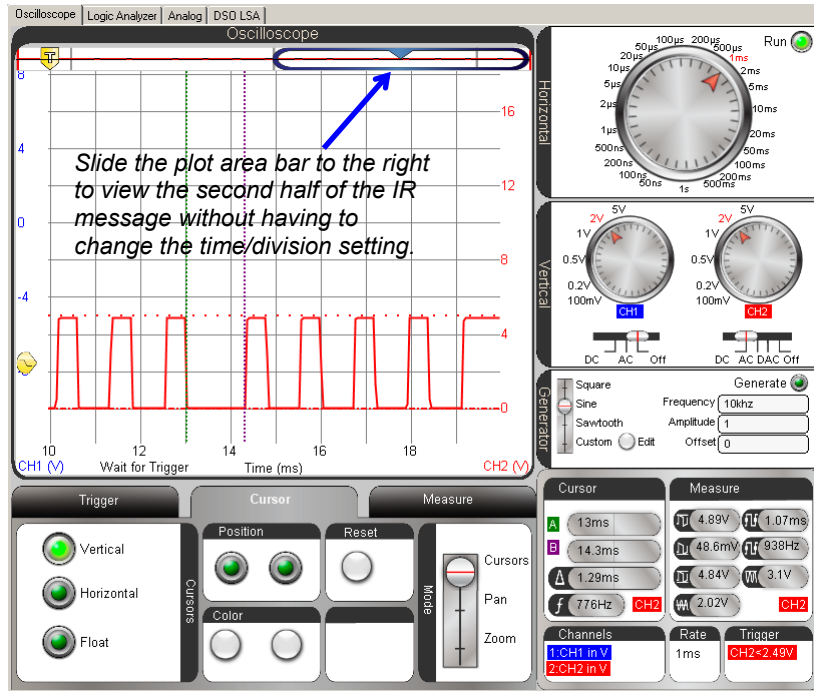


4

With the 1 ms/div Horizontal setting, only half the pulses are visible in Figure 4-29. Figure 4-30 shows an example of the rest of the plot viewed by sliding the plot area bar to the right.

- ✓ Slide the Plot Area bar to the far right of the Preview Bar for a closer inspection of the second half of the IR remote's message.

Figure 4-30: Shift Oscilloscope Screen right to See the Rest of the Message



### Your Turn: IR Remote Buttons Program

Figure 4-31 shows the Debug Terminal after the remote's 3 button was pressed with IrRemoteButtons.bs2 running. IrRemoteButtons.bs2 is an excerpt from IR Remote for the Boe-Bot.

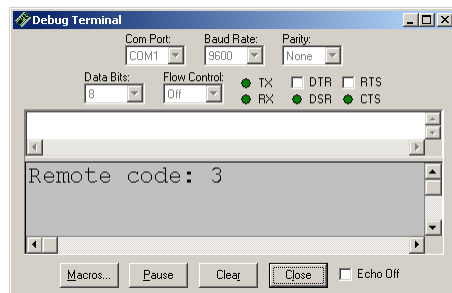


Figure 4-31  
Debug Terminal Remote  
Code Display

..with  
IrRemoteButtons.bs2  
running.

- ✓ Download the IR Remote Book Source Code (.zip) from the IR Remote for the Boe-Bot Kit and Text page at [www.parallax.com](http://www.parallax.com).
- ✓ Open IrRemoteButtons.bs2 with the BASIC Stamp Editor and load it into the BASIC Stamp.
- ✓ Study the codes for buttons like 0 through 9, Power, VOL+, VOL-, CH+, CH-, etc.

```

' -----[ Title ]-----
' IR Remote for the Boe-Bot - IrRemoteButtons.bs2
' Capture and store button codes sent by a universal remote configured to
' control a SONY TV.

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

' -----[ I/O Definitions ]-----

' SONY TV IR remote declaration - input received from IR detector
IrDet          PIN      9

' -----[ Constants ]-----

' SONY TV IR remote constants for non-keypad buttons
Enter          CON      11
ChUp           CON      16
ChDn          CON      17
VolUp         CON      18
VolDn         CON      19
Power         CON      21

' -----[ Variables ]-----

' SONY TV IR remote variables
irPulse        VAR      Word
remoteCode     VAR      Byte

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

DO
  GOSUB Get_Ir_Remote_Code
  DEBUG CLS, "Remote code: ", DEC remoteCode
  PAUSE 100
LOOP

' -----[ Subroutine - Get_Ir_Remote_Code ]-----

```

```
' SONY TV IR remote subroutine loads the remote code into the
' remoteCode variable.

Get_Ir_Remote_Code:

remoteCode = 0                ' Clear all bits in remoteCode

' Wait for resting state between messages to end.

DO
  RCTIME IrDet, 1, irPulse
LOOP UNTIL irPulse > 1000

' Measure start pulse.  If out of range, then retry at Get_Ir_Remote_Code.

RCTIME 9, 0, irPulse
IF irPulse > 1125 OR irPulse < 675 THEN GOTO Get_Ir_Remote_Code

' Get data bit pulses.

RCTIME IrDet, 0, irPulse      ' Measure pulse
IF irPulse > 300 THEN remoteCode.BIT0 = 1 ' Set (or leave clear) bit-0
RCTIME IrDet, 0, irPulse      ' Measure next pulse
IF irPulse > 300 THEN remoteCode.BIT1 = 1 ' Set (or leave clear) bit-1
RCTIME IrDet, 0, irPulse      ' etc
IF irPulse > 300 THEN remoteCode.BIT2 = 1
RCTIME IrDet, 0, irPulse
IF irPulse > 300 THEN remoteCode.BIT3 = 1
RCTIME IrDet, 0, irPulse
IF irPulse > 300 THEN remoteCode.BIT4 = 1
RCTIME IrDet, 0, irPulse
IF irPulse > 300 THEN remoteCode.BIT5 = 1
RCTIME IrDet, 0, irPulse
IF irPulse > 300 THEN remoteCode.BIT6 = 1

' Adjust remoteCode so that keypad keys correspond to the value
' it stores.

IF (remoteCode < 10) THEN remoteCode = remoteCode + 1
IF (remoteCode = 10) THEN remoteCode = 0

RETURN
```

## SUMMARY

This chapter introduced pulse width modulation (PWM), and used the PropScope to study PWM signals in several example applications. The first application involved positive pulses sent to a servo's control input to set its output horn's position. Servo control pulse durations can range from 0.5 ms through 2.5 ms, which maps



(approximately) to 0 through 180° servo horn positions. Servos are designed to receive pulses at a rate of about 50 Hz, but the frequency of the pulses does not have to be precise. In contrast, the positive pulse durations do have to be precise for good servo horn position control.

The PWM command for D/A conversion uses a duty cycle modulated signal to set voltage across the capacitor in an RC circuit. The key attribute of duty cycle modulation is that a signal that stays high for a certain proportion of its cycle time. Infrared object detection and TV remote control demonstrates how PWM can be used for communication. An infrared detector converts brief periods of 40 kHz range infrared signals to brief low signals that contain binary values.

4

### Questions

1. What are two common uses of PWM?
2. How do you think the W in PWM got its name?
3. If a servo control pulse is wider than a previous pulse, what does that instruct a servo to do?
4. Which argument in PULSOUT Pin Duration determines pulse width?
5. What ratio does PWM establish?
6. How does the ratio determine the DAC voltage?
7. What common household device contains an IR detector?
8. How does pulse width relate to the FREQOUT command's Duration argument in infrared object detection?
9. If pulses are negative, what should the trigger edge setting be?
10. What PBASIC command measures pulse duration?
11. What PropScope tool can you use to measure pulse width?
12. What does a SONY remote pulse in the 1.2 ms neighborhood represent?

### Exercises

1. Calculate the pulse width for positioning a servo's horn at 120°.
2. Calculate the pulse width for positioning a servo's horn at 150°.
3. Calculate the servo horn angle that will result from PULSOUT 14, 650 signals.
4. Calculate a good initial Horizontal setting for viewing at least two servo pulses if the PAUSE is reduced from 20 to 7 ms.
5. Predict the voltage of a 60%, 5 V duty cycle signal.
6.  $t_{\text{HIGH}} = 1 \text{ ms}$ ,  $t_{\text{CYCLE}} = 2 \text{ ms}$ . Calculate the duty cycle.
7. Predict the maximum time it should take to capture the first 8 bits of a SONY IR message based on its timing diagram.

- Predict the total time a SONY IR Message should take if it contains 6 binary-1s.

### **Projects**

- Measure the maximum and minimum servo frequencies with PAUSE 20 in a loop with a PULSOUT command. The only thing in the loop that varies is the PULSOUT command. Assume that PULSOUT 14, 300 is the minimum pulse width and PULSOUT 14, 1200 is the maximum.
- Use cursors to measure the total time for an IR message when the 1 button on the remote is pressed. To make it more challenging, set the Horizontal dial so that the message fills almost the entire plot preview.

### **Solutions**

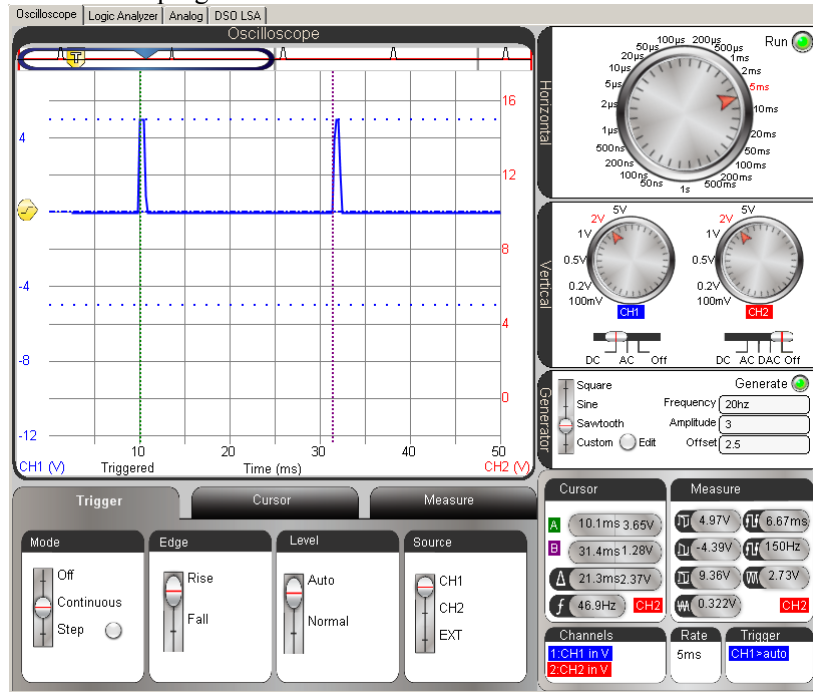
- Q1. Motor control and communication.
- Q2. Probably because the time a pulse lasts translates to its width in an oscilloscope display.
- Q3. Rotate its horn to a position that's counterclockwise from the previous position.
- Q4. The Duration argument.
- Q5. Duty cycle, the ratio of high time to cycle time.
- Q6. It's the percent of the 5 V signal.
- Q7. Television.
- Q8. If the IR detector senses the reflected infrared, its pulse stays low for the duration of the FREQOUT command.
- Q9. Fall.
- Q10. PULSIN
- Q11. The cursors.
- Q12. A binary 1.

- E1.  $t_{\text{PULSE}} = 2 \text{ ms} \times 120^\circ \div 180^\circ + 0.5 \text{ ms} = 1.833 \dots \text{ ms}$ .
- E2.  $t_{\text{PULSE}} = 2 \text{ ms} \times 150^\circ \div 180^\circ + 0.5 \text{ ms} = 2.166 \dots \text{ ms}$ .
- E3.  $(650-250)/1000 \times 180^\circ = 72^\circ$
- E4. Assume pulses are 1.5 ms + 7 ms PAUSE → period T = 8.5 ms. 2 cycles → 2T = 17 ms. 17 ms/10 divisions = 1.7 ms/division, which is close to 2 ms, so use 2 ms/div.
- E5.  $60\% \times 5 \text{ V} = 3 \text{ V}$
- E6.  $1/2 \times 100\% = 50\%$
- E7.  $2.4 \text{ ms} + (1.2 \text{ ms} \times 8) + (0.6 \times 8 \text{ ms}) = 16.8 \text{ ms}$ . In practice, this could be longer because IR remotes don't have to be precise.

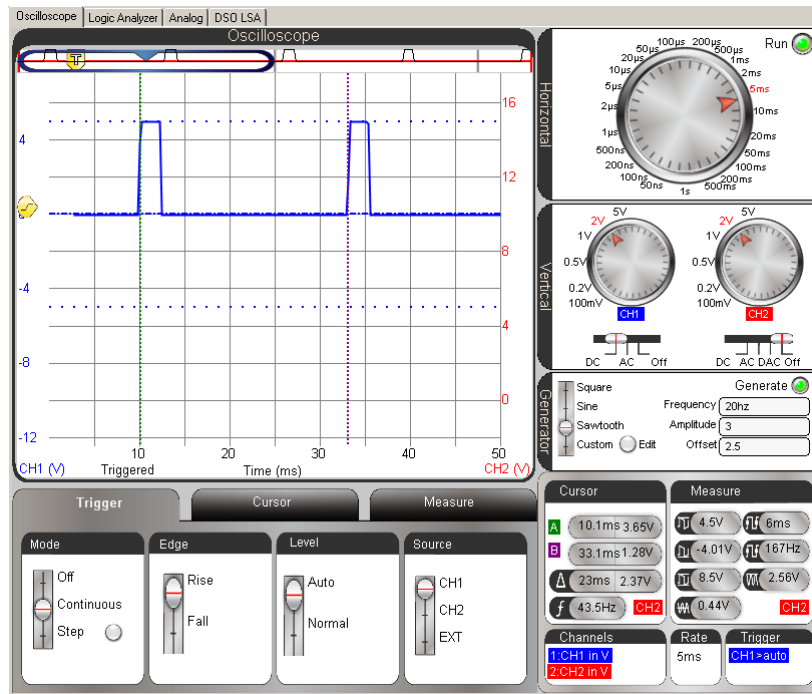
- $2.4\text{ ms} = 1 \times 2.4\text{ ms}$  (start pulse)  
 $+ 7.2\text{ ms} = 12 \times 0.6\text{ ms}$  (high between low pulses)  
 E8.  $+ 3.6\text{ ms} = 6 \times 0.6\text{ ms}$  (binary - 0 pulses)  
 $+ 7.2\text{ ms} = 6 \times 1.2\text{ ms}$  (binary - 1 pulses)  
 $= 20.4\text{ ms}$

4

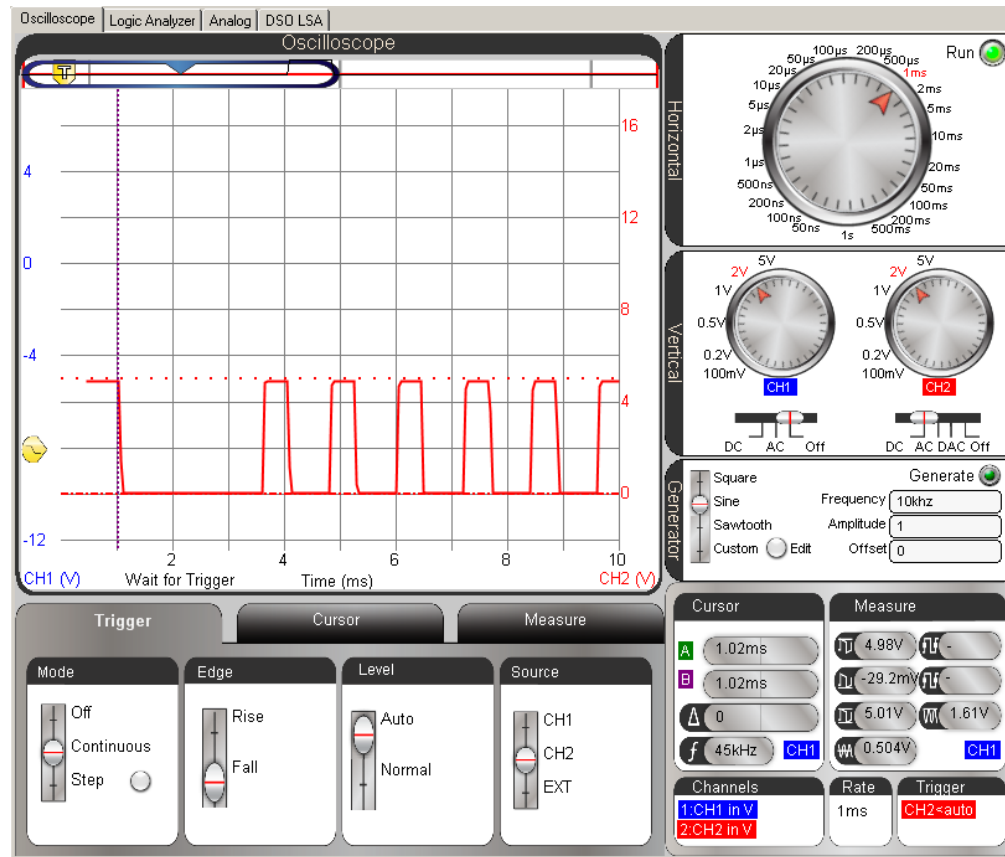
- P1. In ServoCenter.bs2, change PULSOUT 14, 750 to PULSOUT 14, 350, and load the modified program. Answer: 21.3 ms from  $\Delta$  field with cursors.



- Change PULSOUT 14, 350 to PULSOUT 14, 1150, and load the modified program. Answer: 23 ms from  $\Delta$  field with cursors.



P2. Position either cursor at the falling edge. Measure the position of the cursor you positioned (cursor in the example below).



Shift the Plot Position Bar all the way to the right, and then place the cursor at last rising edge and check the cursor again. The total time is the difference between the two cursors. In this example, that's  $18.6 \text{ ms} - 1.02 \text{ ms} = 17.58 \text{ ms}$ .

