Activity 3.1.3 Basic Inputs Programming – VEX

Introduction

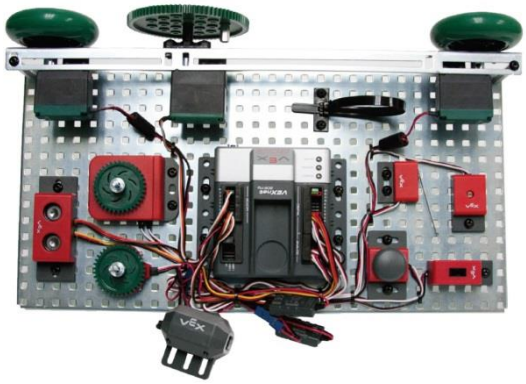
Inputs are devices which provide a processor with environmental information to make decisions. These devices have the capacity to sense the environment in a variety of ways such as physical touch, rotation, and light. An engineer can design a system to respond to its environment through the use of input sensors. In this activity you will use ROBOTC and VEX® robotics platform components to sense the environment.

Equipment

- Computer with ROBOTC software
- PoE VEX® testbed
- PLTW ROBOTC template

Procedure

1. Form groups of four and acquire your group's PoE VEX® Kit under your teacher's direction.
2. Within your four student group, form a two student team known as Team A and a two student team known as Team B.
    a. Team A will use the VEX® Testbed without the ultrasonic and the light sensor.
    b. Team B will use the VEX® Testbed without the servo motor and flashlight.
    c. At the appropriate time, both teams will exchange testbeds.
3. Connect the PoE VEX® testbed Cortex to the PC.

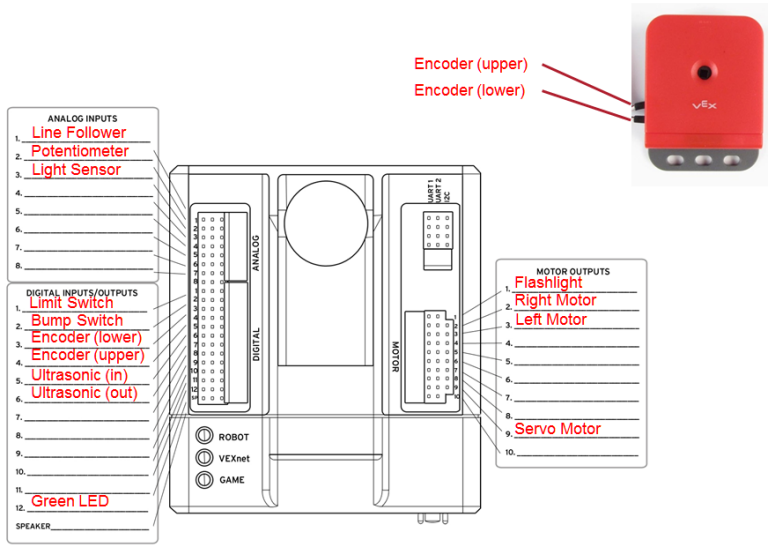


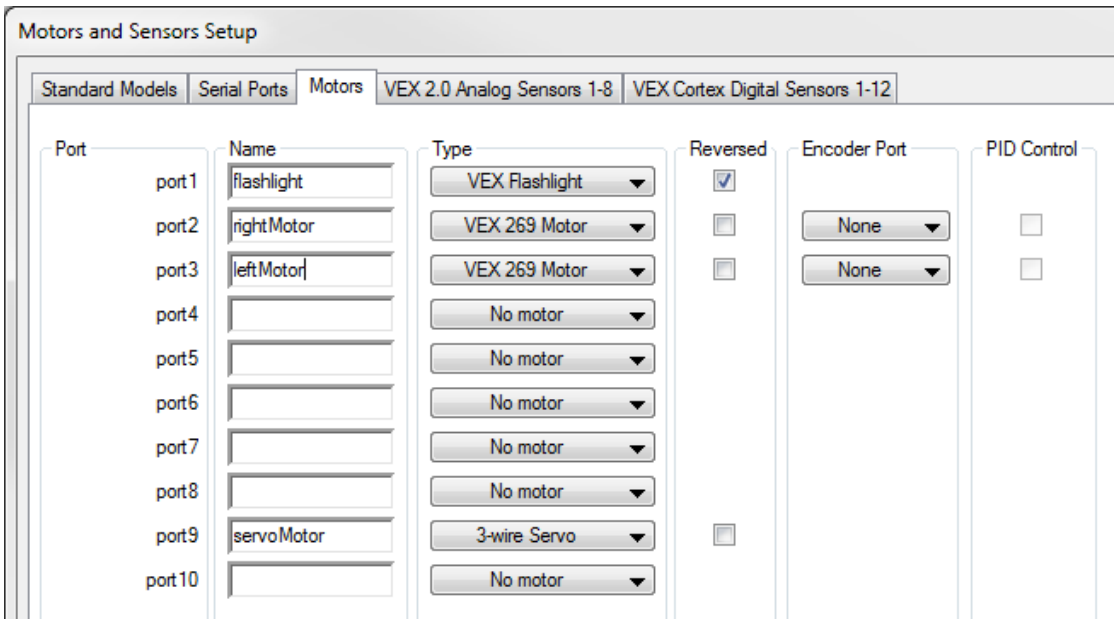
PoE VEX® Testbed

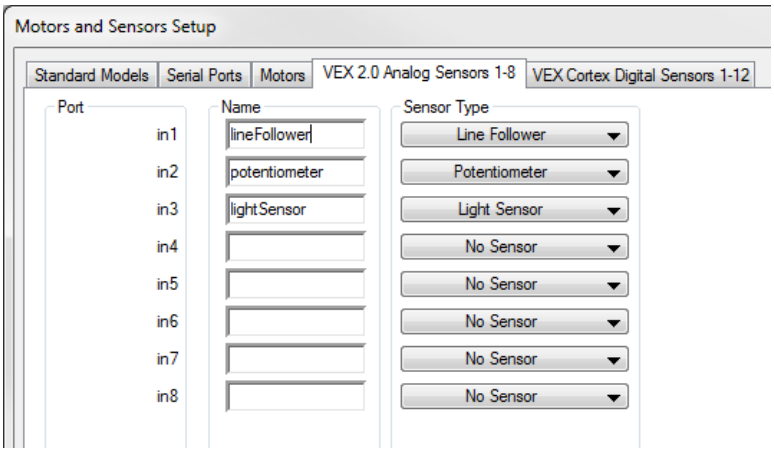
Parts 1 and 2: Using the Bump Switch

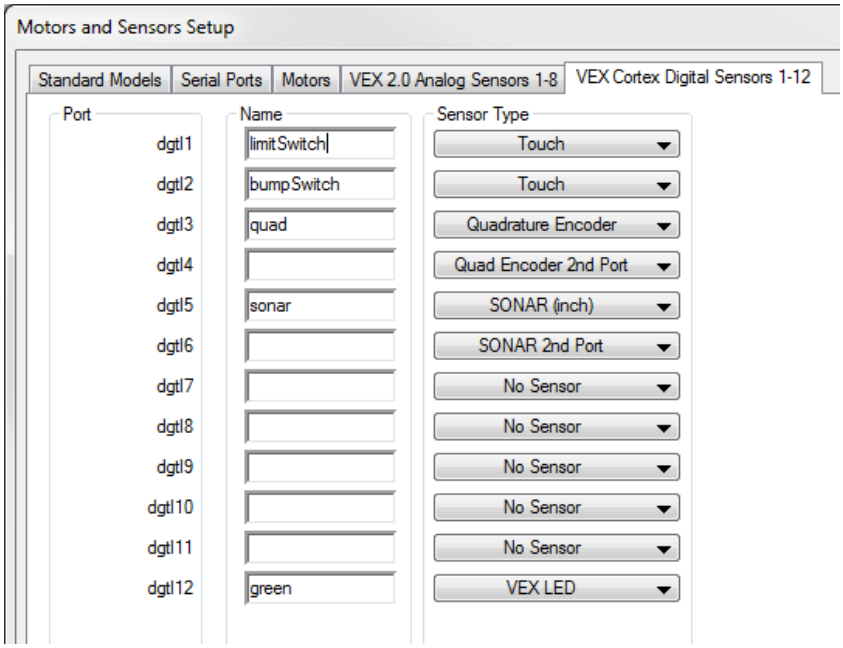
4. Open the PLTW ROBOTC template. Click File, Save As, select the folder that your teacher designated, and then name the file A3_1_3_Part1.
5. In this activity you will use all of the testbed input and outputs. Go to the Motors and Sensors Setup window. Configure the Motors and Sensors Setup to reflect the inputs and outputs to be used. Note that additional motors and sensors that are physically attached may be configured; however, these are not required to be configured. Click OK to close the window.

Cortex Wiring Diagram





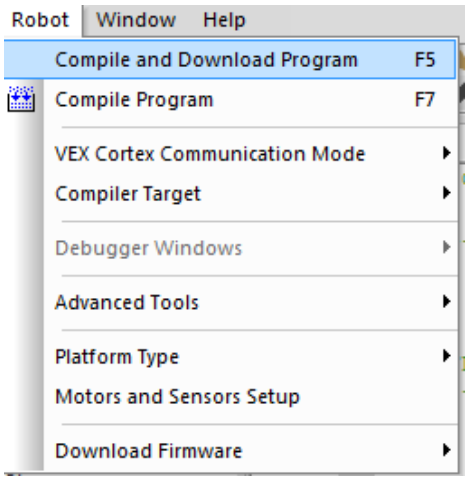




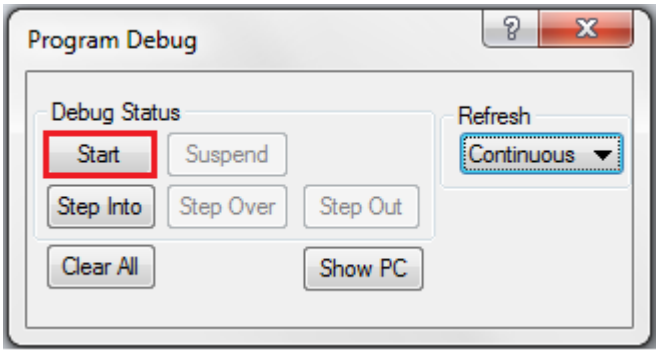
6. Use the program below in the task main() section of the program between the curly braces.

```
untilBump(bumpSwitch);
startMotor(rightMotor, 63);
wait(5);
stopMotor(rightMotor);
```

7. Power on the Cortex.
8. Save the program. Compile and download the program. If you have any errors, check with your instructor to troubleshoot your program.



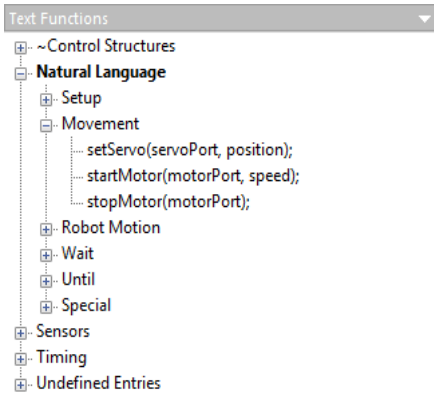
9. Press Start to run the program and observe the behaviors.



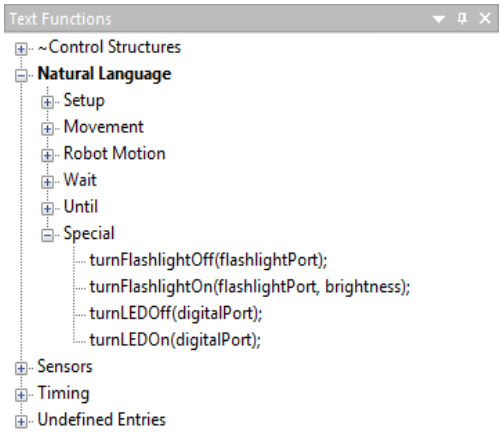
10. Document what this program would look like as pseudocode simple behaviors.

11. Open the PLTW ROBOTC template. Click File, Save As, select the folder that your teacher designated, and then name the file A3_1_3_Part2.
12. The wiring configuration and motors and sensors tabs should be the same as above.

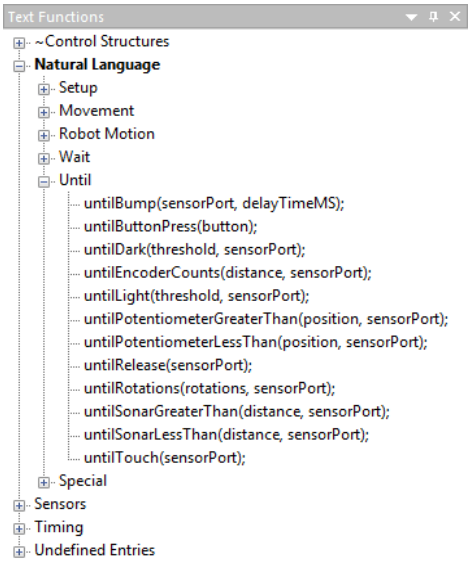
13. Write a program that performs the following simple behaviors. Use the natural language functions where appropriate as shown below. Add comments at the end of each command line to explain the purpose of each step.
    a. Wait for the bumper switch to be bumped. Note that a bump occurs when a switch is pressed and released and not simply pressed and held.
    b. Both motors turn on at half power until the sensor is bumped again.
    c. Both motors should then move in reverse at half power for 3.5 seconds.
    d. Both motors will stop.



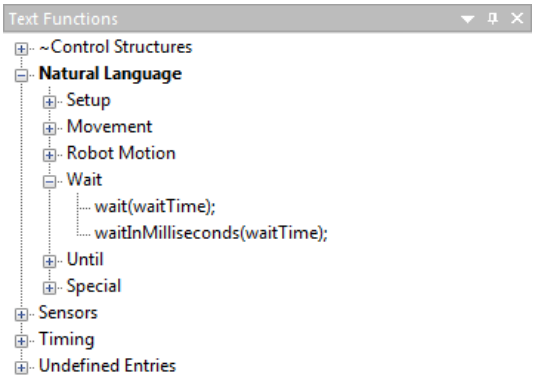
Natural Language Movement



Natural Language Special



Natural Language Until



Natural Language Wait

14. Test the program and troubleshoot if needed until the expected behavior has occurred. Save the program.

Part 3: Using the Potentiometer

15. Open the PLTW ROBOTC template Click File, Save As, select the folder that your teacher designated, and then name the file A3_1_3_Part3.
16. The wiring configuration and motors and sensors tabs should be the same as above.
17. Use the program below in the task main() section of the program between the curly braces.

```
turnLEDOn(green);
untilPotentiometerGreaterThan(2048, potentiometer);
turnLEDOff(green);
startMotor(leftMotor, 63);
wait(3.5);
stopMotor(leftMotor);
```

18. Download and run the program. Observe the behaviors and document what this program would look like as pseudocode simple behaviors.

19. Modify your program to perform the pseudocode below.
    a. Verify that the potentiometer is at a value of less than 2048.
    b. Turn on the greenLED until the potentiometer is at a value greater than 2048.
    c. Turn off the greenLED.
    d. Turn on the leftMotor at half power until the potentiometer is at a value of less than 2048.
    e. Turn leftMotor off.

Potentiometer

20. Test the program and troubleshoot if needed until the expected behavior has occurred. Save the program.

Part 4: Using the Optical Shaft Encoder

21. Open the PLTW ROBOTC template. Click File, Save As, select the folder that your teacher designated, and then name the file A3_1_3_Part4.
22. The wiring configuration and motors and sensors tabs should be the same as above.
23. Use the program below in the task main() section of the program between the curly braces.

```
startMotor(leftMotor, 63);
startMotor(rightMotor, 63);
untilEncoderCounts(480,quad);
stopMotor(leftMotor);
stopMotor(rightMotor);
```

24. Download and run the program. Observe the behaviors and document what this program would look like as pseudocode simple behaviors.

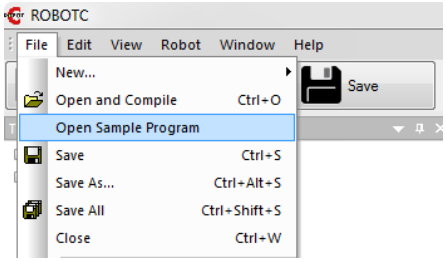
25. Modify your program to perform the pseudocode below.
    a. Turn on both motors forward until the encoder has counted 480 degrees.
    b. Turn on both motors in reverse until another 3.5 rotations of the encoder have passed.
    c. Turn off both motors.

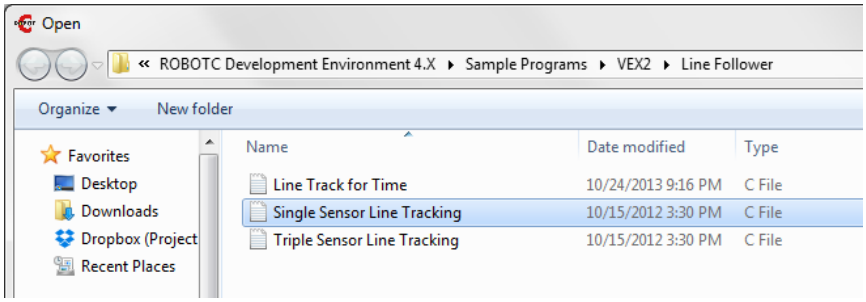
Optical Encoder

26. Test the program and troubleshoot until the expected behavior has occurred. Save the program.

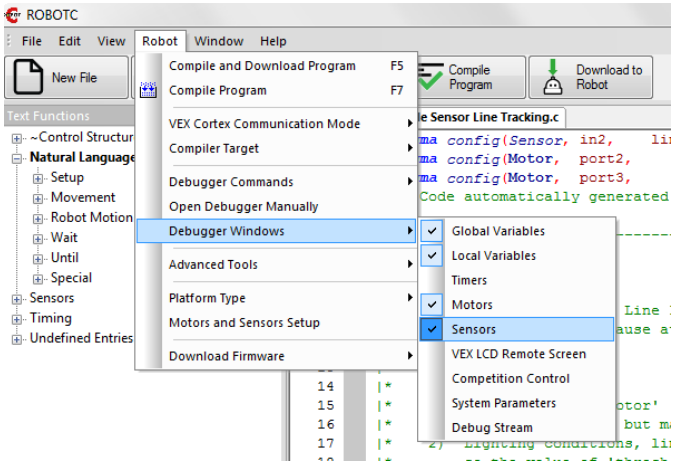
Part 5: Using the Infrared Line Follower

27. Open the PLTW ROBOTC template. Click File, Save As, select the folder that your teacher designated, and then name the file A3_1_3_Part5.
28. The wiring configuration and motors and sensors tabs should be the same as above.
29. Set the line follower threshold. Thresholds allow your robot to make decisions via Boolean comparisons.
    a. Calculate an appropriate threshold with the aid of the Sensor Debug Window.

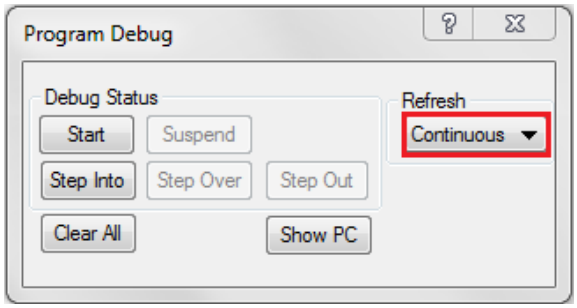




    b. Open the Sensor Debug Window.



c. Verify that the Program Debug Window's Refresh rate displays Continuous. Select Continuous from the dropdown menu if it is paused.



d. Place a white object (e.g., paper) within ¼ and 1/8 in. in front of the line follower sensor. Record the value for that sensor displayed in the Sensors Debug Window. Make sure that enough light is available to illuminate the white object or the sensor will register darkness.

e. Place a dark object within ¼ and 1/8 in. in front of the line follower sensor. Record the value for that sensor displayed in the Sensors Debug Window.

f. Add the two values and divide by two. The result is the threshold for that sensor.

30. Use the program below in the task main() section of the program between the curly braces. Change the value 1425 to the value calculated in the previous step.

```
setServo(servoMotor, 127);
untilLight(1425, lineFollower);
setServo(servoMotor, -127);
wait(2);
```

31. Download and run the program. Observe the behaviors and document what this program would look like as pseudocode simple behaviors.

32. Modify your program to perform the pseudocode below.
    a. Move the servo to position 127 until a dark object is detected.
    b. Move servo to position -127.

Line Follower

33. Test the program and troubleshoot until the expected behavior has occurred. Save the program.

34. Teams prepare to share testbeds. Team A will install one ultrasonic sensor on the testbed. Team A completes the following steps related to the ultrasonic sensor and then exchanges testbeds with team B. Team B completes steps associated with the ultrasonic sensor.

Part 6: Using the Ultrasonic Distance Sensor

35. Open the PLTW ROBOTC template. Click File, Save As, select the folder that your teacher designated, and then name the file A3_1_3_Part6.
36. The wiring configuration and motors and sensors tabs should be the same as above.
37. Use the program below in the task main() section of the program between the curly braces.

```
startMotor(leftMotor, 63);
startMotor(rightMotor, 63);
untilSonarLessThan(20, sonar);
stopMotor(leftMotor);
stopMotor(rightMotor);
turnLEDOn(green);
wait(6.25);
turnLEDOff(green);
```

38. Download and run the program. Observe the behaviors and document what this program would look like as pseudocode simple behaviors.

39. Modify your program to perform the pseudocode below.
    a. Wait until an object is detected within 20 cm to turn both motors on.
    b. Wait for the object to move more than 25 cm away before turning the motors off.

Ultrasonic Sensor

40. Test the program and troubleshoot if needed until the expected behavior has occurred. Save the program.
41. Exchange testbeds. Team B will complete the ultrasonic steps.
42. Follow teacher direction and either print the programs or submit electronically with this activity.

Conclusion

1. Describe any challenges that you encountered while developing the program.

2. Describe three applications for the use of sensors that you worked with in this activity.