Activity 3.1.4 While Loops and If-Else Structures – VEX

Introduction

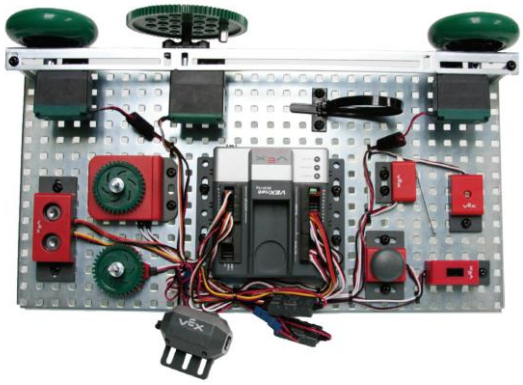
One of the powerful attributes of a computer program is its ability to make decisions. Although it can be argued that only humans are capable of decision making, computers are able to make decisions using criteria. They are able to compare two values and determine whether one is larger than the other. They can determine whether a statement is true or false, based on empirical data.

Equipment

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

Procedure

1. Form teams of two and acquire your team's POE testbed under your teacher's direction.
2. Connect the POE VEX testbed Cortex to the PC.



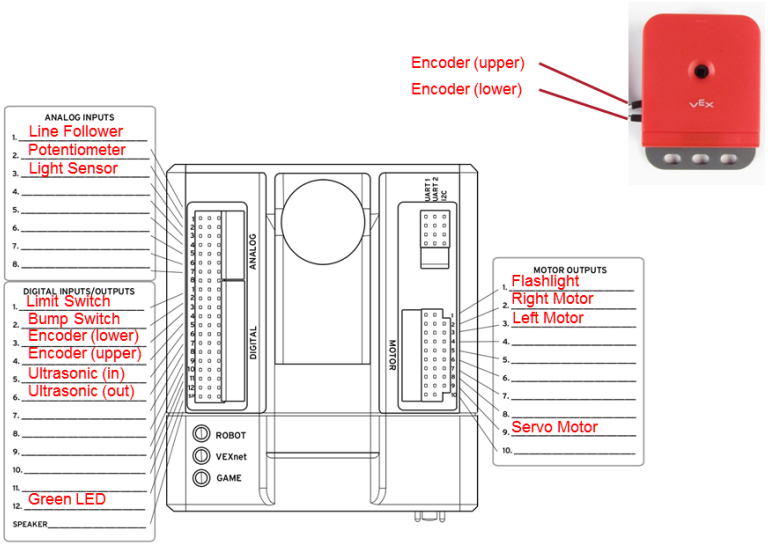
POE VEX Testbed

Part 1: A Flashlight Responding to Light

3. Open the PLTW ROBOTC template. Click File, Save As, select the folder that your teacher designated for you to save your ROBOTC programs in, then name the file A3_1_4_Part1.
4. 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



5. A while loop is a structure within ROBOTC which allows a portion of code to be run over and over, as long as a certain condition remains true.

Below is the pseudocode outline of a while loop.

```
while(condition)
{
  // repeated-commands
}
```

(condition)
Either **true** or **false** (see Reference > Boolean Logic).

Repeated commands
Commands placed here will run over and over as long as the (condition) is **true** when the program checks at the beginning of each pass through the loop.

Below is an example of a program using an infinite While Loop.

```
task main()
{
 bMotorReflected[port2]=1;
 while(1 == 1)
 {
  motor[port3]=127;
  motor[port2]=127;
 }
}
```

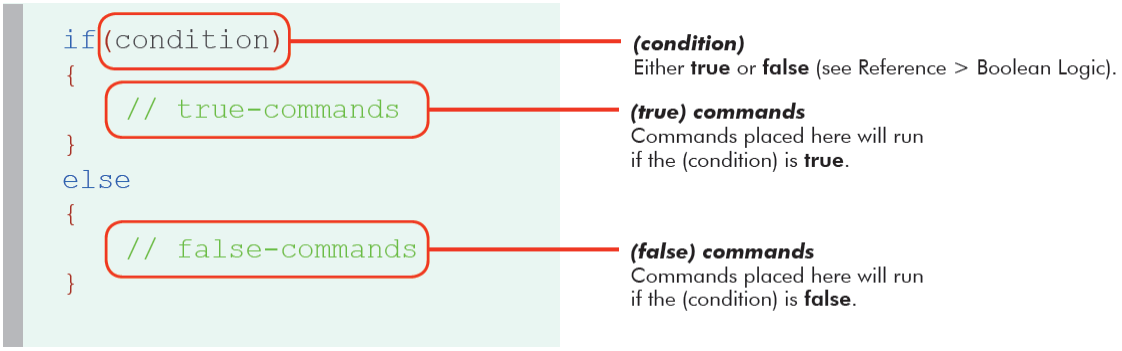
The condition is true as long as 1 is equal to 1, which is always.

While the condition is true, both motors will receive full power.

6. Copy and paste or create the program below in the `task main()` section of the program between the curly braces. Note that the light threshold will vary depending on ambient light, so you may have to change the 700.

```
while(1 == 1)
{
   if (SensorValue(lightSensor)>700)
   {
      turnFlashlightOn(flashlight, 127);
   }
   if (SensorValue(lightSensor) <= 700)
   {
      turnFlashlightOff(flashlight,0);
   }
}
```

7. Save the program, power on the Cortex, compile, and download the program. If you have any errors, check with your instructor to troubleshoot your program.
8. Press Start to run the program and observe the behaviors.
9. Document this program with pseucode and line-by-line comments.
10. An if-else statement is one way to allow a computer to make a decision. With this command the computer will execute one of two pieces of code, depending on whether the condition is true or false. Examine the following code.



11. Modify your program to use an if-else statement as shown below.

```
if (condition)
```

```
{
   statement;
}
else
{
   statement;
}
```

12. Adjust the line-by-line comments, and save your program. Explain why the single if-else structure in #11 might be preferable to the two if structures in #6.

Part 2: A Flashlight Responding to Light and a Switch

13. Open the PLTW ROBOTC template. Click File, Save As, select the folder that your teacher designated, and then name the file A3_1_4_Part2.
14. Write a program that performs the behavior below. You can use the while loop structure below (or any other structure that accomplishes the task) and Boolean Logic table below when developing the program.

    Task: Program the Cortex so that when the limit switch is pressed, the flashlight responds to light. When the limit switch is pressed, the flashlight should turn on when it is dark in the room (or the sensor is blocked) and off when it is bright in the room. When the limit switch is not pressed, the flashlight should always be off. The program should loop indefinitely, waiting until the limit switch is pressed again. If your group doesn't have the flashlight, use a motor instead.

```
while (condition) //repeat indefinitely
{
   while (condition) //repeat while limitSwitch pressed
   {
      if (condition)  //respond to lightSensor
      {
      }
      else
      {
      }
   }
   //do this when the limitSwitch is not pressed
}
```

ROBOTC Symbol	Meaning	Sample comparison	Result
==	"is equal to"	50 == 50	true
		50 == 100	false
		100 == 50	false
!=	"is not equal to"	50 != 50	false
		50 != 100	true
		100 != 50	true
<	"is less than"	50 < 50	false
		50 < 100	true
		100 < 50	false
<=	"is less than or equal to"	50 <= 50	true
		50 <= 100	true
		50 <= 0	false
>	"is greater than"	50 > 50	false
		50 > 100	false
		100 > 50	true
>=	Greater than or equal to	50 >= 50	true
		50 >= 100	false
		100 >= 50	true

Boolean Logic

15. Test the program and troubleshoot until the expected behavior has occurred. Make sure your code is documented with a task description, pseudocode, and line-by-line comments. Save the program.

Part 3: Blinking LED with Switch

16. Open the PLTW ROBOTC template. Click File, Save As, select the folder that your teacher designated, and then name the file A3_1_4_Part3.
17. Copy and paste or create the program below in the `task main()` section of the program between the curly braces.

```
ClearTimer(T1);
while ( time1(T1) < 20000)     //Loop for 20 sec
{
  turnLEDOn(green);
  wait(2);
  turnLEDOff(green);
```

```
    wait(2);
  }
```

18. Compile, download and run the program, and observe its behavior.
19. Modify the program to perform a new task. Copy the following task into your code's task description:

    When the bump switch is pressed, the LED flashes. When the bump switch is not pressed, the LED is off.
20. Write pseudocode to implement this task, describing the simple behaviors. Copy the pseudocode into the task main, turning it into line-by-line comments. Write code to implement the task, changing the comments, if needed, to match your revisions.
21. Follow the teacher direction and either print the programs or submit electronically with this activity.

Part 4: Additional Practice

22. Create programs to accomplish one or more of the following tasks as directed by your teacher. For each task to be completed, open the PLTW ROBOTC template. Use Save As to create a new file named A3_1_4_Part4A (or 4B, etc.). Copy the task into the task decription and create the corresponding pseudocode. Copy the pseudocode into the task main as a starting point for your line-by-line comments

    A. Make a motor spin as long as the bump switch is being held down. Its direction depends on whether a person is within 20 cm of the ultrasonic sensor. If the button is released, the behavior will repeat if it is pressed again.

    B. Make the servo position itself to the left or right depending on whether the line follower is covered by your hand.

    C. Make the servo position itself to the left or right depending on whether the line follower is covered by your hand. This behavior will only apply if the button is pressed; if the button is released, the servo is in a middle position, awaiting the button to be pressed again.

    D. Make one motor spin whenever a button is pressed and a second motor spin whenever a limit switch is pressed. This behavior repeats indefinitely, the two things being independent.

Conclusion

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

2. Describe one application each for While loops and If-Else structures.