kineticvision
Excellence in Product and Process Development

Break Beam Sensor Integration with the Helium IoT Platform

1. Setting up the Atom

1.1. Install Helium Script on your development machine using the guide at https://dev.helium.com/guides/helium-script/#installing-helium-script

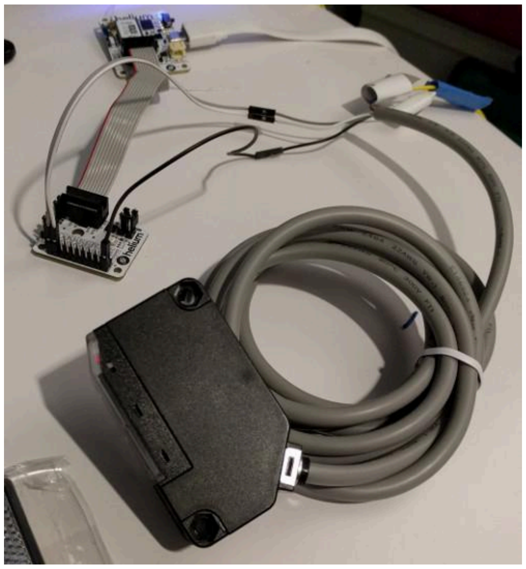
1.2. The command `helium-script --device` should return some information about your device, verifying that everything is hooked up correctly.

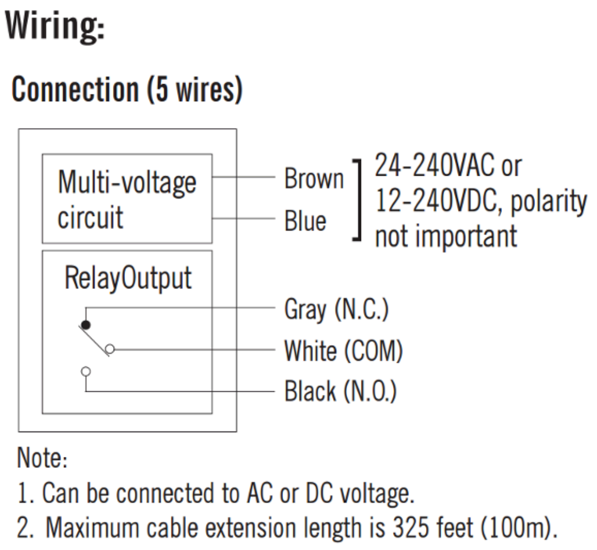
1.3. Helium Script is based on Lua 5.3, making the Lua reference manual a great resource when coding for the Atom: https://www.lua.org/manual/5.3/

1.4. Helium has some example code on their Github so we can just use their code to interface with the Digital Extension Board rather than writing our own from scratch.

1.5. Get the code for the SX1509 (the chip used by the Digital Extension Board) device from: https://github.com/helium/api-examples/tree/master/script/sx1509

2. Connecting the Break Beam Sensor to the Atom





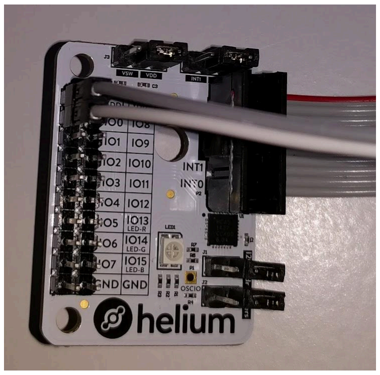
Note:
1. Can be connected to AC or DC voltage.
2. Maximum cable extension length is 325 feet (100m).

2.1. Connect the Helium Digital Extension Board to the Helium Atom Development Board using the included ribbon cable.

2.2. Hook up the break beam sensor as shown below (Black to GND, Gray to VDD, White to IO0).

2.3. Confirm the USB is plugged into the Atom board and the 12V adapter is plugged into the wall.

2.4. Once you are at this point, you should be able to see an LED on the front of the break beam sensor. You should also be able to toggle the sensor using the included reflector. Below is a detailed image of the helium digital board connections.



3. Writing code for the Atom

3.1. The code in `main.lua` (from https://github.com/helium/api-examples/tree/master/script/sx1509) will initialize the chip and set the LED to a random color every half second. Let's start by changing the color of the LED to indicate the status of the break beam.

3.2. Set pin 0 as an input. Looking at line 22, Helium is already setting pin 1 as an input. Since we are using pin 0, change the 1 on lines 22 & 24 to a 0 and pin 0 is now an input.

3.3. Turn off the LEDs to avoid a bright flash of the wrong color at the start. The LED on this board is CMY, so to turn it off set i, j and k to 0xFF on line 28-30 (equivalent to #FFFFFF which is the CMY hex code for black).

3.4. If someone breaks the beam, the he.wait command on line 37 will return immediately and events will evaluate to true.

3.5. Let's set the color of the LED based on if the beam is broken rather than setting it randomly. To check the status of the break beam sensor, read the value of pin 0. This can be done with the digital:read_pin command found in `sx1509.lua`

3.5.1.
```
if events then
  if digital:read_pin(0) then
    i = 0x00
    j = 0x88
    k = 0xff
  else
    i = 0x00
    j = 0xff
    k = 0xff
  end
end
```

3.6. Upload the program to the Helium Atom with `helium-script.exe -upm main.lua sx1509.lua` and load the program by pressing the RST button on the board.

3.7. If everything is working correctly, the LED on the Digital Extension Board should match the LED on the break beam sensor.

4. Sending and receiving data

4.1. Now that we can get the status of the break beam sensor, let's log that data to Helium. This will allow us to track how many times the beam is broken over a longer period of time and analyze it. Maybe we want to know how many cars go down a particular street in a day or how busy that street is at certain times of day.

4.2. Logging a value with Helium is actually pretty simple. `he.send(port, time, type, value)` is all we need to do and Helium takes care of the rest. Name the port with any string, for this tutorial let's choose something descriptive like "beam". The time that the beam was broken is conveniently already in the variable `now`. For the type and value, we'll just send a 1 or a 0 so `i` (for integer) will work fine. Adding an `he.send` to each case in our code above we end up with:

4.2.1.
```
if events then
  if digital:read_pin(0) then
    he.send("beam", now, "i", 1)
    i = 0x00
    j = 0x88
    k = 0xff
  else
    he.send("beam", now, "i", 0)
    i = 0x00
    j = 0xff
    k = 0xff
  end
```

```
end
```

4.3. To actually run this code on the Helium Atom, use `helium-script -upm main.lua sx1509.lua`

4.4. To view the data from Helium, please visit dashboard.helium.com

4.5. To pull this data into your application, check out the API available at https://dev.helium.com/cloud-api

5. Connecting Helium data to the wider world

5.1. Once sensor data is stored in the Helium cloud it can be integrated into applications through several methods. There is a command line interface(CLI), REST API, and a Node-Red instance on the Helium dashboard site (alpha feature). Documentation for the CLI and API are located https://dev.helium.com/.

5.2. To create a node-red flow that sends an alert (sms or email) each time that the beam sensor is triggered follow the steps below:

5.2.1. Go to https://dashboard.helium.com/flows.

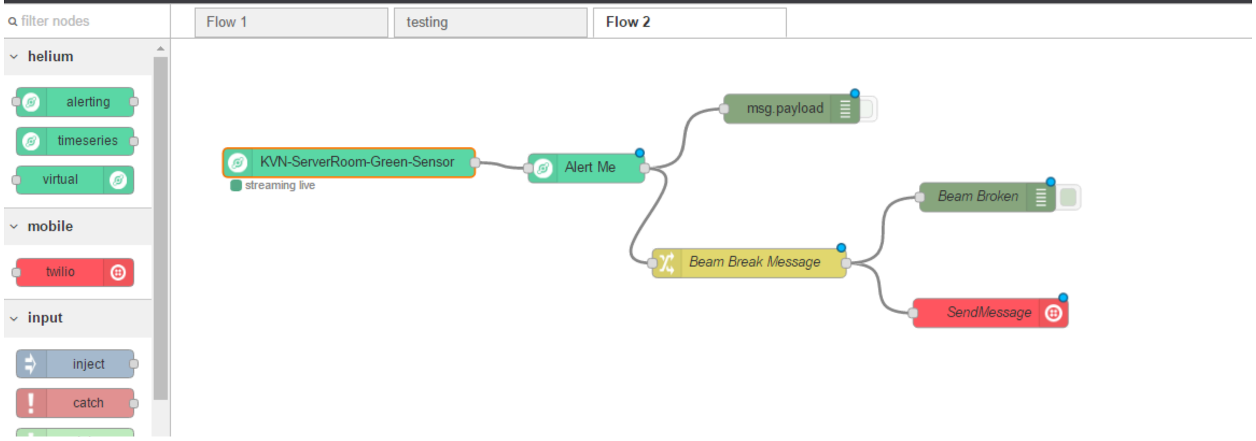
5.2.2. Drag the timeseries Helium node onto the canvas and double click to open the configuration window. Set the Data Source to be an individual sensor and then select the desired sensor.

5.2.3. Add an alerting Helium node onto the canvas and connect it to the timeseries node. Configure this node to send an alert if the value on port 'beam' is over 0.

5.2.4. The alerting node will pass on its payload whenever the sensor is triggered but to send a custom message add a switch function node to the canvas and connect it to the the alerting node. Configure the switch function node with the desired message in the msg.payload topic.

5.2.5. Add a twilio or email node to the canvas and route the output of the switch node to it.Configure this node with the appropriate email or phone number.

5.2.6. Below is an image of the complete workflow along with a debug node.



6. Conclusion

6.1. In this tutorial, we took a fairly "dumb" sensor (a break beam that just acts as a relay) and hooked it up to the Helium environment and a messaging service fairly easily . If you have any questions or want to checkout some other types of sensors, please come see us at the Kinetic Vision & Helium sponsor table.