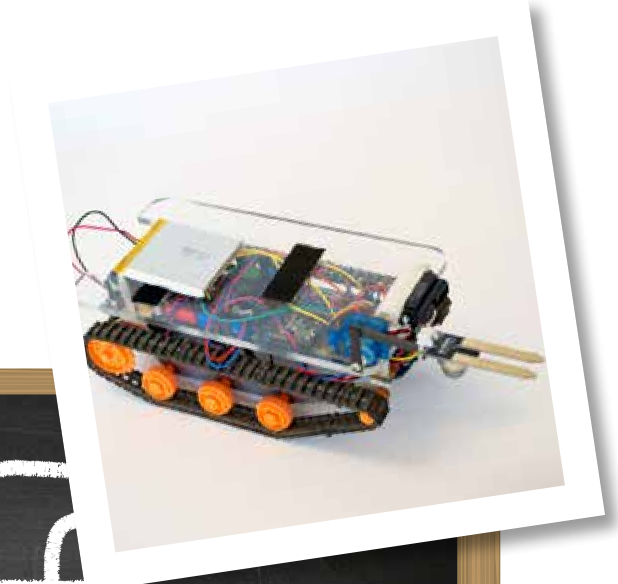
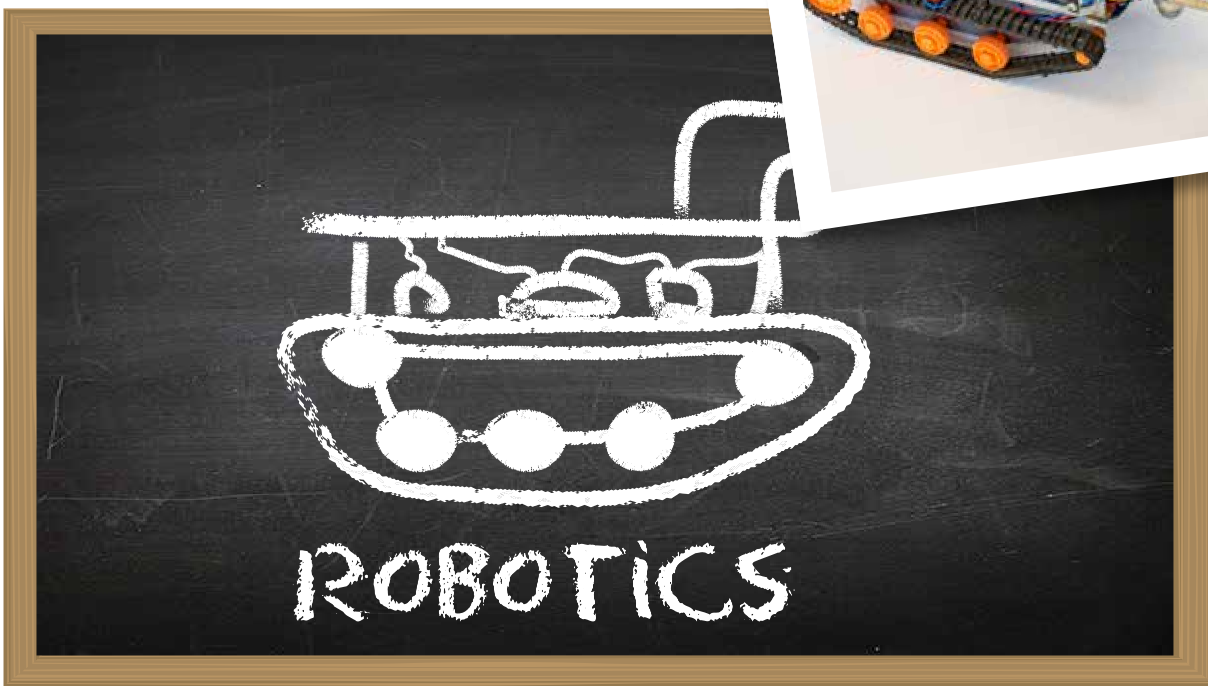
STEM | Science | Technology | Engineering | Mathematics





ROBOTICS ADVENTURE BOOK

Scouter Manual

A word with our Scouters:

This activity is designed around a space exploration theme. Your Scouts will learn to program the Robot to perform different functions that are useful for exploration in a remote area.

In this booklet, you can find sample programs for activities that your Scouts should train the robot for. These are for your preparation so that you can better help your Scouts when they are trying to design their own programs.

When working on the activities, ask the Scouts to write the steps that the robot should take (the algorithm) before they begin working with the programming software. This will allow them to have a clear structure for the program before they begin dealing with the details of commands.

Here is the scenario the Scouts are working with:

Imagine

It's 2028 and it's been 15 years since you first had a chance to explore science, engineering, technology and mathematics in Scouts. Now **you're a young engineer and you've got a chance to work on a brand new space probe**.

In the last 15 years, remarkable advances have been made by scientists exploring the fundamentals of physics, and **we've cracked the problem of near light speed trave**l.

For 20 years we've been discovering planets circling other stars and star systems (Exoplanets). N**ow, we're set to go and explore several promising candidates and look for molecular signs of life**.

Here's the catch: the robotic probe that you will design will take nearly 20 years to reach these planets, and signals to and from the robots will take nearly 10 years each way. **The robots will have to explore their new planets on their own, with no direction from Earth**.

Your Mission

You will program a prototype robot to search in a pattern, sampling to find water and hydrocarbons.

There are no GPS satellites orbiting these far-off worlds. Their magnetic fields won't be anything like that of Earth, so even a compass is of no use. All you'll have to go on for position is how much the robot's wheels have turned.

As a young engineer, what are the first things that you'd want to know?

Getting To Know Your Robot

The hardware used for the STEM Robots consists of an Arduino base platform with tracks, drive motors, a microprocessor board, LEDs for signaling, and digital and analog input and output ports ("pins").

The digital and analog input and output ports permit additional instrumentation to be added to the basic unit functionality.

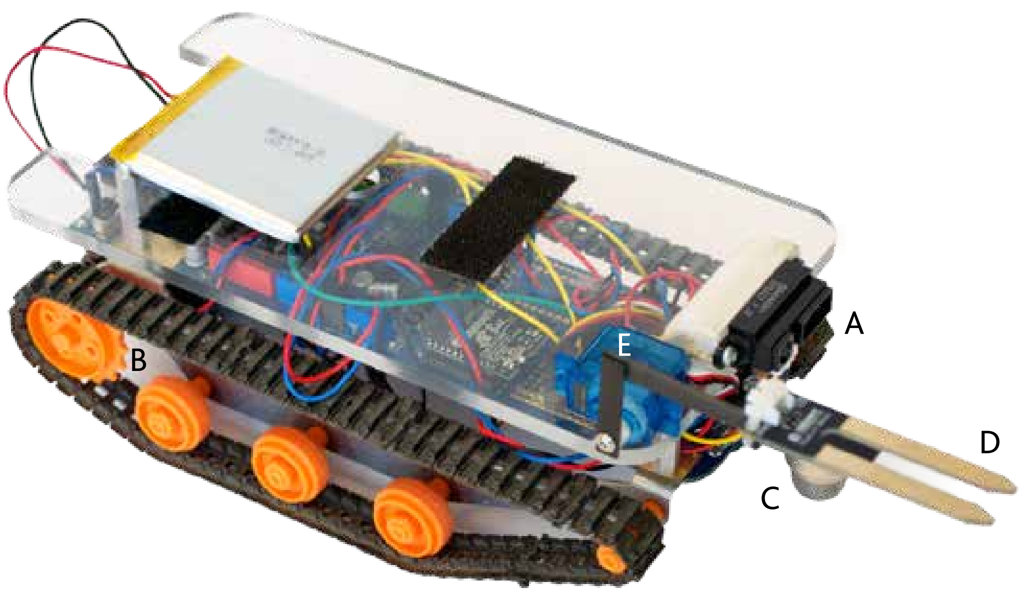
The additional instrumentation, in the case of the STEM Robots, consists of:

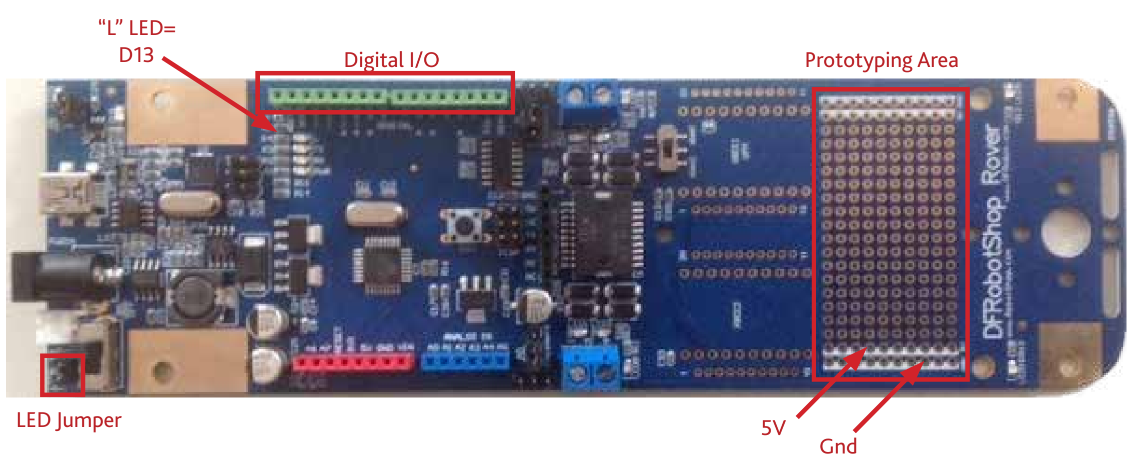
A "Proximity" detector: an infrared detector

B "Distance travelled" detectors: an encoder on each set of drive tracks

C & D "Signs of life" detectors: a hydrocarbon detector and a moisture detector

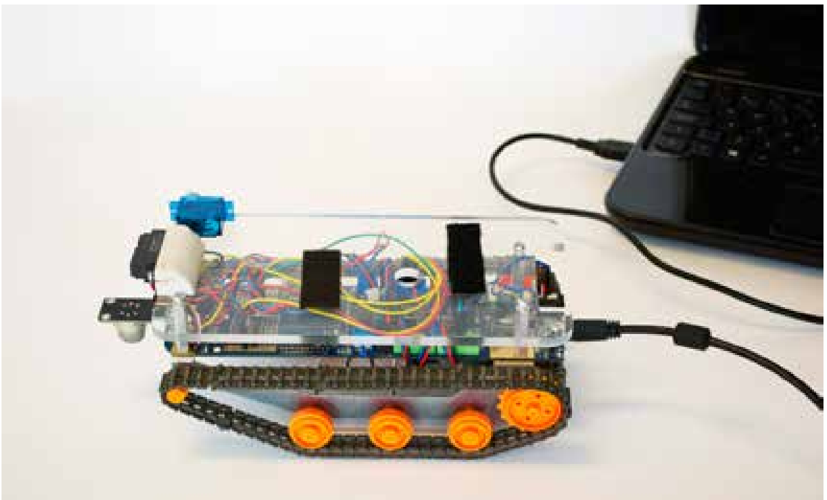
E "Detector operator": a servo motor used to control the position of the moisture detector



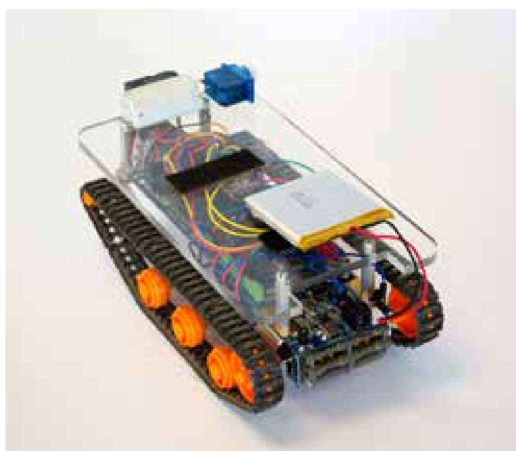


How to Connect Parts of the Robot

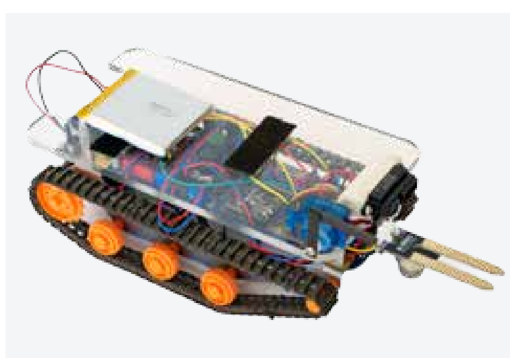
The robot connects to the computer with the USB cable.



The battery is connected through the plug next to the USB port on the robot.



The moisture sensor is connected through the plug with blue, red and black wiring. It is then screwed to the moisture arm.



Overview of Robot Electronic Controls

Inputs

Inputs are used to obtain information about the world. In this robot, inputs include analog values from the wheel encoders and the various instrumentation sensors (moisture, hydrocarbon, and proximity):

- Pin 0 – right track optical encoder (to measure distance)
- Pin 1 – left track optical encoder (to measure distance)
- Pin 2 – infrared sensor (to detect proximity to an object)
- Pin 3 – moisture sensor (dry, moist, very wet)
- Pin 4 – hydrocarbon sensor (to detect evidence of the molecules of life)

The analog inputs give a value between 0 and 1024 that represents a voltage between 0V and 5V. Sensors provide voltage outputs that mean different things for different sensors; you have to read the "Data Sheet" to know exactly what these are.

Outputs

Outputs are used to "send" information to the real world or peripheral devices. On this robot, outputs include:

- Pin 5 & 6 - Analog values (in the form of DC voltages) to the track drive motors to control their speed
- Pin 7 & 8 - Digital values ('high'/'low') to control the motor drive directions
- Pin 11 - Analog value (also DC voltage) for positioning of the Servo
- Pin 12&13 – Digital values ('high'/'low') to control the LEDs

Summary of Input/ Outputs

Inputs

- Right Track Encoder
- Left Track Encoder
- Infrared Sensor (Proximity)
- Moisture Sensor
- Hydrocarbon Sensor

Outputs

- Left Track Motor Speed
- Right Track Motor Speed
- Left Track Motor Direction
- Right Track Motor Direction
- Servo Motor Control
- Blue Indicating LEDs
- Green Indicating LEDs

Data Sheet: Outputs

Pin	Function	Command	Parameters
0 -4	(don't use)		
5	Left motor speed	Set Analog Pin# 5	Stop=0 to fast=255*
6	Right motor speed	Set Analog Pin #6	Stop=0 to fast=255*
7	Left motor direction	Set Digital Pin #7	Low= Forward High=Backward
8	Right motor direction	Set Digital Pin #8	Low= Forward High=Backward
9-10	(don't use)		
11	Moisture sensor arm	Servo (pin# 11, angle)	Angle in degrees
12	Blue LEDs	Set Digital Pin #12	Low= off High=On
13	Green LEDs	Set Digital Pin #13	Low= off High=On

Data Sheet: Inputs

Pin	Function	Command	Parameters
0	Right track optical encoder	Set Number Variable (Variable, pin # 0)*	3 edges = 2cm An edge is a change from below 600 to above 600
1	Left track optical encoder	Set Number Variable (Variable, pin # 1)	3 edges = 2cm An edge is a change from below 600 to above 600
2	Infrared Sensor	Set Number Variable (Variable, pin # 2)	Volts=Sensor Value *5/1024 Distance in cm=27*volts^(-1.1)
3	Moisture Sensor	Set Number Variable (Variable, pin # 3)	0-300 = dry soil 300-700 = humid soil 700-950 = in water
4	Hydrocarbon Sensor	Set Number Variable (Variable, pin # 4)	<150 = none 150-450 = maybe >450 = yes!!!
5-7	(don't use)		

Installing the Programming Software (Arduino) and the Robot on the Computer

The robot kit comes with a laptop that has all the programs installed, but you can bring in more computers to your meeting so that more Scouts can work on their programming simultaneously. To do this, you need to install the programming software and the robot's driver on the additional computers.

Install the "Arduino" compiler:

1. Open your browser to http://arduino.cc/en/Main/Software

2. Click the "Windows" link under "Download" "Arduino 1.0.5" and save the "zip" file.

3. When the download finishes, unzip the downloaded file. Make sure to preserve the folder structure. Double-click the folder to open it. There should be a few files and sub-folders inside.

Install the graphical programming environment "ArduBlocks":

4. Open your browser to http://blog.ardublock.com/engetting-started-ardublockzhardublock/ and follow the instructions there to:

   a. Download the ardublock-all.jar file.

   b. Put it in the Documents\Arduino\tools\ArduBlockTool\tool\ or My Documents\Arduino\tools\ArduBlockTool\tool\ (whichever matches your operating system) folder under your home directory.

At this point you can open the programming environment even if you don't have the robot. If you want to do that, skip to step 12. You can always come back to step 5 when you do have the robot.

Connect the Robot

The robots use a version of the Arduino Uno board. These boards automatically draw power from either the USB connection to the computer or an external power supply.

5. Connect the robot to your computer using the USB cable. The green power LED (labelled PWR) should go on.

Install the drivers for the Arduino Uno (Robot) :

(See also: step-by-step screenshots for installing the Uno: http://arduino.cc/en/Main/ArduinoBoardUno)

6. Plug in the robot and wait for Windows to begin its driver installation process. After a few moments, the process will fail, despite its best efforts.

7. Click on the Start Menu, and open up the Control Panel. While in the Control Panel, navigate to System and Security. Next, click on System. Once the System window is up, open the Device Manager.

8. Look under Ports (COM & LPT). You should see an open port named "Arduino UNO (COMxx)"

9. Right click on the "Arduino UNO (COmxx)" port and choose the "Update Driver Software" option.

10. Next, choose the "Browse my computer for Driver software" option.

11. Finally, navigate to and select the Uno's driver file, named "ArduinoUNO.inf", located in the "Drivers" folder of the Arduino Software download (not the "FTDI USB Drivers" sub-directory).Windows will finish up the driver installation from there.

Opening the programming environment:

12. Double-click the "Arduino" application.

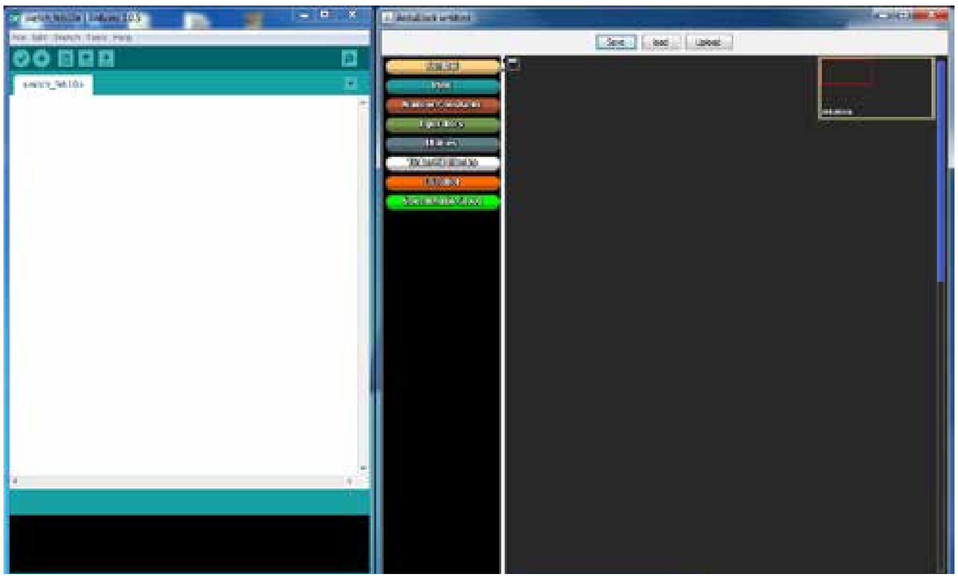
13. Under the "Tools" menu, select "ArduBlock"

14. From there you can start building a program by dragging code blocks from the menu on the left side of the ArduBlock window.

Programming the Robot- Arduino

To program the robot, you can use the Arduino Software and its graphical tool called ArduBlock.

Steps to create and save a program:



- Run Arduino (sketch window will open)
- Select Ardublock from Tools menu (Ardublock window will open)
- Create new program by using the Ardublock tools
- Save program on the laptop
- To load a saved program, select Load and browse for the folder and program name

Steps to compile and upload a program to your robot:

- Make sure robot power switch is turned Off
- Connect laptop to robot using USB cable
- Select Upload
  - System will convert Ardublock program to Arduino program and compile it
  - System will report any errors
  - If no errors found, system will load program to robot, overwriting the previous program and display the message: "Done uploading"
- To run the program, turn the robot power switch On. If the program commands the robot to drive, disconnect the robot from the laptop.

ArduBlocks Programming Tools

Control: lets you execute instructions based on certain conditions

- Loop – keep running the set of instructions (every program should be written inside a loop).
- If – run a set of instructions if the test condition is true.
- If else – run one set of instructions if the test condition is true, and another if it is false.
- While – run a set of instructions as long as the test condition is true.

Pin: lets you name and describe the pins on the Arduino board to the program

- Set digital pin – lets you assign a value to a digital pin
- Set analog pin – sets a value for a PWM (Pulse Width Modulates) digital pin
- Servo – lets you specify which pin controls a servo motor

Number/Constants: used to set up variables and give them initial values

- Set digital var – create a variable that takes only two values
- Set number var – create a numeric variable that can take many values
- Constants – a number, HIGH, LOW, TRUE, FALSE

Operators: where you find logical and mathematical operations

- Logical operators – and, or, not
- Mathematical operators - +, -, x, ÷, %, min, max, and a bunch more

Utilities: common functions you can use

- Delay – stops the program for ***# of milliseconds*** or ***microseconds***

Activities to Complete the Mission

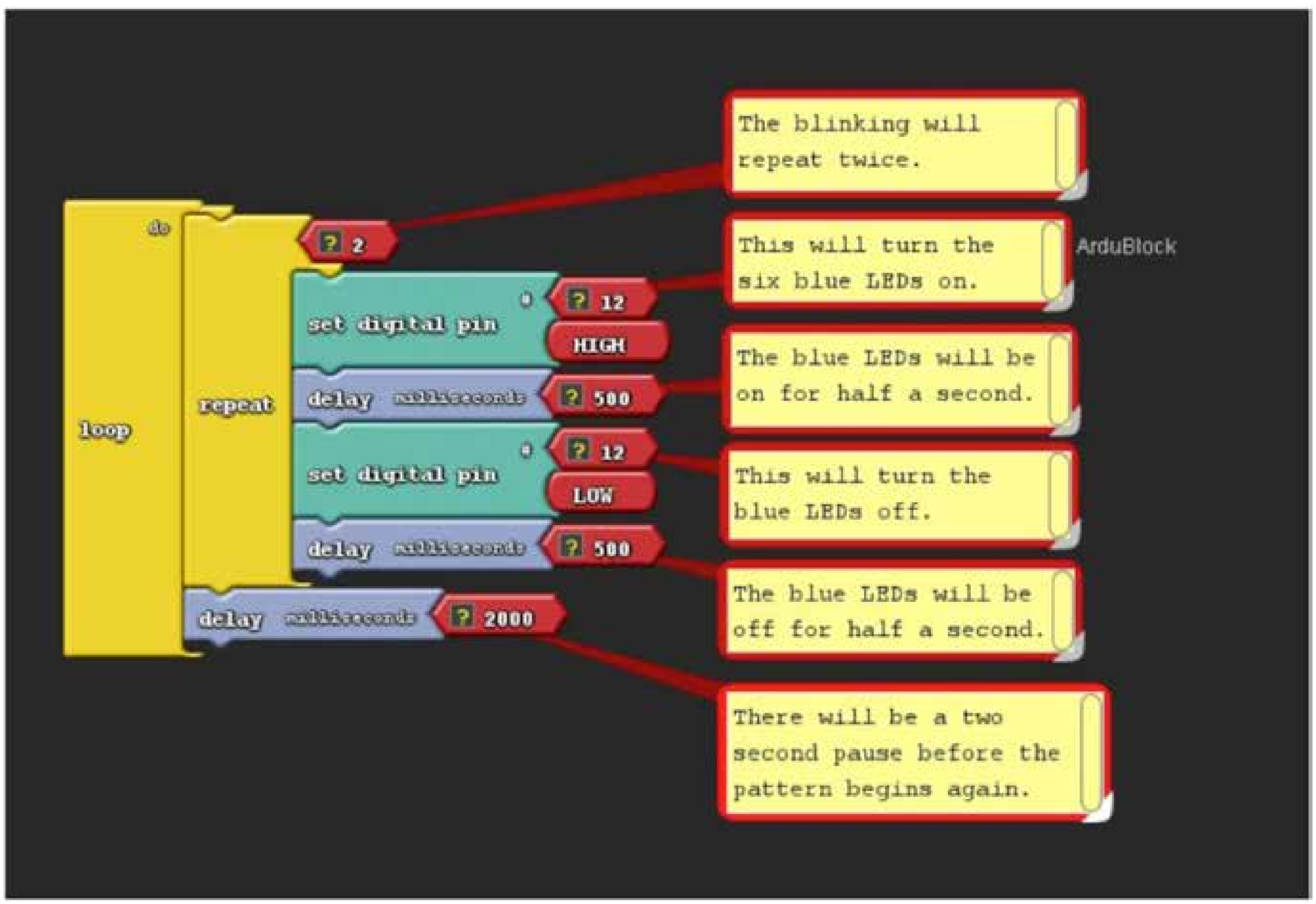
Like the engineers who've sent people to space and rovers to other planets before you, you'll take a step by step approach, proving each new capability and then adding to it until you have everything you need.

What steps do you think you should take to complete your robot programming mission?

(The following pages have a list of suggested steps–but you can make up your own plan and design your own activities).

Activity 1 – Blink Blink

Objective	How does this get used?
• Blink the 6 blue LED lights twice in a row. • Blink the 6 blue LED lights 4 times in a row.	Your prototype does not include the high gain antenna & transmitter for sending data back to Earth, so we will simulate the short/restricted messages using the LEDs.
What you need to know	**Extend / Challenge**
• Set Digital Pin block • Delay block • Which pins?	• Can you make it blink a pattern? • Decide on a different # of blinks for different sensors & values.



Activity 2 – Learning to Drive

Objective

- Drive forward for 3 seconds then back.

How does this get used?

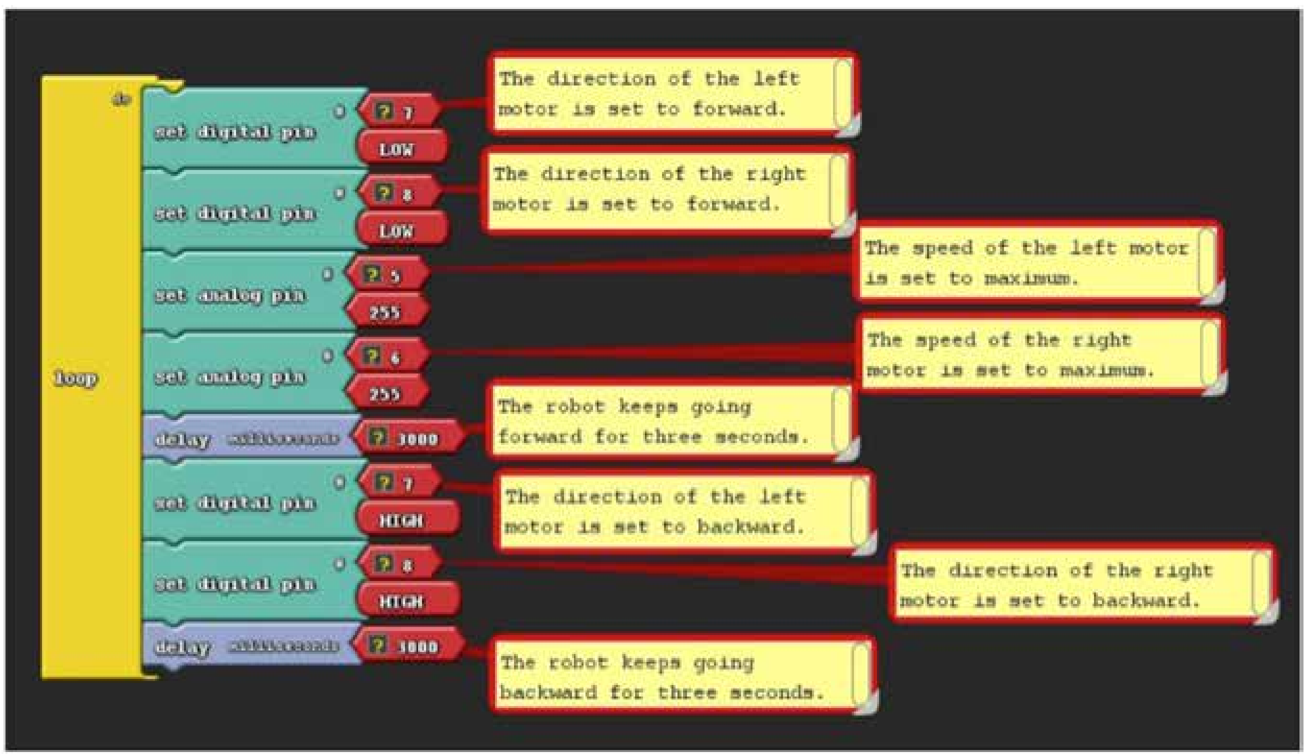
Your prototype will need to explore an area, moving forward and backwards.

What you need to know

- Which pins control the direction of the motors?
- Which pins control the speed of the motors?

Extend / Challenge

- Speed up and slow down.



Activity 3 – Learning to Turn

Objective

- Turn 90 degrees (one side driving)
- Rotate 90 degrees (one forward, one back)

How does this get used?

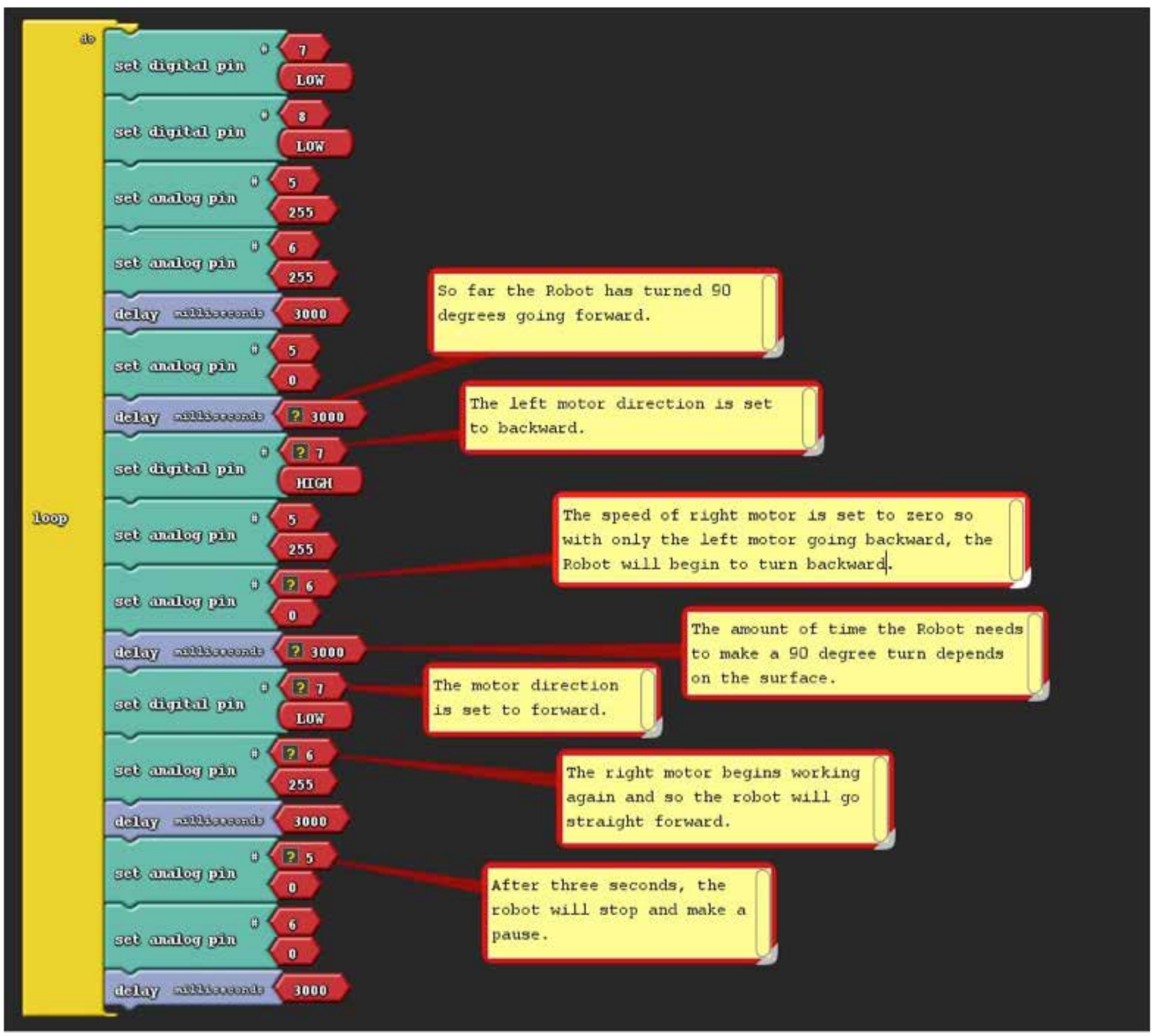
Your prototype will need to turn in a controlled manner to move from one search area to another

What you need to know

- Pins for track direction
- Pins for track motor speed
- How long does it take to rotate 90 degrees as opposed to some other angle?

Extend / Challenge

- Drive in a square.
- Was it easy to always get a square?



Activity 4 – How Far Did We Go?

Objective

- While driving forward, read the optical encoder.

How does this get used?

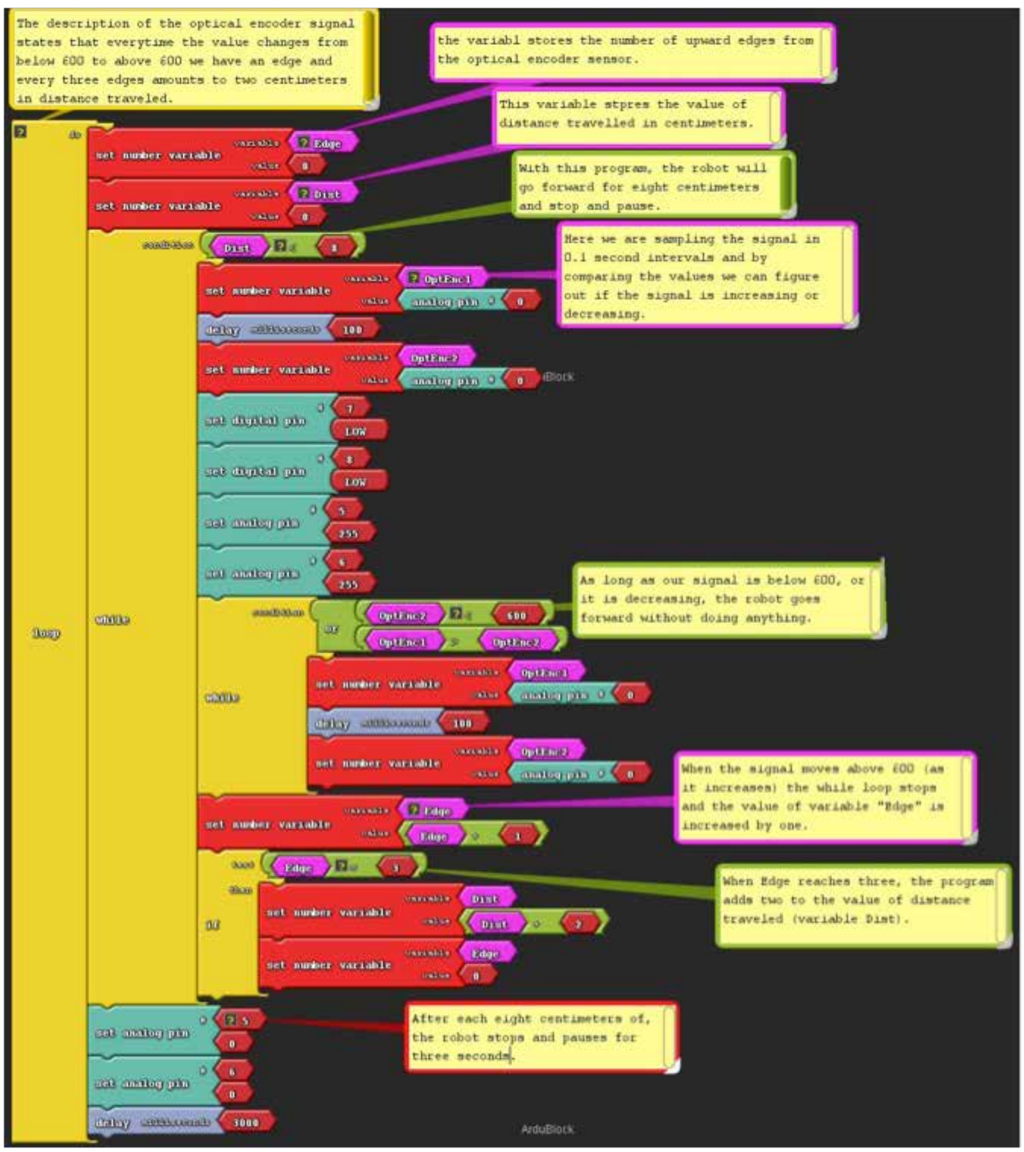
- Be able to drive a known distance and turn accurately to drive a search pattern (stripes or expanding square).

What you need to know

- How does the optical encoder work?
- Reading the analog input

Extend / Challenge

- Use the optical encoders to make a more accurate turn or rotate, rather than just using time.



Putting Together the Drive Subroutines

Rather than putting all the commands to make up driving, turning, and keeping track of distance into your main loop, you have created a block of subroutines to do those things that you can use over and over. In fact, you have even gone further and made them so that the robot will continue to drive while the program does other things like read a distance sensor or the sensors for moisture and hydrocarbons.

- The new operations (in subroutines) you have are:
- Drive (direction, distance)
- Turn (direction, angle)
- Rotate (direction, angle)
- Stop
- How Far? – Returns the distance since the start of the last "Drive" call
- Done? – Returns true if the last Drive, Turn or Rotate has completed

Activity 5 – Sensing Proximity

Objective

- To be able to detect the presence of an object ahead
- To stop before running into the object

How does this get used?

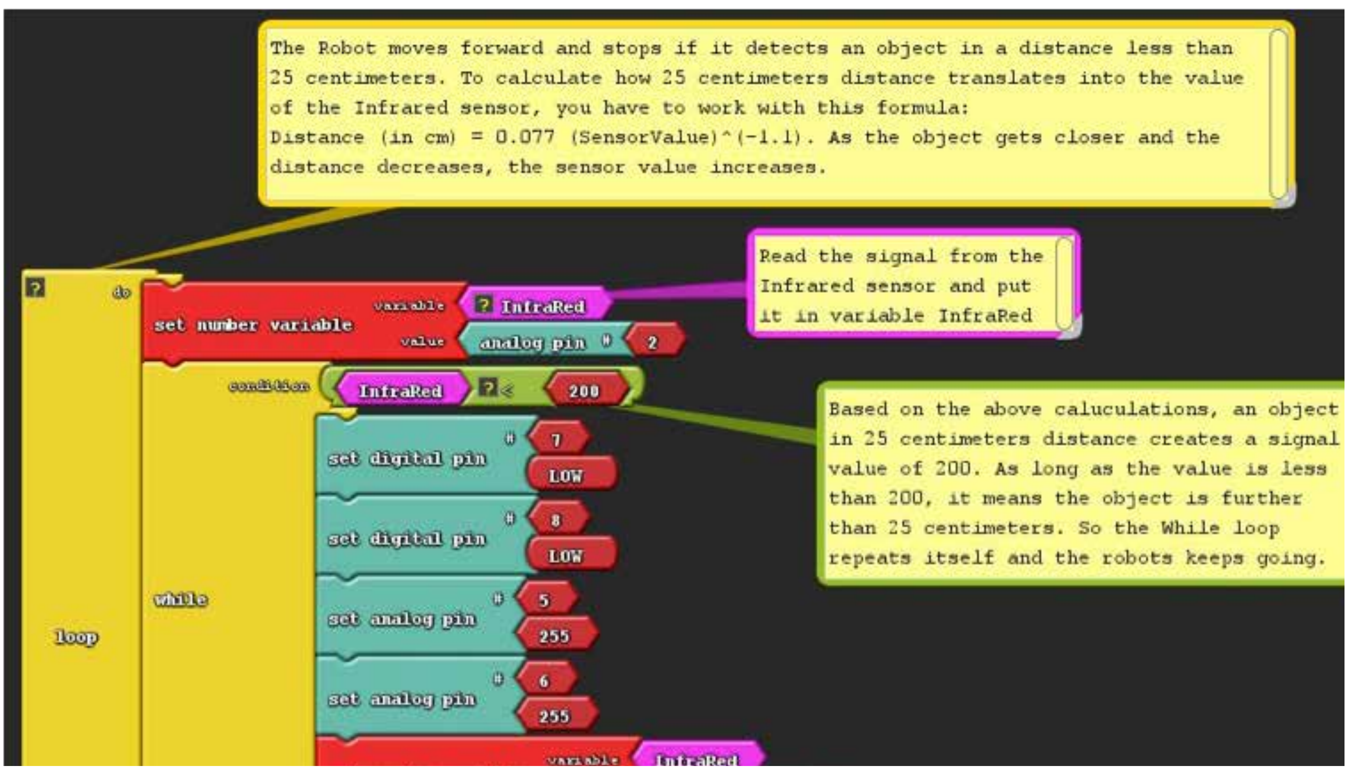
- You can expect that there will be unknown objects on the planet; your prototype needs to detect and go around them rather than crash and get damaged.

What you need to know

- Pin for infrared sensor
- How the infrared sensor works
- How to convert a value to a distance

Extend / Challenge

- Can you drive around an object in the field and pick up the same search pattern?



Activity 6 – Sensing Hydrocarbon

Objective

- Test for the presence of hydrocarbons in a particular location
- Blink blue LED lights according to test results

How does this get used?

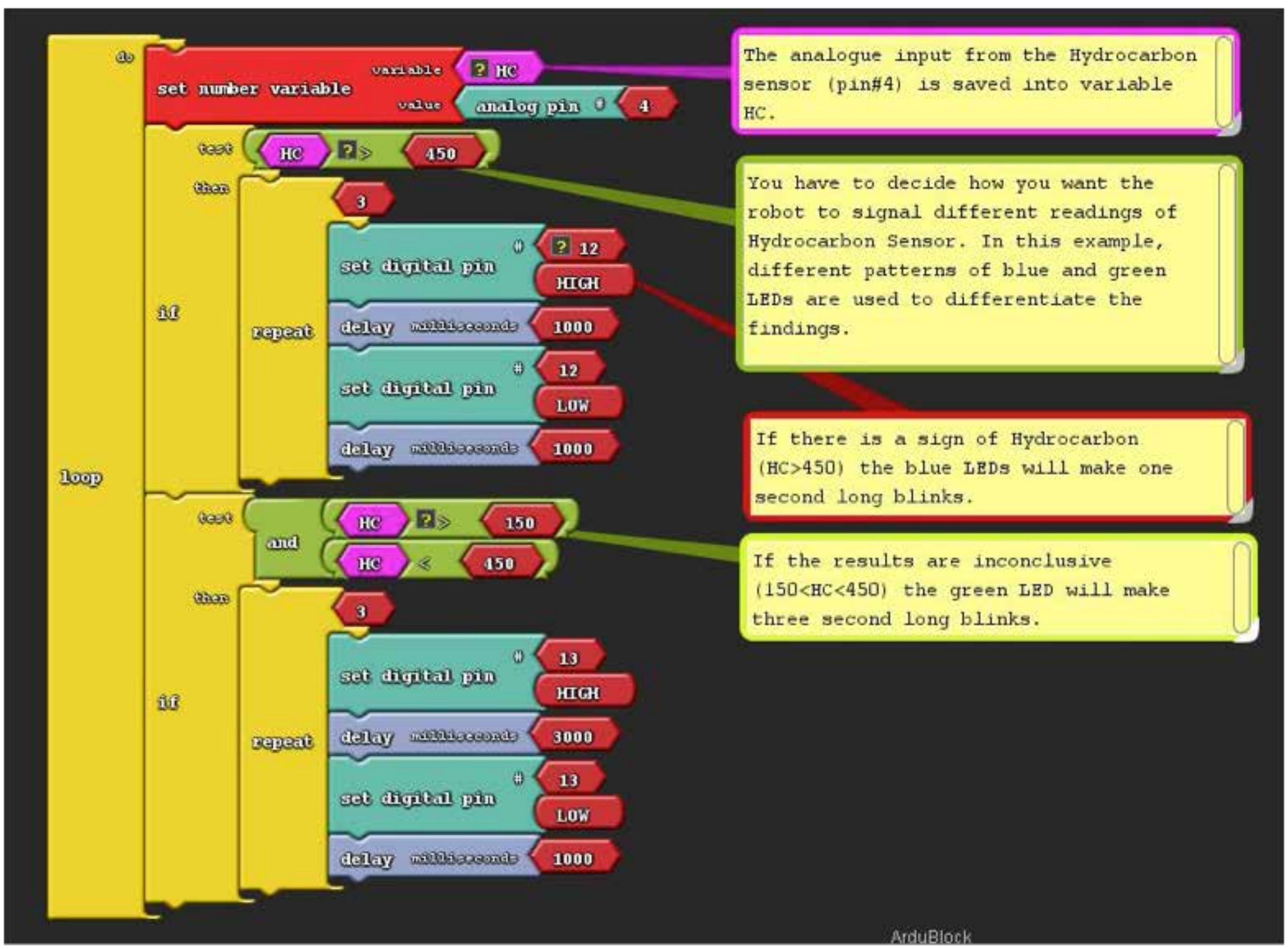
- At regular intervals during its exploration, your prototype should test for signs of life

What you need to know

- Pin for hydrocarbon sensor
- How does the sensor work?
- Range of values that would imply some form of hydrocarbon is present

Extend / Challenge

- Determine a message sequence that indicates the strength of hydrocarbon presence
- Over what size of area is it present?



Activity 7 – Sensing Moisture

Objective

- Lower the arm, and test for the presence of moisture in a particular location
- Blink blue LED lights based on the test results

How does this get used

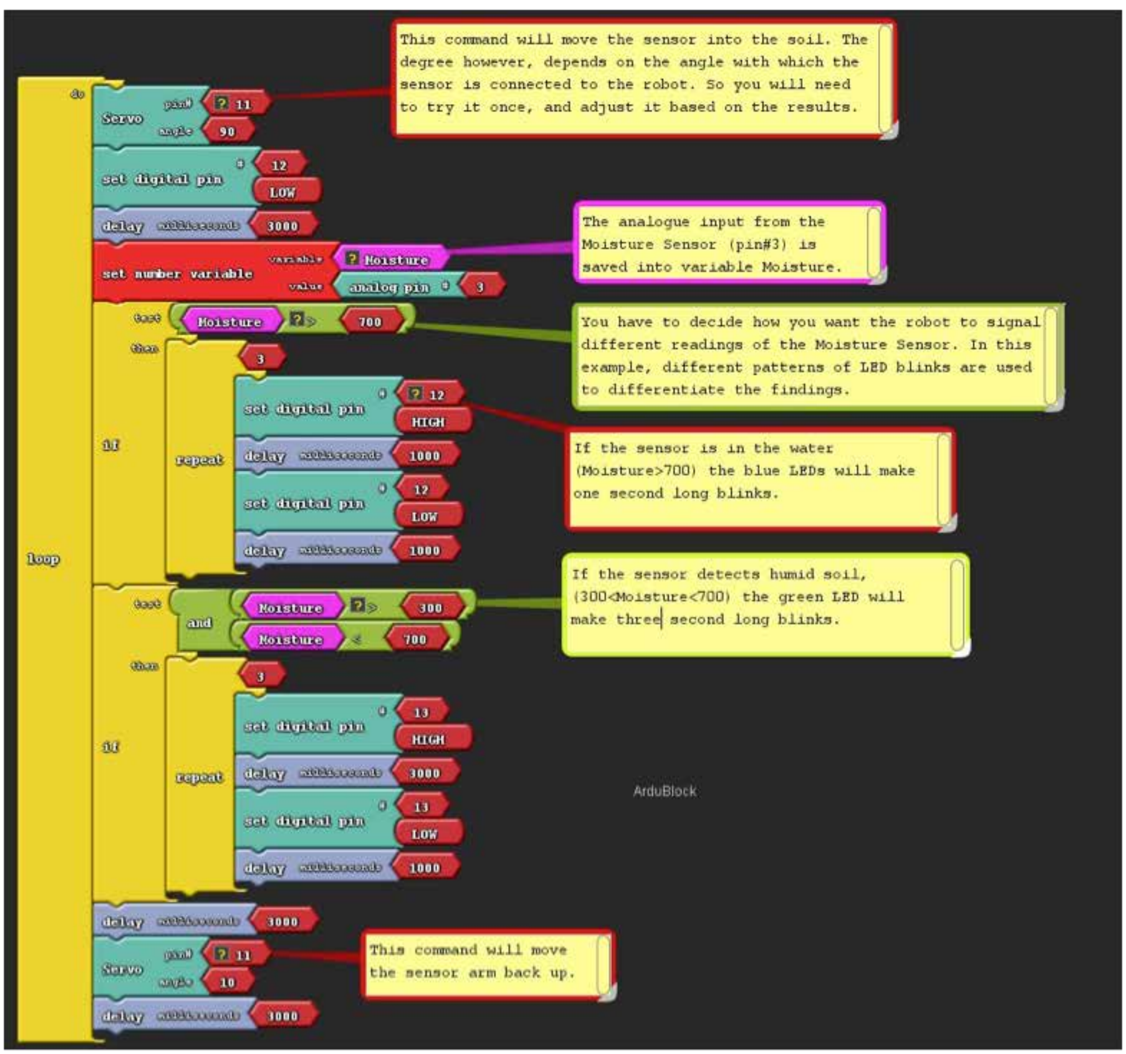
- Moisture is a necessity for life. Your prototype needs to sample the ground at regular intervals for the presence of moisture.

What you need to know

- Pin for moisture sensor
- How to raise and lower the arm to which moisture sensor is attached
- Sensor values which indicate dry, moist, very wet

Extend / Challenge

- Chart out an area which contains sufficient moisture to sustain life



Get out there and explore!

- Have your robot conduct a search of the test area.
- Find the presence of hydrocarbons and moisture and see your robot signal what it finds–all by itself, using your programmed instructions.

Challenge Activity

What did you find?

- Drive to multiple samples.
- Requires you to back up after taking a sample, then drive around an obstacle and resume your original pattern.