PARALLAX INC

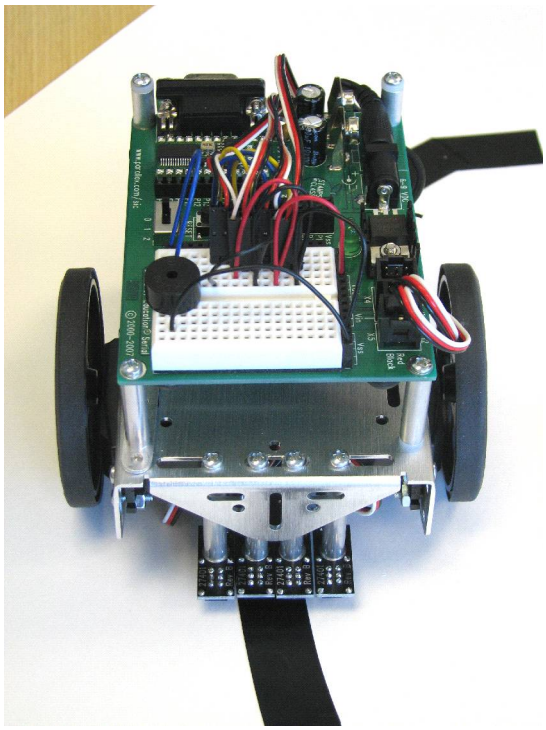
Web Site: www.parallax.com
Forums: forums.parallax.com
Sales: sales@parallax.com
Technical: support@parallax.com

Office: (916) 624-8333
Fax: (916) 624-8003
Sales: (888) 512-1024
Tech Support: (888) 997-8267

QTI Line Follower AppKit for the Boe-Bot (#28108)

Boe-Bot Line Following with QTIs

The QTI sensor is a close-proximity infrared emitter and receiver pair mounted on a tiny PCB. It can be used as an analog sensor to differentiate between different levels of infrared reflectivity. It can also be used as a purely digital device that returns a 1 when it detects a black line or a 0 if it detects a white background. An array of four QTI sensors used as digital devices makes an effective and flexible line-follower for the Boe-Bot robot.



The QTI positions are adjustable for different sizes and types of lines. This activity demonstrates how the QTIs can be used for digital line following on a simple 3/4-inch wide electrical tape course with a white background. For an in-depth look at how the QTI sensors function, and for the complete QTI Line Follower source code, see the Stamps in Class "Mini Projects" sticky-thread at the top of the Stamps in Class forum at on http://forums.parallax.com.

For this activity, you will need to supply your own:

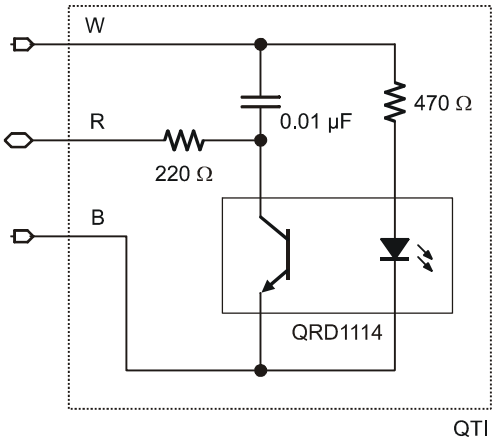
- Built and tested Boe-Bot robot
- #2 Philips-head screwdriver
- Black ¾-inch electrical tape
- White poster board

Kit Contents

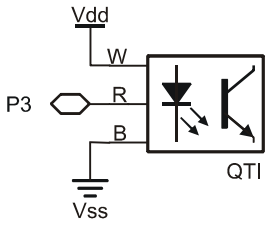
Part #	Description	Qty
805-00001	10-inch 3-lead extension cable	4
800-00016	3-inch jumper wires (bag of 10)	2
150-01030	Resistor - 10 kΩ	4
555-27401	QTI sensor	4
451-00303	3-pin male-male header	4
710-00007	7/8-inch screw, pan head, 4-40	4
700-00060	Standoff, round, 1-inch, 4-40	4
713-00007	Spacer, round, 1/2 inch	4
700-00002	3/8-inch screw, pan head, 4-40	4
700-00015	Washer, nylon, #4	4

A Closer Look at the QTI

The QTI module is designed for close proximity infrared (IR) detection. Take a look at the small square black box just above the QTI label. It's nested below the capacitor and between the two resistors. That's a QRD1114 reflective object sensor. There's an infrared diode behind its clear window and an infrared transistor behind its black window. When the infrared emitted by the diode reflects off a surface and returns to the black window, it strikes the infrared transistor's base, causing it to conduct current. The more infrared incident on the transistor's base, the more current it conducts.

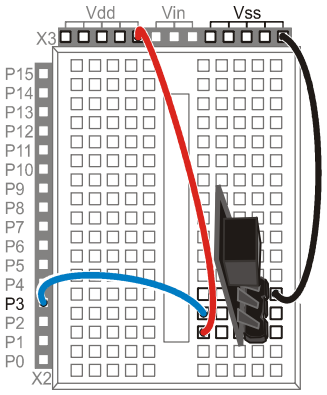


When used as an analog sensor, the QTI can detect shades of gray on paper and distances over a short range if the light in the room remains constant. With this circuit, you can set P3 high and then test it with **RCTIME** to measure how long it takes the capacitor to discharge through the IR transistor. Since the IR transistor conducts more or less current depending on how much IR it receives, the **RCTIME** measurement can give you an indication of distance or shade of gray.



Vdd = 5 VCD
Vss = 0 V (ground)

```
' AnalogQti.bs2
' {$STAMP BS2}
' {$PBASIC 2.5}

time VAR word

DO
  HIGH 3
  RCTIME 3, 1, time
  DEBUG CLS, ? time
  PAUSE 100
LOOP
```



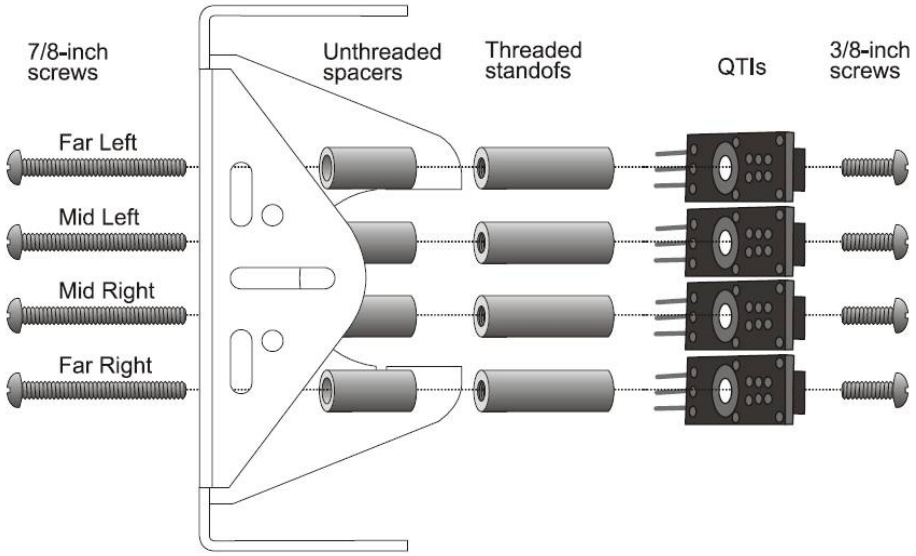
If all you want to know is whether a line is black or white, the QTI can also be converted to a digital sensor. This is how the array of four QTI sensors will be used later in our Boe-Bot line-following application.

When W is connected to Vdd and B is connected to Vss, the R terminal's voltage will drop below 1.4 V when the IR transistor sees infrared reflected from the IR LED. When the IR LED's signal is mostly absorbed by a black surface, the voltage at R goes above 1.4 V. Since the BASIC Stamp interprets any voltage above 1.4 V as 1 and any voltage below 1.4 V as 0, this circuit gives us a quick and easy way to detect a black line on a white background.

Mounting the QTIs

1. Match the components in your kit to the Kit Contents above to make sure all pieces are present. If anything is missing, contact Parallax Tech Support.
2. Referring to the picture on the next page, insert the 7/8-inch screws through the top of the Boe-Bot chassis, at the three slots near the front. Two screws will go through the center slot, and one screw each in the right and left slots.
3. On the underside of the chassis, slip a 1/2-inch unthreaded spacer on each screw, followed by a 1-inch threaded standoff.

4. Attach a QTI sensor to the other end of each threaded standoff, using a 3/8-inch screw. The sensors should be facing downwards, and the 3-pin headers on each sensor should be pointing towards the back of the chassis.
5. If necessary, slightly loosen the 7/8-inch screws and adjust the position of the QTI sensors so that they are closely positioned edge to edge.
6. Tighten all connections securely.

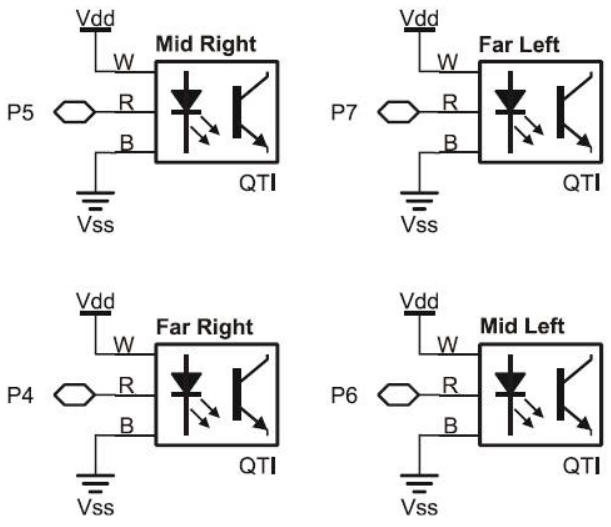


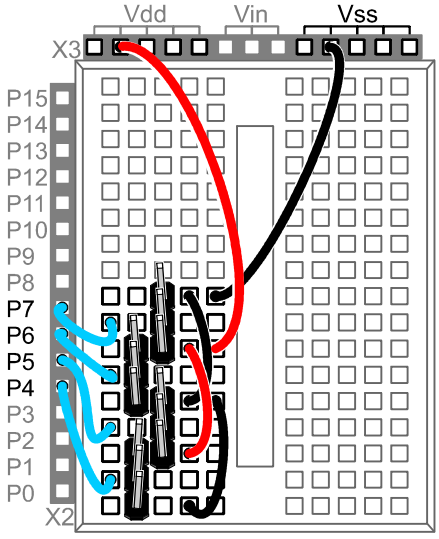
Building the Sensing Circuits

Each QTI B pin is tied to Vss (ground) and each W pin is connected to Vdd (5 V). R pin connects to a BASIC Stamp I/O pin:

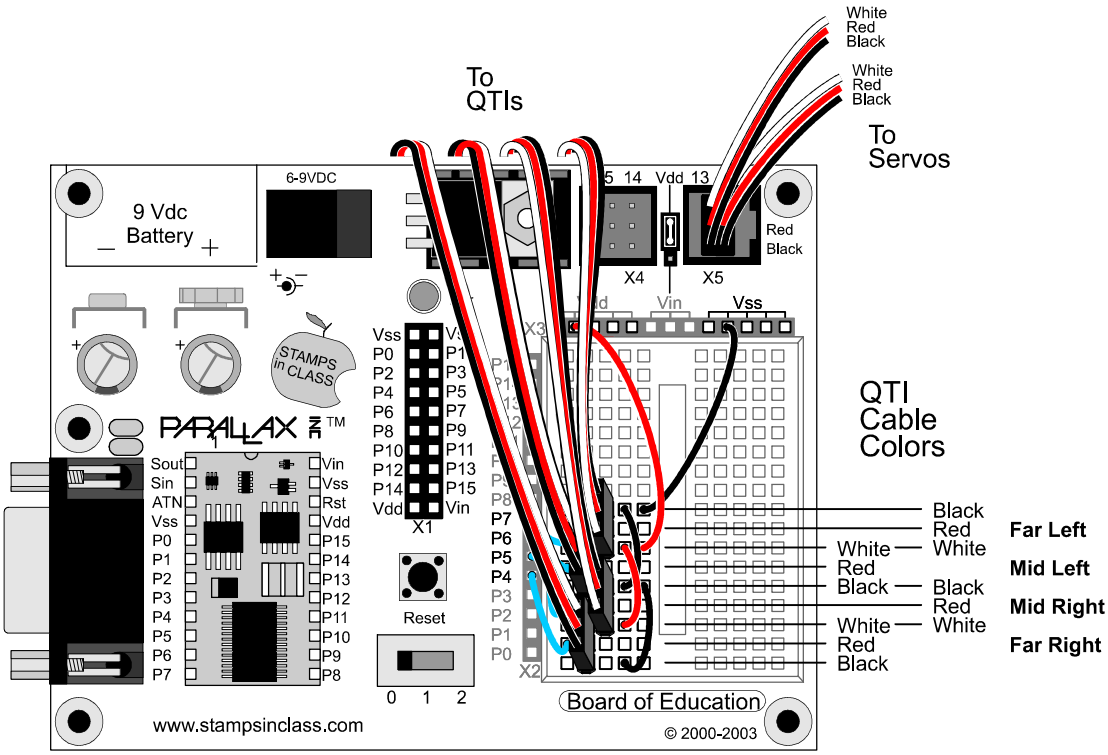
- Far right to P4
- Mid right to P5
- Mid left to P6
- Mid right to P7

√ Use the schematic and the wiring diagram below to build the circuits for the three-pin headers. Although there are many ways this circuit can be built, the setup shown below is useful because it's thrifty with breadboard real-estate. Also, if you are using a HomeWork Board with your Boe-Bot, it won't interfere with the servo connections.





√ Use the figure below to connect the servo extension cables to the male-male headers that were plugged into the breadboard above. The other end of each cable is plugged into a QTI. Be careful when you make these connections, and pay close attention to the wire colors listed in the figure. Notice that the far left and mid left cables have white wires that plug onto header pins in the same breadboard row. The same applies to the mid right and far right QTI cables. Instead of common white wires, the mid left and mid right QTI cables share a pair of black wires.



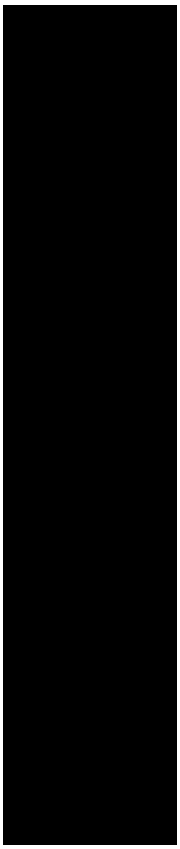
Testing for Line Detection

It's a good idea to test all the sensors with the Debug Terminal before taking the Boe-Bot for a spin on the line following course.

√ Affix a few inches of 3/4-inch wide electrical tape to a white sheet of paper, or use the test line to the right of the program listing on the next page.
√ Enter and run CheckQtiSubroutine.bs2
√ Place the far left QTI directly over the electrical tape (and the other QTIs over white background).
√ The Debug Terminal should read 1000.
√ Repeat this process until all QTIs are tested.

If you had problems with two QTIs sensing 1 when only one of them should have, try adjusting the standoffs so that the QTIs are further apart. Not so far that you can get 0000 when the line is between two QTIs though! On the other hand, if only one QTI sensed 1 when the stripe was between two of them, they may need to be positioned closer together. Otherwise, your Boe-Bot is ready for line following.

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

' CheckQtiSubroutine.bs2
' Displays QTI sensor states. 0 means white surface, 1 means
' black.

qtis VAR Nib                  ' qti black/white states

OUTB = %1111                  ' Set OUTB bits to 1

DEBUG CRSRX, 8, "FMMF", CR,   ' Display bit positions
      CRSRX, 8, "LLRR", CR

DO                            ' Main DO...LOOP
  GOSUB Check_Qtis            ' Get QTI states
  DEBUG BIN4 ? qtis, CRSRUP   ' Display QTI states
  PAUSE 100                   ' 1/10 s delay
LOOP

Check_Qtis:
  ' Result -> qtis variable. 0 means white surface, 1 means
  ' black surface.

  DIRB = %1111                  ' P7..P4 -> output
  PAUSE 0                       ' Delay = 230 us
  DIRB = %0000                  ' P7..P4 -> input
  PAUSE 0                       ' Delay = 230 us
  qtis = INB ' Store QTI outputs in INB

  RETURN
```



The BASIC Stamp has a variety of parallel I/O control features that make it possible to perform operations on groups of I/O pins. CheckQtiSubroutine.bs2 uses the BASIC Stamp's **DIRB** and **OUTB** variables to control the directions and output states of the I/O pins P7, P6, P5, and P4. When these I/O pins are set to input with **DIRB = %0000**, the program also takes a snapshot of the binary states these I/O pins sense as a result of the four QTIs' R pin voltages with the command **qtis = INB**.

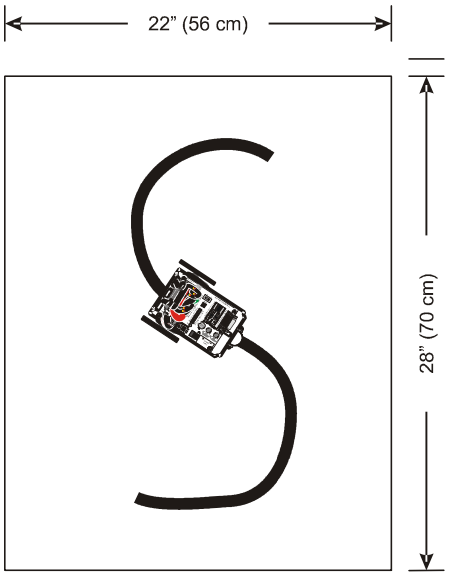
Simple Line-Following

LineFollowWithCheckQtis.bs2 is designed to start following a line as soon as you place a QTI over the electrical tape. It will stop line following as soon as it runs out of electrical tape. Start with an easy course, like a large S shape as shown below. If the QTIs passed the "Testing for Line Detection" tests, it should navigate the course with ease.

- √ Enter and run LineFollowWithCheckQtis.bs2.
- √ Place the Boe-Bot on the course so that in straddles the line with the mid left and mid right QTIs over the electrical tape.

If your QTIs' cables are reversed, the Boe-Bot will appear to try to jump off the line at the first opportunity. Otherwise, it should faithfully follow the line until it reaches the end of the tape.

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

' LineFollowWithCheckQtis.bs2
' Navigates based on values acquired with the
' Check_Qtis subroutine.



qtis VAR Nib                ' black/white states

OUTB = %1111                ' Set OUTB bits to 1



DO                          ' Main DO...LOOP

  GOSUB Check_Qtis          ' Get QTI states

  SELECT qtis               ' Control servo
                            ' speeds/directions
    CASE %1000              ' Rotate right
      PULSOUT 13, 650
      PULSOUT 12, 650
    CASE %1100              ' Pivot right
      PULSOUT 13, 750
      PULSOUT 12, 650
    CASE %0100              ' Curve right
      PULSOUT 13, 800
      PULSOUT 12, 650
    CASE %0110              ' Straight ahead
      PULSOUT 13, 850
      PULSOUT 12, 650
    CASE %0010              ' Curve left
      PULSOUT 13, 850
      PULSOUT 12, 700
    CASE %0011              ' Pivot left
      PULSOUT 13, 850
      PULSOUT 12, 750
    CASE %0001              ' Rotate left
      PULSOUT 13, 850
      PULSOUT 12, 850
    CASE ELSE               ' Do nothing
      PAUSE 3
  ENDSELECT

LOOP

Check_Qtis:

' Result -> qtis variable.
' 0 means white surface
' 1 means black surface.

  DIRB = %1111              ' P7..P4 -> output
  PAUSE 0                   ' Delay = 230 us
  DIRB = %0000              ' P7..P4 -> input
  PAUSE 0                   ' Delay = 230 us
  qtis = INB                ' Store QTI outputs
                            ' in INB
  RETURN
```



Keep in mind that this is a bare-bones line following example. Making it robust and versatile is up to you. The Boe-Bot starts maneuvers when it sees one of the cases listed in the **SELECT...CASE** statement. This is convenient for a simple demonstration program because the Boe-Bot doesn't go anywhere when it is on an all white or all black surface. It only starts navigation when it detects a line. You can add extra code, especially in the form of **CASE** statements to deal with special situations and more complex courses.

- √ Try other courses to test the limits of the program.
- √ Try modifying the program to solve courses that the unmodified program could not solve.
- √ Modify the program so that it smoothes out the Boe-Bot's responses to changes in the line's direction. Your code should take steps toward a maximum speed each time it detects that a given pattern is detected.
- √ Detect intersections and make random turns.
- √ Challenge a friend, or set up a Boe-Bot line following competition.

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Parallax:
28108