Chapter #1: Detect Distance with the Ping)))$^{(TM)}$ Ultrasonic Sensor

WHAT IS THE PING))) SENSOR?

The Ping))) sensor is a device you can use with the BASIC Stamp to measure how far away an object is. With a range of 3 centimeters to 3.3 meters, it's a shoe-in for any number of robotics and automation projects. It's also remarkably accurate, easily detecting an object's distance down to the half centimeter.

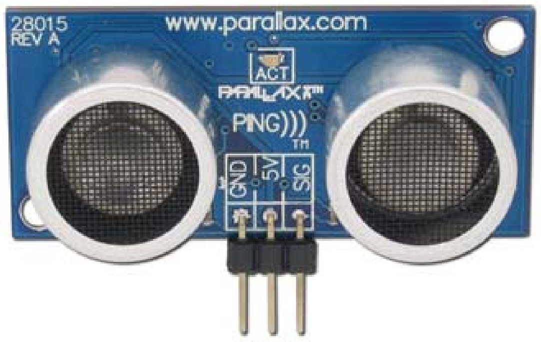
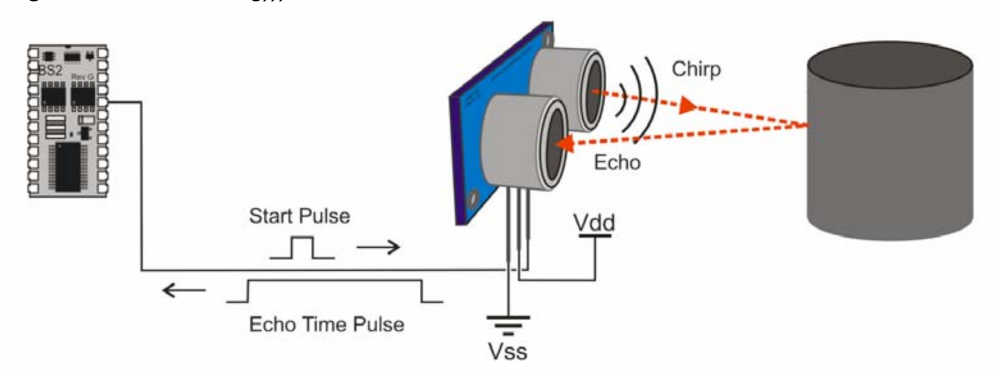


Figure 1
The Ping))) Sensor

HOW DOES THE PING))) SENSOR WORK?

Figure 2 shows how the Ping))) sensor sends a brief chirp with its ultrasonic speaker and makes it possible for the BASIC Stamp to measure the time it takes the echo to return to its ultrasonic microphone. The BASIC Stamp starts by sending the Ping))) sensor a pulse to start the measurement. Then, the Ping))) sensor waits long enough for the BASIC Stamp program to start a **`PULSIN`** command. At the same time the Ping))) sensor chirps its 40 kHz tone, it sends a high signal to the BASIC Stamp. When the Ping))) sensor detects the echo with its ultrasonic microphone, it changes that high signal back to low. The BASIC Stamp's **`PULSIN`** command stores how long the high signal from the Ping))) sensor lasted in a variable. The time measurement is how long it took sound to travel to the object and back. With this measurement, you can then use the speed of sound in air to make your program calculate the object's distance in centimeters, inches, feet, etc...

Figure 2 - How the Ping))) Sensor Works



The Ping))) sensor's chirps are not audible because 40 kHz is ultrasonic.

What we consider sound is our inner ear's ability to detect the variations in air pressure caused by vibration. The rate of these variations determines the pitch of the tone. Higher frequency tones result in higher pitch sounds and lower frequency tones result in lower pitch tones.

Most people can hear tones that range from 20 Hz, which is very low pitch, to 20 kHz, which is very high pitch. Subsonic is sound with frequencies below 20 Hz, and ultrasonic is sound with frequencies above 20 kHz. Since the Ping))) sensor's chirps are at 40 kHz, they are definitely ultrasonic, and not audible.

ACTIVITY #1: MEASURING ECHO TIME

In this activity, you will test the Ping))) sensor and verify that it gives you echo time measurements that correspond to an object's distance. You will also modify the example program to convert these times into centimeter measurements.

Parts Required

All you'll need is a Ping))) sensor and three jumper wires to make it work. The Ping))) sensor has protection against programming mistakes (and wiring mistakes) built-in, so there's no need to use a 220 Ω resistor between P15 and the Ping))) sensor's SIG terminal.

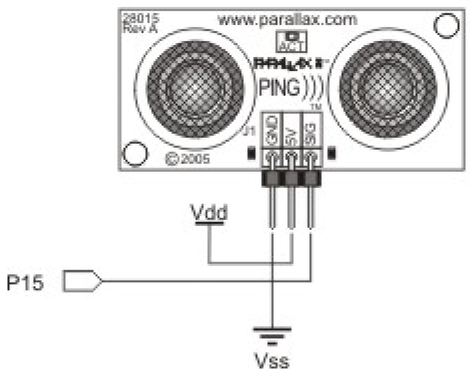
(1) Ping))) Ultrasonic Distance Sensor
(3) Jumper Wires

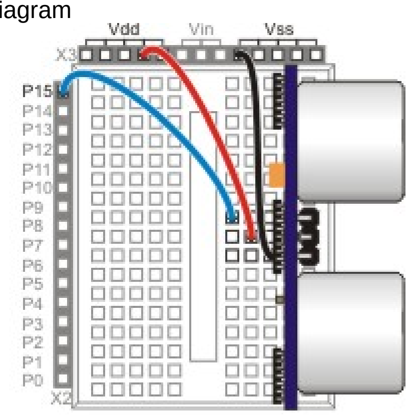
Ping))) Sensor Circuit

Figure 3 shows a schematic and wiring diagram you can use to test the Ping))) sensor.

√ Build the circuit.

Figure 3 - Ping))) Sensor Schematic and Wiring Diagram





Testing the Ping))) Sensor

As mentioned earlier, the Ping))) sensor needs a start pulse from the BASIC Stamp to start its measurement. A pulse to P15 that lasts 10 µs (**`PULSOUT 15, 5`**) is easily detected by the Ping))) sensor, and it only takes a small amount of time for the BASIC Stamp to send. A **`PULSIN`** command that stores the duration of the Ping))) sensor's echo pulse has to come immediately after the **`PULSOUT`** command. The result the **`PULSIN`** command stores is the round trip time for the Ping))) sensor's chirp to get to the object, reflect and return.

Example Program - PingTest.bs2

You can test this next program by measuring the distances of a few close-up objects. For close up measurements, the Ping))) sensor only needs to be roughly Boe-Bot height above your working surface (8 to 10 cm). However, if you are measuring objects that are more than a half a meter away, make sure to keep your Ping))) sensor about half a meter or more above the floor.

√ Place your Board of Education with the Ping))) sensor circuit on something to keep it at least 8 cm above the table surface.

- √ Place an object (like a water bottle, box, or paper target) 15 cm from the front of the Ping))) sensor.
- √ Enter, save, and run PingTest.bs2.
- √ The Debug Terminal should start reporting a value in the neighborhood of 450. Values of 438 to 466 mean the distance is between 15 and 16 cm.
- √ Move the target to a distance of 30 cm from the Ping))) sensor and verify that the value of the **`time`** variable doubled.
- √ Point your Ping))) sensor at a variety of near and far objects and observe the time measurements.
- √ Multiply your measurements by 0.03434 to convert to centimeter measurements, and verify that the measurements are correct.

```
' PingTest.bs2

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

time VAR Word

DO

  PULSOUT 15, 5
  PULSIN 15, 1, time
  DEBUG HOME, "time = ", DEC5 time
  PAUSE 100

LOOP
```

Your Turn - Displaying Centimeter Measurements

The next activity will introduce how to derive constants like 0.03434 for converting the echo time measurements to centimeters and other units. But first, let's look at how the PBASIC **`**`** operator makes it possible to multiply the **`time`** variable by a value like 0.03434. To convert 0.03434 to a value the **`**`** operator can use, multiply it by 65536, and use whatever's to the left of the decimal point. Since 0.03434 × 65536 = 2250.5, we'll use 2251 with the **`**`** operator for the time to centimeter conversion. Here's the conversion statement with the constant we just figured along with a **`DEBUG`** command to display the centimeter value.

```
  time = time ** 2251
  DEBUG CR, "Distance = ", DEC4 time, " cm"
```

√ Save PingTest.bs2 as PingCentimeters.bs2.

√ Add the two new lines of code to the program's **`DO...LOOP`** between the **`DEBUG`** and **`PAUSE`** commands. When you're done, the **`DO...LOOP`** should look like this:

```
DO

  PULSOUT 15, 5
  PULSIN 15, 1, time
  DEBUG HOME, "time = ", DEC5 time
  time = time ** 2251
  DEBUG CR, "Distance = ", DEC4 time, " cm"
  PAUSE 100

LOOP
```

√ Run your modified program and verify that the program correctly displays both the echo time and centimeter measurements.