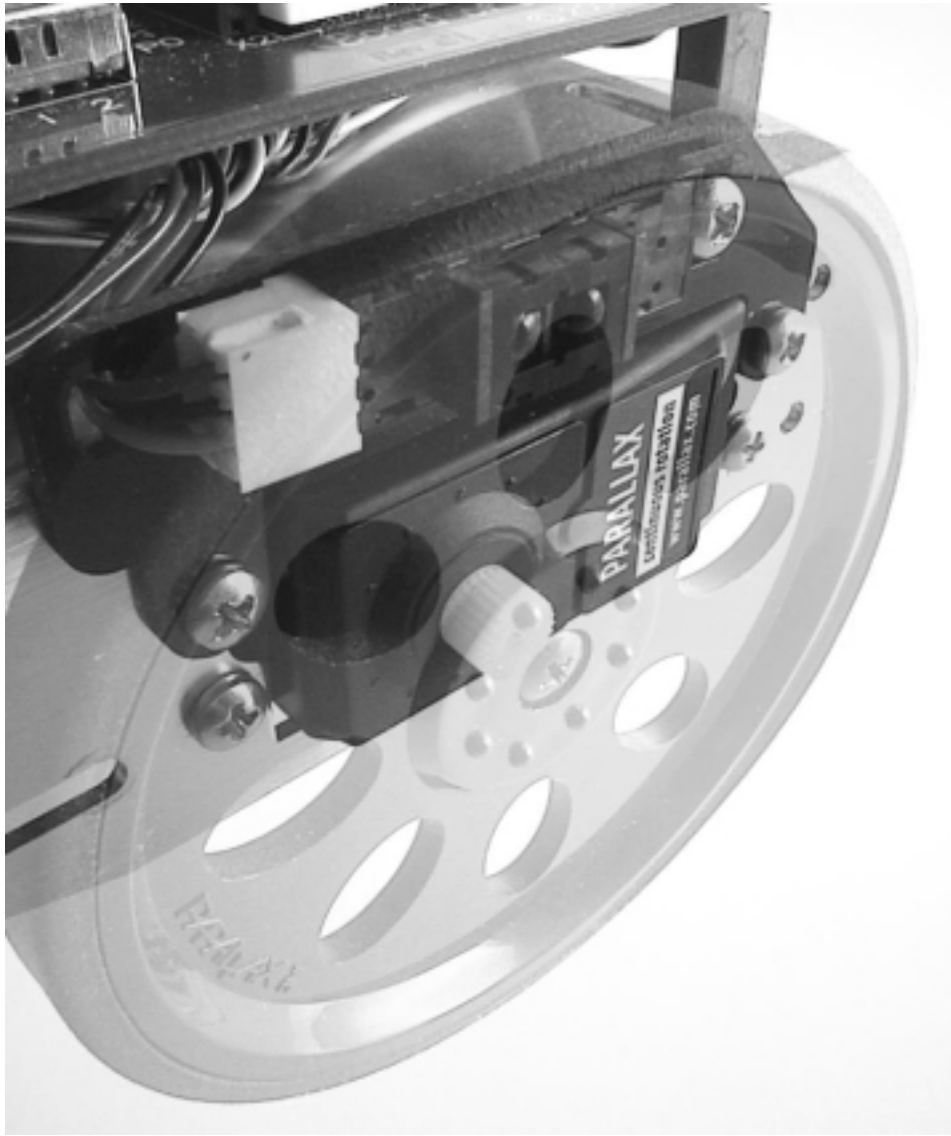
Sample Program (Star_Pattern.BS2)

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

'Program to traverse a 5-pointed star. Straight sections are about 25" long.
'Angle of each turn is 144-degrees. Use it to test the encoders. If the left one
'fails to function, the Boe-Bot will just continue going in a straight line
'without stopping. If the right one fails, it will continue to circle in the turn.
'Ideally, it will come to rest at the point and direction in which it started.
'But it's uncalibrated, so that's somewhat unlikely!

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

RIGHT     CON 0         'Constants defining direction subscripts.
LEFT      CON 1

SenseR    PIN 10        'Righthand encoder input.
SenseL    PIN 11        'Lefthand encoder input. (MUST be SenseR + 1.)

MotorR    PIN 12        'Righthand motor output.
MotorL    PIN 13        'Lefthand motor output. (MUST be MotorR + 1.)

Sense     CON SenseR    'Base address for encoders.
Motor     CON MotorR    'Base address for motors.

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

Prev      VAR Bit(2)    'Previous readings from encoders.
New       VAR Bit(2)    'Current readings from encoders.
Side      VAR Bit       'Side index (LEFT or RIGHT).

Counter   VAR Word(2)   'Pulse counters for encoders.
i         VAR Byte      'FOR/NEXT index

'---------[Program begins here.]--------------------------------------------

PAUSE 3000                          'Time to remove finger from reset button...
FOR i = 1 TO 5
  Counter(LEFT) = 0                 'Initialize LEFT counter.
  DO WHILE (Counter(LEFT) < 50)
    PULSOUT MotorL, 850             'Pulse the servos to go straight
    PULSOUT MotorR, 650             'by 50 pulse edges (about 25").
    GOSUB Update                    'Update the encoder counts.
    PAUSE 20                        '20 ms pulse gap.
  LOOP
  PAUSE 500                         'Wait 1/2 second before turn.
  Counter(RIGHT) = 0                'Initialize RIGHT counter.
  DO WHILE (Counter(RIGHT) < 20)
    PULSOUT MotorR, 650             'Pulse the RIGHT servo to turn a 40% circle.
    GOSUB Update                    'Update the encoder counts.
    PAUSE 20                        '20 ms pulse gap.
  LOOP
  PAUSE 500                         'Wait 1/2 second before next side.
NEXT
END

'---------[Subroutines]-----------------------------------------------------

Update:                             'Update pulse counts from the encoders.
                                    'Just call it often enough to catch all the
                                    'changes.
  FOR Side = RIGHT TO LEFT                  'For both encoders...
    New(Side) = INS.LOWBIT(Sense + Side)    'Read new encoder value.
    IF (New(Side) ^ Prev(Side)) THEN        'Different from last value?
      Prev(Side) = New(Side)                '  Yes: Update with new value.
      Counter(Side) = Counter(Side) + 1     '       Update counter.
    ENDIF
  NEXT
  RETURN
```

Boe-Bot Digital Encoder Kit #28107

PARALLAX inc

www.parallax.com
support@parallax.com



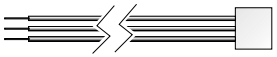
599 Menlo Drive, Suite 100 • Rocklin, CA 95765 • USA
888-512-1024 (toll-free) • 916-624-8333 • 916-624-8003 (fax)

Document revision: 2005 Aug 23

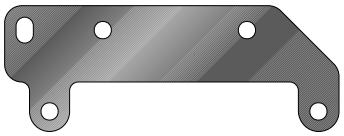
Parts List

2 ea. Reflective Sensors

2 ea. Connecting Cable



2 ea. Mounting Plates



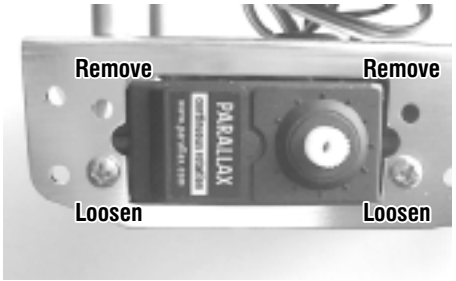
2 ea. #4-40 x 1/4" Panhead Machine Screw

2 ea. #4-40 Hex Nut

2 ea. 10K 1/4W Resistor

Assembly

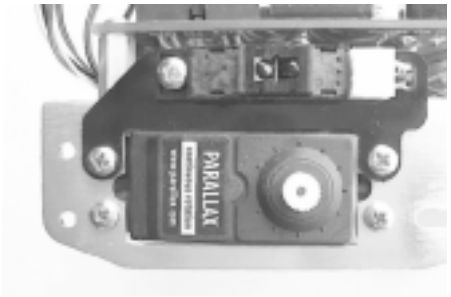
To assemble the Bot-Bot Wheel Encoder Kit, you will need a small Phillips screwdriver and a pair of needlenose pliers. Begin by removing both wheels from the Boe-Bot. Next, remove the top two screws (the ones nearest the crease in the chassis) holding each servo. Loosen the remaining servo mounting screws:



Plug a connecting cable into each sensor, as shown:

Next, take one of the mounting plates, and thread the wires from one of the cables through the oblong hole. Insert the sensor's alignment pin into the mating mounting plate hole. Secure the sensor to the mounting plate with a screw through the other set of holes and a nut on the back. Repeat for the other sensor, but with the opposite orientation. When finished you will have two mounted sensors that are mirrors of each other:

Using the servo screws removed in the first step, mount each sensor to the chassis so the lenses are near to the servo shaft. Do not tighten the screws yet. See photo:

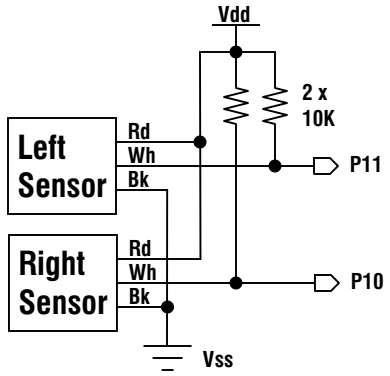


Viewing the servos from the chassis bottom, adjust their positions so that their shafts align as closely as possible. Either center both in their respective cutouts, or push them both toward (not away from) the sensor connector. Just make sure they're the same. Now tighten all eight servo screws.

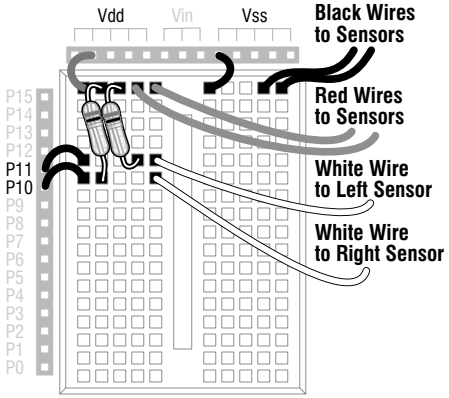
The sensor wires may now be brought out from under the front of the printed circuit board and inserted into the prototyping area. The wires are assigned as follows:

Red: **Vdd**
Black: **Vss**
White: **Signal (open collector)**

The encoders should be wired according to the schematic on the next page. **Note:** If your wires are red, black, and **yellow**, contact Parallax for a different wiring diagram.



A typical implementation of this schematic is shown below:

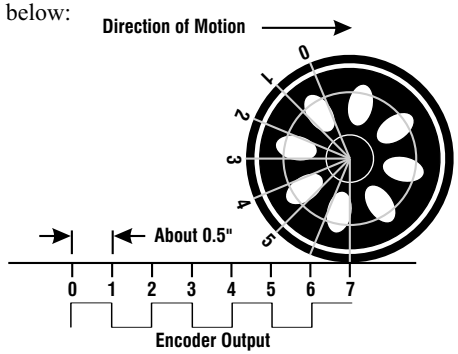


Operation

The sensors emit infrared light and look for its return from a reflective surface. They are calibrated for optimal sensing of surfaces a few millimeters away. The Boe-Bot's wheels, even though they are black, reflect sufficient IR to cause the sensors to respond. When a sensor "sees" part of a wheel, it pulls its output low. When it's looking through a hole, its output floats, and the pullup resistor pulls it high.

Because the sensors emit and detect only modulated IR (at about 7.8KHz) they are relatively insensitive to ambient light. Be aware, though, that some fluorescent fixtures may also emit light at this frequency and could interfere with their operation.

As a Boe-Bot wheel turns, the sensor will see an alternating pattern of hole - no hole - hole - no hole, etc. Its output will be a square wave whose frequency corresponds to the speed of rotation. If the Boe-Bot is rolling along the floor, each edge of the square wave will mark an increment of travel slightly more than 1/2 inch (1.27 cm), as shown in the illustration below:



By tracking changes in the encoders' outputs, it's possible for a BASIC Stamp program to tell how far the Boe-Bot has traveled. Notice that the encoders themselves do not tell the Stamp which direction the wheels are turning — only when and how far. But if the Stamp program is driving the wheel servos, it knows which direction each wheel is turning and can apply this additional information along with the encoder outputs. In the most sophisticated applications, it is possible not only to keep track of the Bot-Bot's position and direction, but also to coordinate the rotations of the two wheels, using the encoders as feedback, to obtain any desired motion contour.

Be aware, though, that wheel encoders are never perfect. Uncertainties in the effective wheel diameters can lead to position errors. Further uncertainties in effective wheel spacing during turns can result in direction errors. And even small position and direction errors have a way of accumulating quickly if not periodically corrected using external references. To obtain the best accuracy, always run on a hard, flat, smooth surface, such as a vinyl or hardwood floor — never on carpet.

The following sample program uses feedback from the encoders to traverse a star-shaped pattern on the floor. It is uncalibrated so is therefore unlikely to return exactly to its starting point. But try it anyway to see how close it gets. More complicated examples, including coordinated motion, X-Y position and direction odometry, calibration techniques, and the theory behind them, may be downloaded from the Parallax website at:

www.parallax.com/detail.asp?product_id=28107