Coding Sensors Guide (ROBOTC®)

All sensors (except the built-in rotation sensor in the motor) connect to Ports 1, 2, 3, or 4. These ports are located on the bottom of the NXT brick and pass information through the NXT connector wires. Ensure that the wires connecting each sensor to the NXT brick are plugged into the correct port and that this has been correctly set up in the Motors and Sensors Setup under Robot in the ROBOTC® top menu.

1. Open the Motors and Sensors Setup and select the Sensors tab at the top.

2. Choose the port and the type of sensor that has been plugged into that port.

3. Give the sensor a name. It is best to choose a simple name that accurately represents the sensor because that is the name that will be used throughout the code.

   **Touch Sensor:** Choose Touch option.

   **Light Sensor:** Choose Light Active if the reflected light value is desired or Light Inactive if the ambient light value is desired. With the Light Active setting the robot will produce its own light, and with the Light Inactive setting it will not.

   **Sound Sensor:** Choose the Sound DB option if the true decibel value is desired or the Sound DBA option if the decibel value as a human would interpret it is more appropriate.

   **Ultrasonic Sensor:** Choose the SONAR option. The value of this sensor is returned in centimeters.

**Note:** The touch sensor will provide a digital signal, while the others will provide an analog signal. This means that the touch sensor will only be able to send two different pieces of information to the NXT brick (True or False, or in this case, pressed or not pressed), while the other sensors will send a value between 0 and 100. This is important to keep in mind when creating programs that use information from the sensors on the NXT Brick.
Coding Sensors Guide (ROBOTC®)

Example 1: Touch Sensor

The following program uses data from the touch sensor. The program will drive forward until the touch sensor is pressed. Once the touch sensor is pressed, the robot will stop and the program will end.

```c
task main()
{
    while(SensorValue(touch) == 0)
    {
        motor[motorD] = 70;
        motor[motorE] = 70;
    }
    motor[motorD] = 0;
    motor[motorE] = 0;
}
```

The While Loop allows the robot to continue to run until the case inside the brackets is no longer true.

**Note:** The numbers 0 and 1 take the place of the words “False” and “True” in the code, respectively.

The robot is in a False state when the touch sensor (in this case, named “touch”) is not pressed. It will continue to run the code inside the loop until the condition in the brackets is True. In other words, the touch sensor must be “True” (or pressed) in order to exit the loop. Therefore, once the touch sensor has been pressed, the code outside of the While Loop will run; the motors will stop and the program will end.

Example 2: Ultrasonic Sensor

The following program uses data from the ultrasonic sensor. The program will drive forward until the value reads 50 or less. Once the ultrasonic sensor reads a value that is in this range, the robot will stop and the program will end.

```c
task main()
{
    while(SensorValue(ultrasonic) > 50)
    {
        motor[motorD] = 70;
        motor[motorE] = 70;
    }
    motor[motorD] = 0;
    motor[motorE] = 0;
}
```

The While Loop allows the robot to continue to run until the case inside the brackets is no longer true. The robot’s ultrasonic sensor (in this case, named “ultrasonic”) will return a value between 0 and 100. It will continue to check the value returned by the sensor and will run the code inside the While Loop until the value is no longer above 50. When the value reads 50 or less, the While Loop will terminate and the code outside of the While Loop will execute. Therefore, once the value returned by the ultrasonic sensor reads 50 or less, the code outside of the While Loop will run; the motors will stop and the program will end.