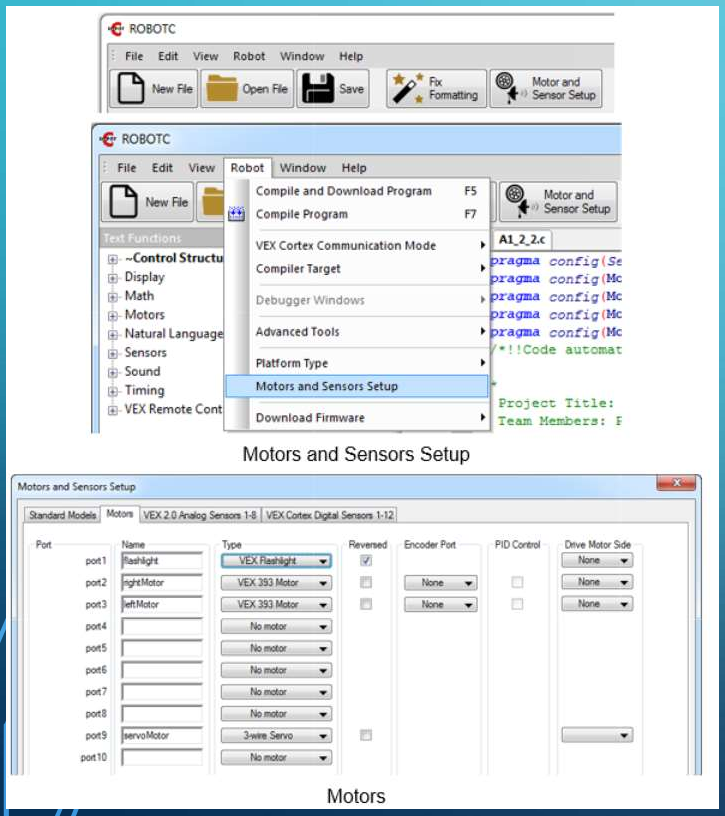
ROBOTICS (ROBOT C/VEX)

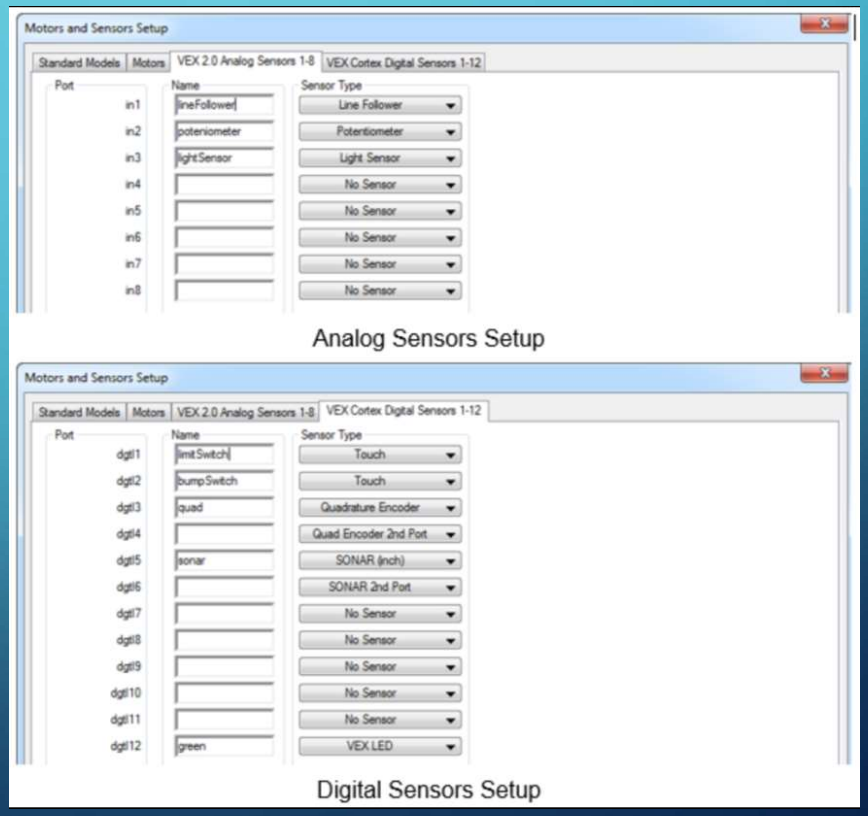
GRACE HOLLOWAY

KIERAN KELLEHER

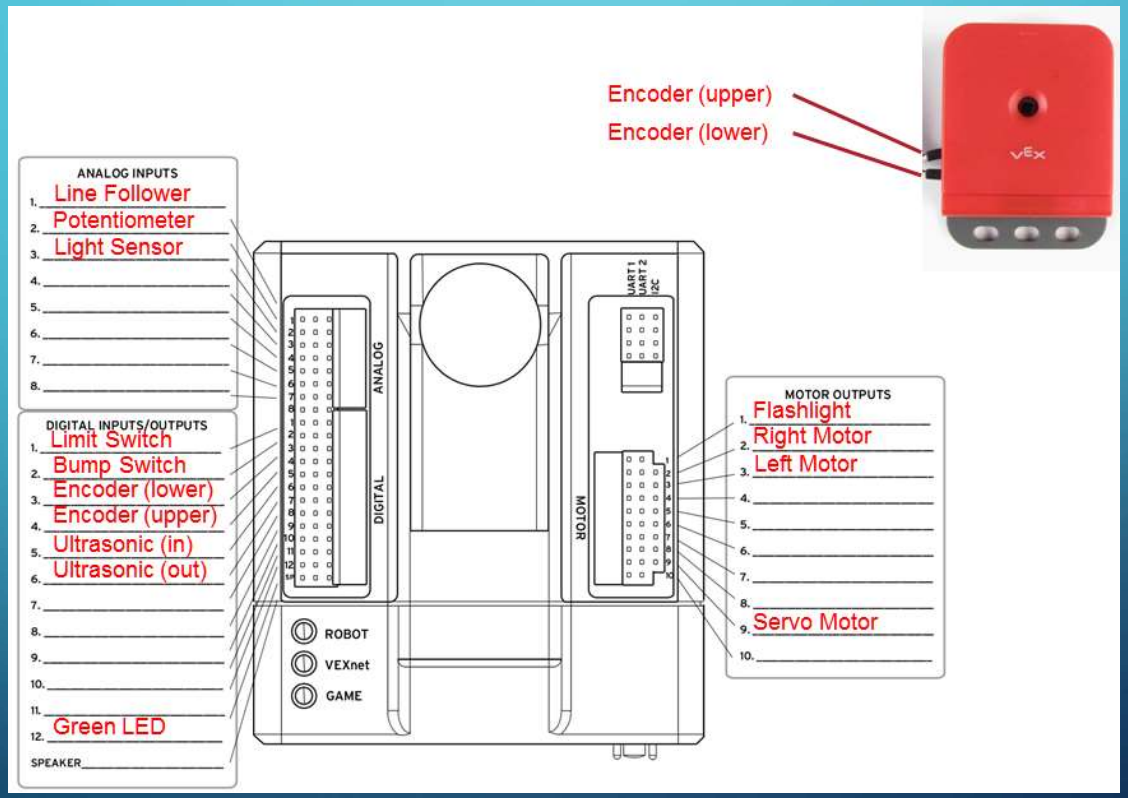
SETTING UP ROBOT C

- Connect the testbed to computer via USB
- Setting up the motors and sensors:





SETTING UP TESTBED

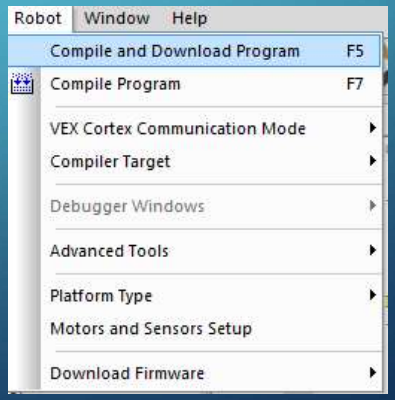


BASIC INPUTS AND OUTPUTS

- Use this program below in the task main() section of the program between the curly braces

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

- Power on the cortex
- Save the program. Compile and download the program. If you have any errors, check with me or Kieran so that we can troubleshoot



WHAT WOULD THIS PROGRAM LOOK LIKE AS PSEUDOCODE SIMPLE BEHAVIORS?

- Basically writing your code in English—what actions does your code produce?
- The right motor turns for 5 seconds and then stops once the bumper switch is switched

WHILE LOOP STRUCTURES IN ROBOTC

- 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.

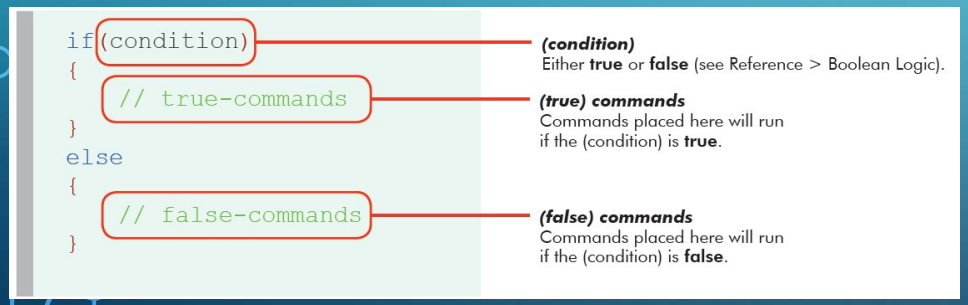
- Use the program below in the task main() section of the program between the curly braces. Note that the light threshold will vary depending on ambient light, so you may have to change the 700 light sensor value.

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

- Press start and observe the behaviors
- What happens?

IF-ELSE STRUCTURES IN ROBOTC

- 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.

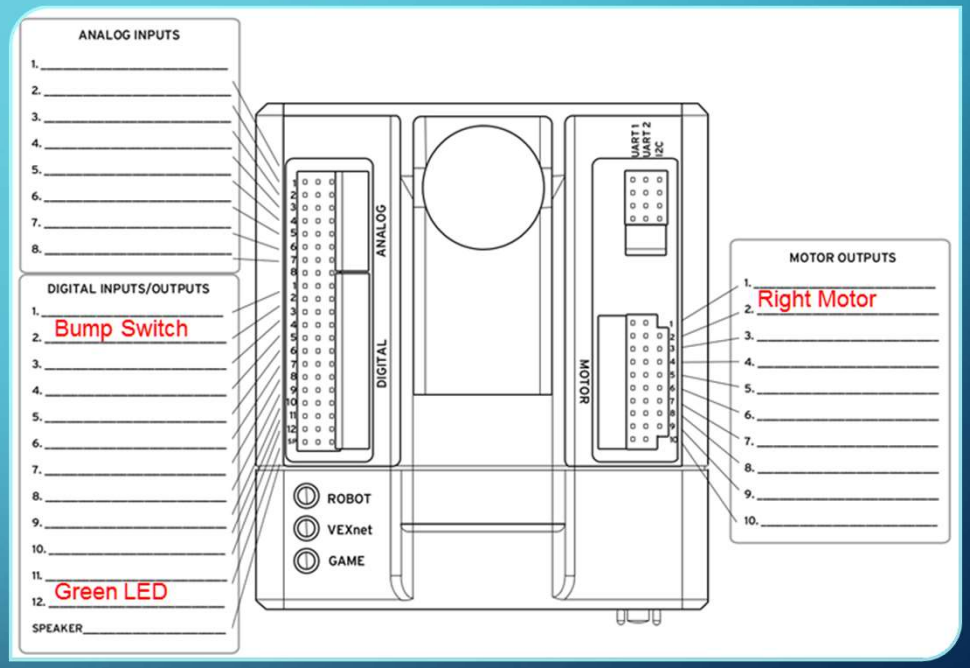


- Modify the previous "while loop" code to use an if-else statement
- if (*condition*)
- {
- *statement;*
- }
- else
- {
- *statement;*
- }

- Why might a single if-else structure might be preferable to the two previous “if” statements in the while loop structure?

VARIABLES AND FUNCTIONS

- A program can accomplish a given task in any number of ways. Programs can quickly grow to an unmanageable size, so variables and functions provide a technique to reduce the size of the program. A variable is a space in your robot's memory where you can store data, such as whole numbers, decimal numbers, and words. Functions group together several lines of code, which can then be referenced many times in task main or even other functions. The use of variables and functions allows for complex control of a system.
- Motors and sensors configuration:



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

```
int motorCount;
motorCount = 0;
while (motorCount < 3)
 {
   startMotor(rightMotor, 95);
   wait(2);
   stopMotor(rightMotor);
   wait(2);
   motorCount = motorCount + 1;
 }
```

- Save the program, power on the cortex, compile, and download the program. Press start and observe the beahviors

CHALLENGE TIME >:)

- Create a program that will run a motor back and forth 20 times, 0.5 seconds each way. Test the program and troubleshoot until the expected behavior has occurred

```
int motorCount;
  motorCount = 20;
  while (motorCount < 3)
  {
    startMotor(rightMotor, 95);
    wait(0.5);
    stopMotor(rightMotor);
  wait(0.5);
    motorCount = motorCount + 1;
  }
```