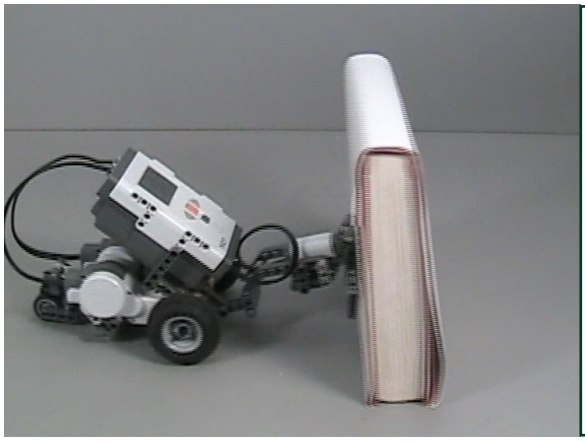
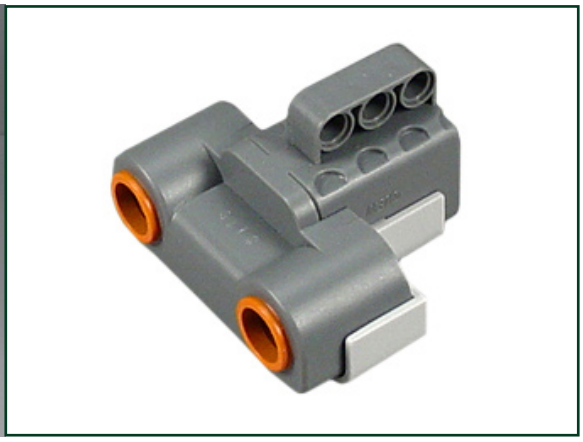
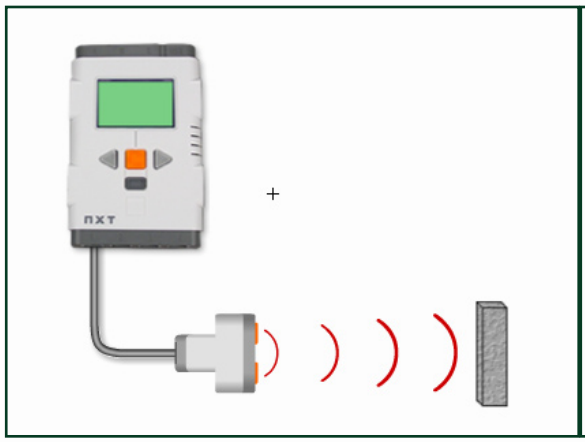
Wall Detection A Sonic Sojourn

Robots are precise, reliable, intelligent machines, but only when they are programmed to both sense and respond appropriately. Using a sensor which can only detect an obstacle by contact has drawbacks. You would rather not have to bump into something to know it's there, and neither would your robot.

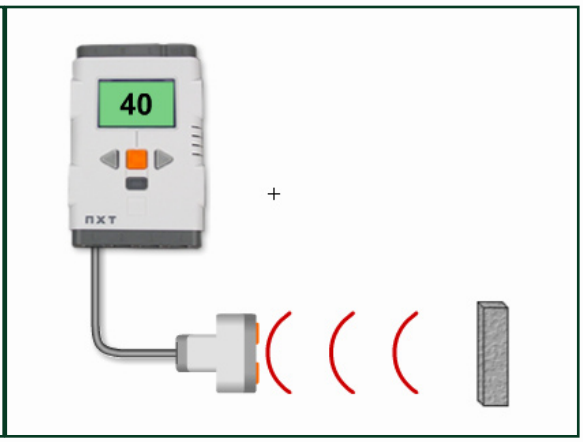




Above right is an Ultrasonic Sensor. Using the same physical principle that a bat or a submarine uses to find its way around, the Ultrasonic Sensor measures distances using sound. It then tells the robot how far away the nearest object in front of it is.



Ultrasonic Sensor detecting a wall (1)
The Ultrasonic Sensor sends out ultrasonic sound waves.



Ultrasonic Sensor detecting a wall (2)
The sound waves hit an obstacle and deflect back. The Ultrasonic Sensor receives the deflected sound waves, then calculates the difference between the time it sent the sound waves and the time it received them. Since the waves travel at a known speed (the speed of sound), the Ultrasonic Sensor can then calculate the distance to the obstacle (in this case, 40 centimeters).

The program you'll write in this lesson will work in a very similar way to the Touch Sensor program you wrote in the previous unit, but instead of using a Touch Sensor to detect obstacles by contact, it will use an Ultrasonic Sensor to detect them at a distance.

Wall Detection A Sonic Sojourn (cont.)

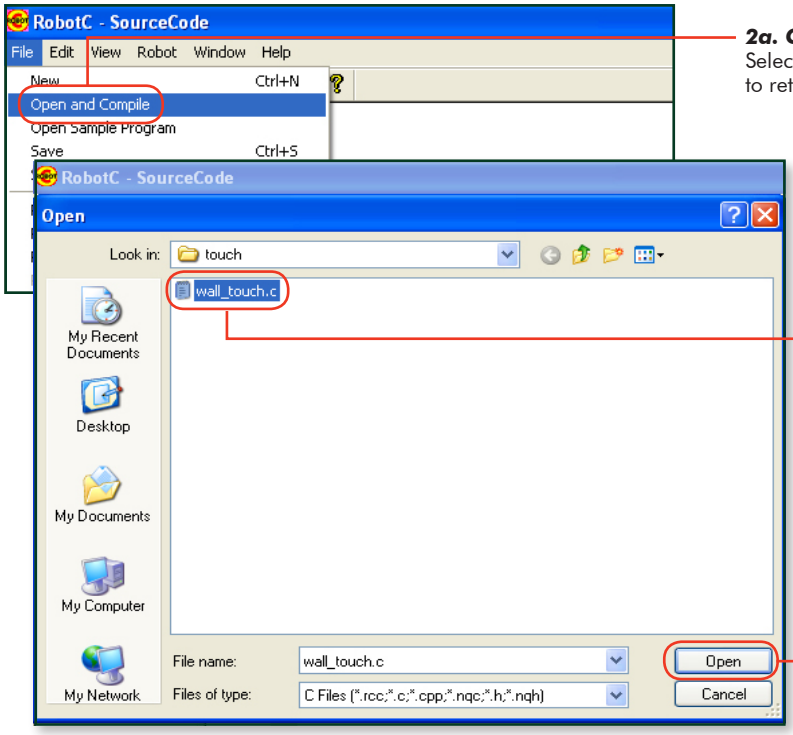
In this lesson, you will learn to use feedback from an Ultrasonic Sensor to make the robot detect a solid object and stop when it's 25 cm away.

1. Build the Ultrasonic Sensor attachment, and connect it to your robot.

1. Build the Ultrasonic Sensor attachment
Building instructions are available through the main lesson menu. Connect the Ultrasonic Sensor to port 1.

---

2. Open the "wall_touch" program you wrote for the previous section.



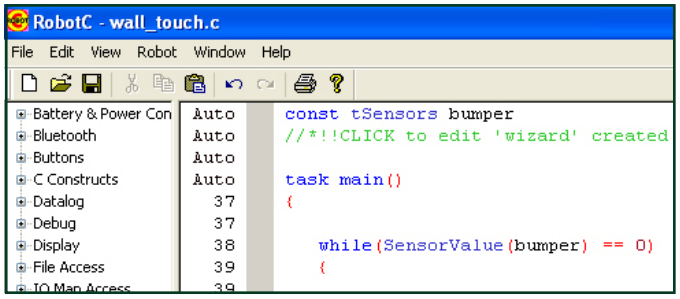
2a. Open Program
Select File > Open and Compile to retrieve your old program.

2b. Select the program
Select "wall_touch".

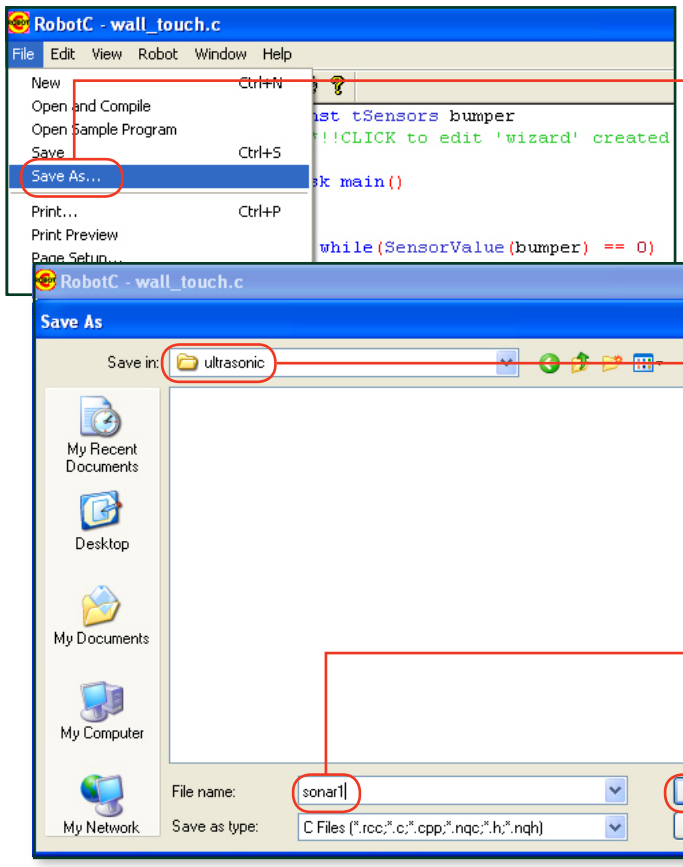
2c. Open the program
Press Open to open the saved program.

Wall Detection A Sonic Sojourn (cont.)

Checkpoint
The program should look like the one below.



3. Save this program under a new name, "sonar1".



3a. Save program as...
Select File > Save As... to save your program under a new name.

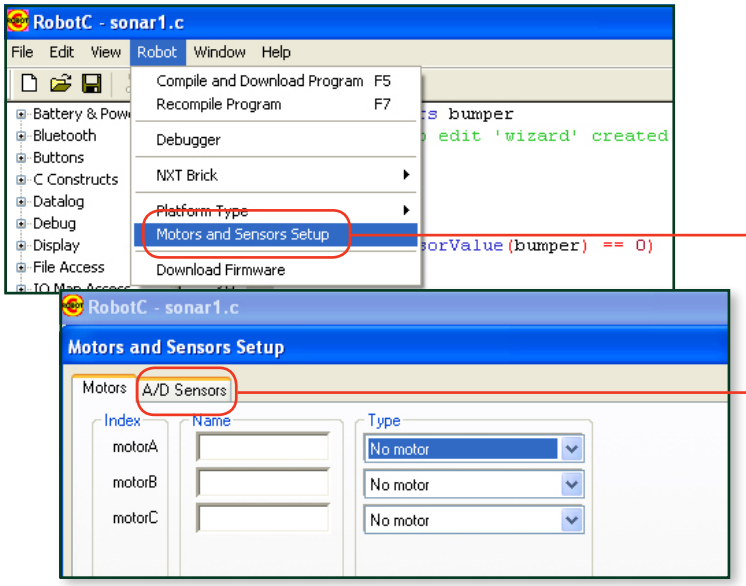
3b. Browse
Browse to and/or create an appropriate folder.

3c. Name the program
Give this program the name "sonar1".

3d. Save the program
Press Save to save the program with the new name.

Wall Detection A Sonic Sojourn (cont.)

4. Open the Motors and Sensors Setup menu, and go to the Sensors tab.

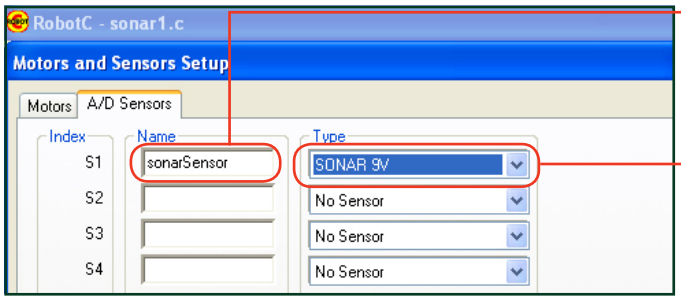


4a. Open "Motors and Sensors Setup"
Select Robot > Motors and Sensors Setup to open the Motors and Sensors Setup menu.

4b. Select the A/D Sensors tab
Click the "A/D Sensors tab" on the Motors and Sensors Setup menu.

---

5. Use the Motors and Sensors Setup interface to name the S1 sensor "sonarSensor", then select "SONAR 9V" as its type.



5a. Name sensor "sonarSensor"
Enter the name "sonarSensor" in the S1 name box.

5b. Make type "SONAR 9V"
Use the dropdown box to make "SONAR 9V" the sensor type.

Wall Detection A Sonic Sojourn (cont.)

Checkpoint
Your program should look like this. The while() loop is the focal point of its structure.

```
Auto const tSensors sonarSensor = (tSensors) S1;
Auto //*!!CLICK to edit 'wizard' created sensor
 1
 2 task main()
 3 {
 4
 5    while(SensorValue(bumper) == 0)
 6    {
 7
 8      motor[motorC] = 50;
 9      motor[motorB] = 50;
10
11    }
12
13    motor[motorC] = -50;
14    motor[motorB] = -50;
15    wait1Msec(2000);
16
17 }
```

while
The keyword while signals the beginning of the while loop.

The (condition)
As long as the (condition) is satisfied, the loop will continue repeating.

The {commands}
These commands are repeated over and over while the (condition) remains true.

The program uses the while() loop to check a certain (condition) to see whether it should keep looping or not. The (condition) right now is satisfied as long as the bumper is 0, or unpressed. The robot keeps running as long as this is true.

But now we're using the Ultrasonic Sensor. Having the (condition) look for a sensor value of 0 no longer makes sense, because the Ultrasonic Sensor can report a large range of values, not just one or zero. Remember, the Ultrasonic Sensor measures distance. It gives you a number that indicates the number of centimeters to the nearest detectable object in front of the sensor. It could be 1, 250, or anything in between.

The while() loop, however, doesn't want 250 different values, it just wants to make one decision: continue looping or go on to the next section of the program. The task is to get the robot to stop around 25 cm away from the obstacle. Ask yourself when the robot needs to run, and when it needs to stop. "The robot should run while...".

We'd like the robot to move forward while it is more than 25 cm away from the box, that is, while the distance to the box is greater than 25 (centimeters). Once the robot gets closer than 25cm, it should stop and move on to the next part of the program.

So, let's try that.

Wall Detection A Sonic Sojourn (cont.)

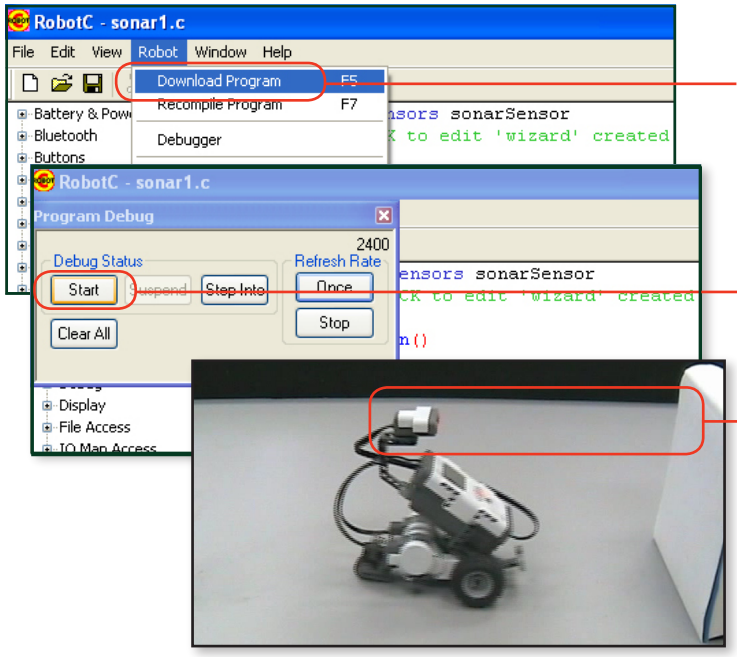
5. Change the loop's condition to make it run while the Ultrasonic Sensor's value is greater than 25cm.

```
Auto const tSensors sonarSensor = (tSensors) S1;
Auto //*!!CLICK to edit 'wizard' created sensor
 1
 2 task main()
 3 {
 4
 5    while(SensorValue(sonarSensor) > 25)
 6    {
 7
 8       motor[motorC] = 50;
 9       motor[motorB] = 50;
10
11    }
12
13    motor[motorC] = -50;
14    motor[motorB] = -50;
15    wait1Msec(2000);
16
17 }
```

5b. Change sensor name
Change the sensor name in the while () loop condition to "sonarSensor".

5b. Modify this code
Change the while() loop condition's value so that it will check whether the sonarSensor's value is greater than 25 cm.

6. Download and run the program. Disconnect the robot and move it onto the course if needed.



6a. Download the program
Click Robot > Download Program.

6b. Run the program
Click "Start" on the onscreen Program Debug window, or use the NXT's on-brick menus.

6c. 25cm Stop
The robot runs forward until the Ultrasonic Sensor detects an object < 25 cm away.

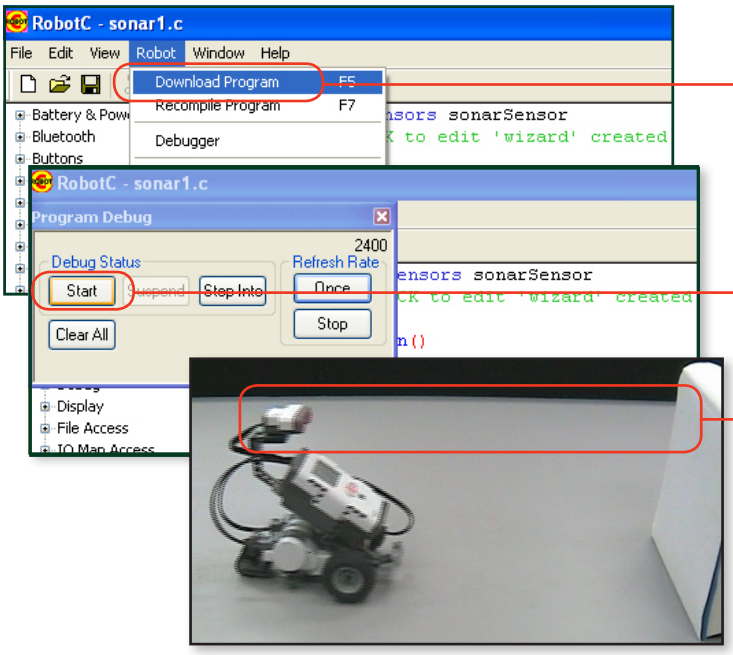
Wall Detection A Sonic Sojourn (cont.)

7. So we've succeeded in making the robot stop when it's 25 centimeters from an obstacle. Now let's try making the robot stop at some other distance from an obstacle.

```
Auto const tSensors sonarSensor = (tSensors) S1;
Auto //*!!CLICK to edit 'wizard' created sensor
 1
 2 task main()
 3 {
 4
 5    while(SensorValue(sonarSensor) > 40)
 6    {
 7
 8       motor[motorC] = 50;
 9       motor[motorB] = 50;
10
11    }
12
13    motor[motorC] = -50;
14    motor[motorB] = -50;
15    wait1Msec(2000);
16
17 }
```

7. Modify this code
Change the while() loop condition's value so that it will check whether the sonarSensor's value is greater than 40 cm.

8. Download and run the program. Disconnect the robot and move it onto the course if needed.



8a. Download the program
Click Robot > Download Program.

8b. Run the program
Click "Start" on the onscreen Program Debug window.

8c. 40cm Stop
The robot runs forward until the Ultrasonic Sensor detects an object < 40 cm away.

Wall Detection A Sonic Sojourn (cont.)

End of Section

You have modified your program to stop when the robot detects an object closer than a specified distance.

The number that you use to determine how far the robot stops is called a threshold. Thresholds are values that set a cutoff in a range of values, so that even though there are many possible values, every one of them will fall either above the threshold or below it.

In the case of the Ultrasonic Sensor, we set the threshold to 25 in our initial program, and made the distinction that values "greater than 25" will let the loop continue running, while values less than or equal to 25 will make the loop stop.

Then we changed the threshold to a different distance value, and saw how it affected the robot's behavior. By using thresholds, we can make use of the range of values an Ultrasonic Sensor provides to make a robot stop at whatever distance from an obstacle we want.