

Installing ARTIK / ARTIK Cloud nodes

First, ensure Node-RED is installed on your ARTIK board. If you don't have Node-RED installed on your board, please run the two commands below from your board console.

```
#dnf install npm
#npm install -g node-red
```

Once you have Node-RED installed, install ARTIK and ARTIK Cloud custom nodes:

```
#npm install -g node-red-contrib-artik node-red-contrib-artik-cloud
```

Start your Node-RED, and you can see the ARTIK/ARTIK Cloud nodes listed in the nodes container.



As you can see, we have 4 custom nodes to interface with ARTIK 5 and 10 I/Os and 2 custom nodes for ARTIK Cloud integration.

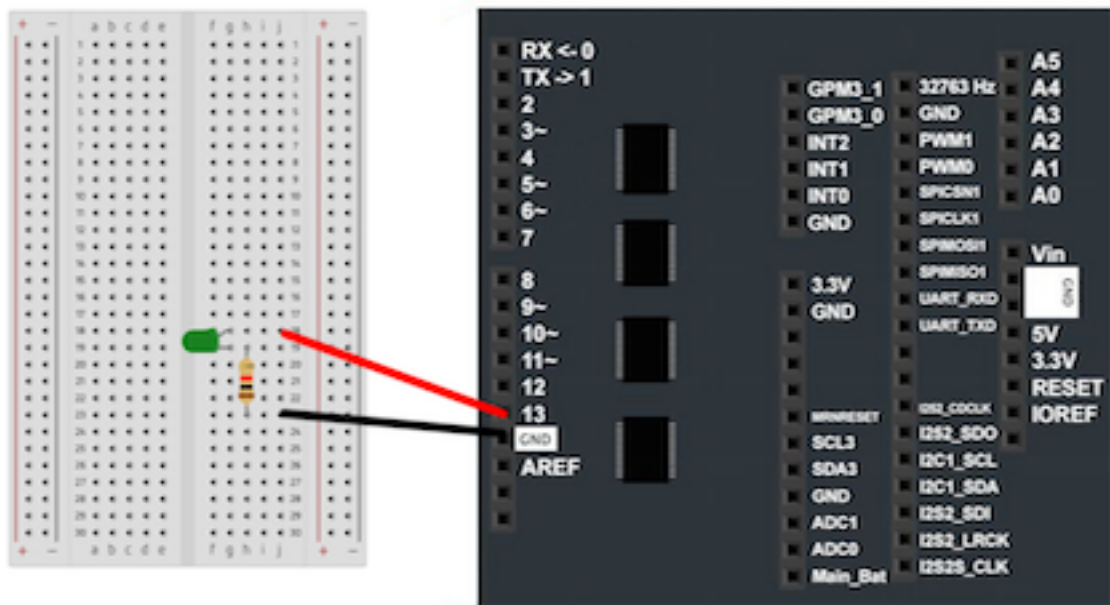
In this section we will show you how to use the ARTIK and ARTIK Cloud custom nodes to build IoT applications. These samples are illustrated by using ARTIK 5, but are fully portable to ARTIK 10.

Developing by using ARTIK / ARTIK Cloud nodes

Example 1: Blink an LED (artik out Node)

We will first show you how to “blink an LED” by using **artik out** node. **artik out** node is a node that sets a GPIO pin in the OUT direction and can also set the output state to HIGH or LOW on ARTIK.

Building the circuit



Developing Node-RED flow

1. Drag an “artik out” node to the canvas, double click to configure it. In the Edit dialog, we select “ARTIK 5” as our target platform, then choose “Pin 13” as our digital output pin.

As shown in the circuit, turning Pin 13 to “High” will turn on the LED, and “Low” will turn it off. Here in this node, we set Pin 13’s Initial State to “Low” and State to “High”. When Node-RED flow reaches this node, LED should be turned on.

Name the node as “OutputHigh”.

Edit artik out node

Cancel Done

Name OutputHigh

Platform Artik 5

Pin Pin 13

State High

☒ Set initial state?

Initial State Low

2. We need another “artik out” node to set Pin 13’s State to “Low” so we can toggle the LED.

Edit artik out node

Cancel Done

Name OutputLow

Platform Artik 5

Pin Pin 13

State Low

☐ Set initial state?

3. Now, we add an inject node and a function node to our flow.



In inject node, we configure to repeat our flow every 1 second, and name the node as “Init”.

Edit inject node

Cancel

Done

✉ Payload

▼ timestamp

📄 Topic

🔄 Repeat

interval

every

1

seconds

☒ Inject once at start?

🏷 Name

Init

Note: "interval between times" and "at a specific time" will use cron.
See info box for details.

In function node, we set the number of outputs to be 2, so we can alternate between OutputHigh and OutputLow functions every second. Name the node as “ToggleLED”.

In this function, we use a global variable ‘state’ to track the state of Pin 13 output. If its current value is 0 (LOW), we trigger the execution of OutputHigh function to turn on the LED and set ‘state’ value to 1 (HIGH). Otherwise, we trigger the execution of OutputLow function to turn off the LED and set ‘state’ value to 0 (LOW).

```
var state = global.get('state')||0;

if (state === 0) {
  global.set('state',1);
  return [msg, null];
} else {
  global.set('state',0);
  return [null, msg];
}

return msg;
```

Edit function node

Cancel

Done

Name

ToggleLED

Function

```
1 var state = global.get('state') || 0;
2
3 if (state === 0) {
4   global.set('state', 1);
5   return [msg, null];
6 } else {
7   global.set('state', 0);
8   return [null, msg];
9 }
10
11 return msg;
```

Outputs

2

Here is what the final flow looks like:

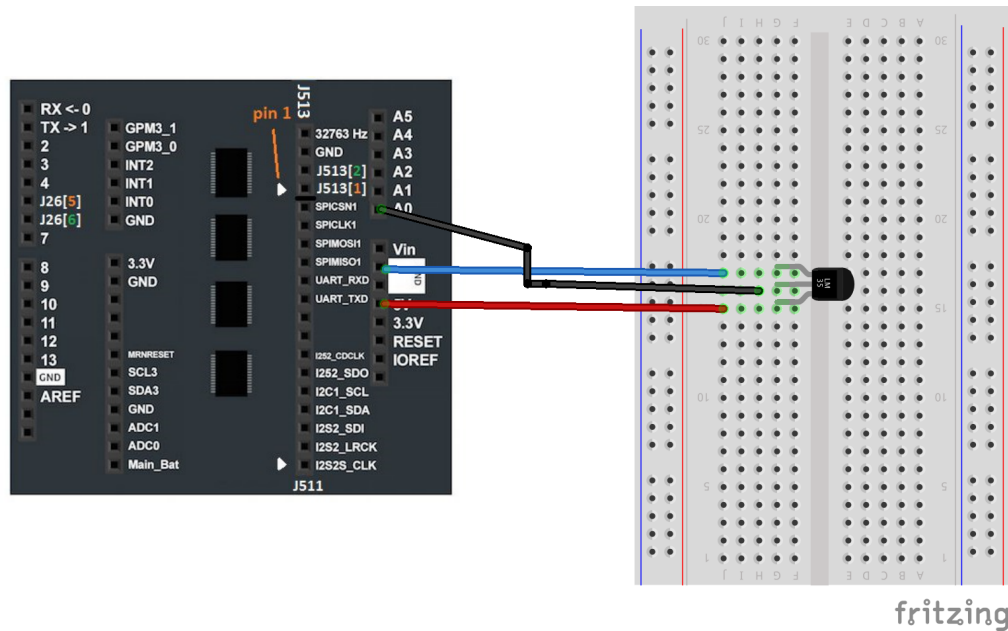


Deploy the flow, and you should see your LED blinks.

Example 2: Stream temperature data to ARTIK Cloud

In the 2nd example, we are going to use **artik adc** node and **artik cloud out** node to collect analog sensor data and stream the data to ARTIK Cloud.

Building the circuit



Developing Node-RED flow

1. Drag an **artik adc** node to the canvas. On ARTIK 5 board, we have a temperature sensor connected to analog pin 0 on header J25. Select “ARTK 5 ” as the target platform and “ADC 0” as the analog pin# where we read the data from.

Edit artik adc node

CancelDone

NameReadSensor

PinADC 0

PlatformArtik 5

2. Connects a function node to the right side of artik adc node, where we will convert the voltage reading from ADC 0 pin to a temperature. Name the node as “ConvertToTemperature”. The function is defined like below:

```

var voltage_raw = msg.payload;
var voltage = voltage_raw * 2 * 0.439453125;

//Converting from 10mv per degree with 500mV offset
var temperatureC = (voltage - 0.5) * 100 ;
var temperatureF = (temperatureC * 9.0 / 5.0) + 32.0;
msg.payload = {
  "temperature": temperatureF
};
return msg;

```

3. Wire up an inject node and a debug node to the beginning and end of the flow. In “inject” node, we configure to read temperature sensor data every 10 seconds, name the node as “Init”.

In “debug” node, we simply show msg.payload info on Node-RED Debug panel. Name the node as “DebugMsg”.

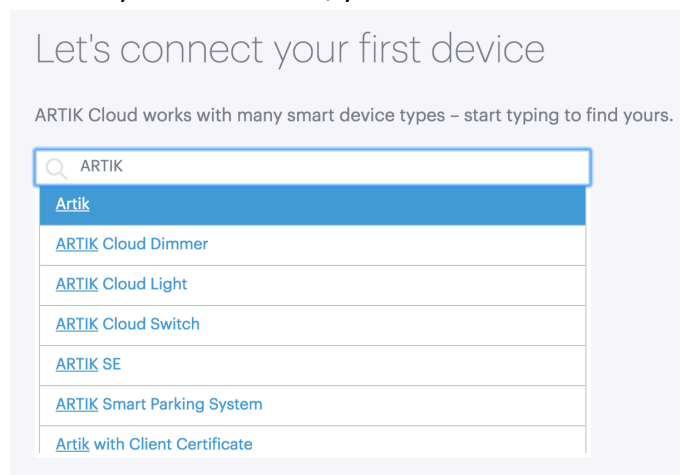


Now, we can see the temperature updated on the debug panel every 10 seconds.

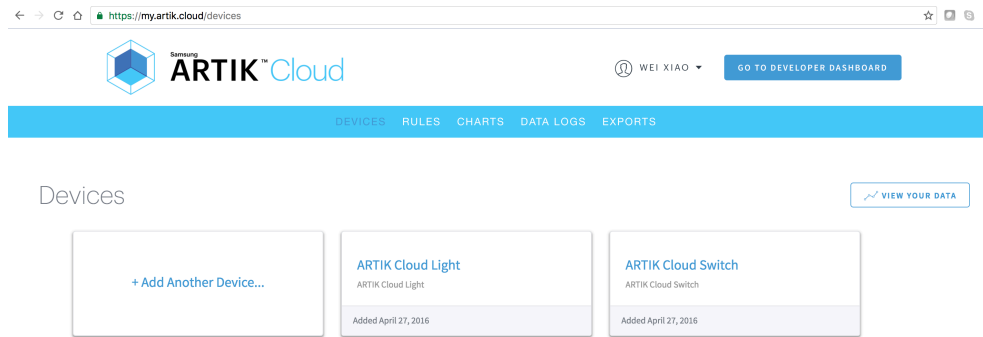
4. Next step, we will stream temperature to ARTIK Cloud. In order to do this, we need to create a temperature device instance in ARTIK Cloud.

4.1 Log into ARTIK Cloud user portal <https://artik.cloud>.

If this is your first device, you will be re-directed to the screen below:



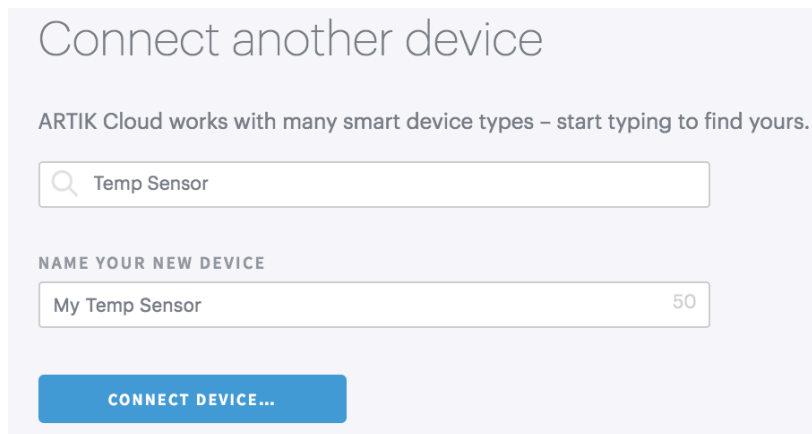
Otherwise, Under “MY ARTIK CLOUD”/”DEVICES”, you will see all you connected devices.



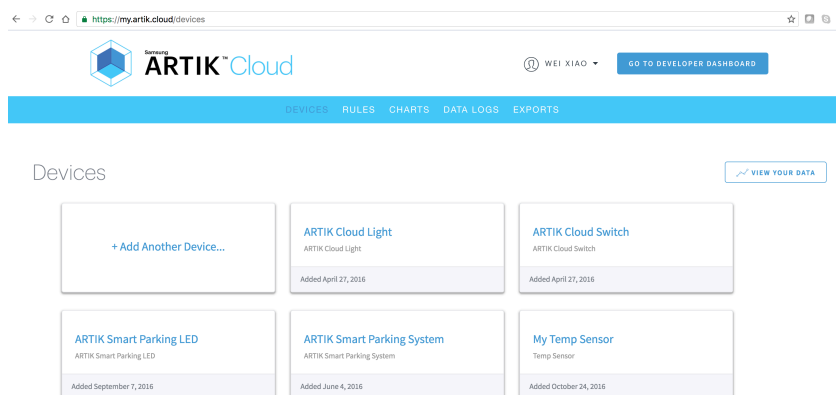
Click “Add Another Device...” link.

4.2 Search for “Temp Sensor” and select it.

“Temp Sensor” is a public device type which you can inherit. Name your instance as “My Temp Sensor”.



4.3 Click the “CONNECT DEVICE...” button. A new device will be created.



- 4.4 Click on newly created “My Temp Sensor”, you will see the Device Info popup, which shows your Device Type, Device ID, Device name etc. details. Click the “GENERATE DEVICE TOKEN...” link to generate a device token.

Device Info

DEVICE TYPE

Temp Sensor

CONNECTED SINCE

October 24, 2016

LAST DATA TRANSFER

Never

DEVICE ID

4b53d6fcce284b7db7cb7568c75ae6a4

DEVICE TYPE ID

dt8888ced85b524b81a991d712b77433f6

NAME

My Temp Sensor50

DEVICE TOKEN

d8a8b519586e4f0dad7963fe0db506c3

REVOKE TOKEN

SAVE CHANGES

DELETE

5. Now, let’s go back to our Node-RED flow and plug an artik cloud out node to into our existing flow. In the artik cloud out node, we will enter our Temp Sensor instance device ID and device Token.

Edit artik cloud node

Cancel

Done

Name

ARTIK Cloud

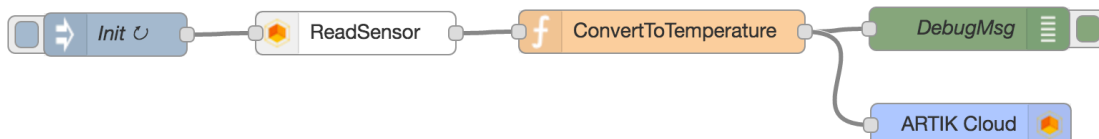
Device ID

4b53d6fcce284b7db7cb7568c75ae6a4

Device Token

.....

The final flow looks like:

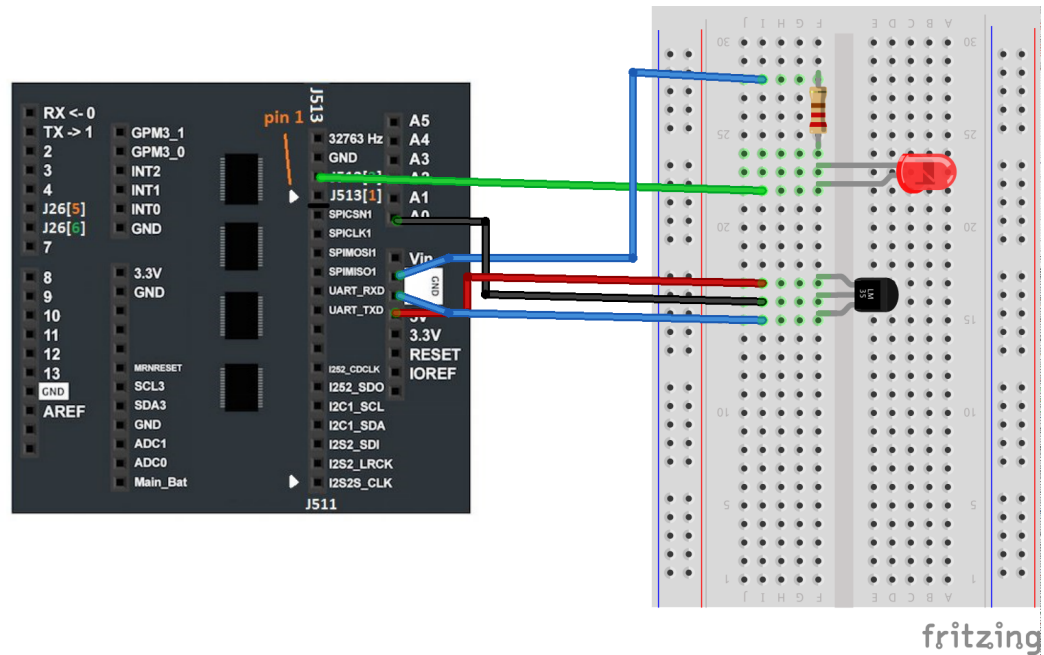


6. Go back to ARTIK Cloud user portal, you can visualize the streamed temperature data.

Example 3: Use Rules Engine to control LED

In this example, we will extend what we developed in Example 2. We will have the temperature sensor and an LED connected to our ARTIK 5 board, and use ARTIK Cloud Rules Engine to trigger cross-device actions.

Building the circuit



Adding Rules in ARTIK Cloud

1. From ARTIK Cloud developer portal, we create a new device “ARTIK LED”, which has a Manifest like below:

Device Fields	Device Actions	Activate Manifest						
Describe fields for each piece of data produced by this device.	Describe actions that this device is capable of receiving.	Publish this device manifest on the ARTIK Cloud platform.						
Your manifest is ready to be activated and does not require approval before going live. Activating this manifest will not make your device type public.								
Fields		Actions						
<table border="1"><tr><td>LED</td><td>Boolean</td></tr></table>		LED	Boolean	<table border="1"><tr><td>setOn</td><td>Action</td></tr><tr><td>setOff</td><td>Action</td></tr></table>	setOn	Action	setOff	Action
LED	Boolean							
setOn	Action							
setOff	Action							

We will use its “LED” field to track if the LED is ON or OFF, and trigger setOn, setOff actions by using Rules Engine.

2. Create an instance of “ARTIK LED” device from ARTIK Cloud developer portal by following the same steps from Example 2. Name your device instance “My ARTIK LED”, and get a copy of your device ID and device token.
3. As the next step, we will add rules so “My Temp Sensor” can turn on or off “My ARTIK LED” when certain conditions are met.

Go to ARTIK Cloud user portal, select Rules, and click “+NEW RULE “ button. Or first rule is:

```
IF
My Temp Sensor temperature is more than 80
THEN
Send to ARTIK LED the action setOn
```

Choose device activity to monitor

IF

My Temp Sensor: temperature X is more than 80

+ NEW CONDITION

Send actions to your devices

THEN

My ARTIK LED: setOn X

+ NEW ACTION

Describe your rule

RULE TITLE

Blink LED when temperature is higher than 80 degree 13

DESCRIPTION

IF My Temp Sensor temperature is more than 80
THEN send to My ARTIK LED the action setOn

1312

Add a 2nd rule:

IF

My Temp Sensor temperature is less than or equal to 80

THEN

Send to ARTIK LED the action setOff

Choose device activity to monitor

IF

My Temp Sensor: temperature X is less than or equal to 80

+ NEW CONDITION

Send actions to your devices

THEN

My ARTIK LED: setOff X

+ NEW ACTION

Describe your rule

RULE TITLE

Turn off LED when temperature is lower than/equal to 80 degree 2

DESCRIPTION

IF My Temp Sensor temperature is less than or equal to 80
THEN send to My ARTIK LED the action setOff

1310

Developing Node-RED flow

1. Drag an “artik cloud in” node and a “debug” node to the canvas, and wire them together.
2. artik cloud in node is capable of receiving both messages and actions. In this example, we will configure it to receive actions triggered by “My Temp Sensor”. Double click the node, select “Receive Actions”, and enter your “My ARTIK LED” device’s device ID and token. Name the node as “ARTIK Cloud In”.

In debug node, change the output to “complete msg object”.

Edit artik cloud node

CancelDone

Name

ARTIK Cloud In

☐ Receive Messages

☒ Receive Actions

Device ID

1874179c6aec450592b6292688f2f224

Token

.....

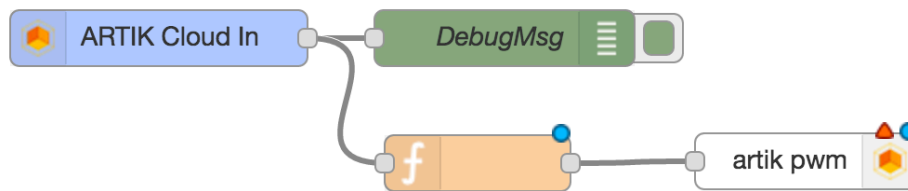
ARTIK Cloud In DebugMsg

3. Now warm up the temperature sensor, and when the temperature fluctuates, you should be able to see debug messages like below from the Debug Panel:

```
10/28/2016, 4:41:16 PM  DebugMsg
msg : Object
{ "actions": [ { "name": "setOff", "parameters": {} } ], "_msgid": "ba078778.45f878" }

10/28/2016, 4:41:35 PM  DebugMsg
msg : Object
{ "actions": [ { "name": "setOn", "parameters": {} } ], "_msgid": "4f464d14.b0b9b4" }
```

4. Drag a function node and an artik pwm node to the canvas and put them under “DebugMsg” node.



In “function” node, we check if we received a “setOn” or “setOff” action. When LED is ‘setOn’, we define msg.payload.state to be 1, msg.payload.dutyCycle as 500000000

```
var actions = msg.actions;
var action = actions[0].name;

if (action === 'setOn') {
  msg.payload = {
    "state": 1,
    "dutyCycle": 500000000,
    "peroid": 1000000000,
  };
} else if (action === 'setOff') {
  msg.payload = {
    "state": 0
  };
}
return msg;
```

(500ms), and msg.payload.peroid as 1000000000 (1s), and use them as PWM parameters for artik pwm node.

When LED is ‘setOff’, we simply set msg.payload.state to 0 to turn off the PWM output. In ‘artik pwm’ node, our configuration looks like below:

Edit artik pwm node

Cancel

Done

Name

LED

Pin

PWM 0

Duty cycle

Duty cycle

Period

Period

State

High

☒ Set initial state?

Initial State

Low

Duty cycle

Initial duty cycle

Period

Initial period

Now, when your temperature reaches over 80 degree, you should see your LED blinks.