Integrating Cayenne on TTN

RM1xx Series

Application Note v1.0

# Intro duction

Cayenne is an IoT data processing system that can display transmitted data in visualized form with drag-and-drop configuration. A LoRaWAN server (e.g. The Things Network - TTN) can be set up so that it receives data from end-devices and forwards that data to an external application server in the format it can understand. This app note shows to capture the data such as temperature on the DVK-RM191 and send it over to Cayenne in order to show data in icon and graph formats.

# Requirement

* DVK-RM816 or DVK-RM191 with the latest firmware (v101.5.0.9 was used in this test with RM191)
* [UwTerminalX](https://github.com/LairdCP/UwTerminalX) (v1.09a or later recommended)
* smartBASIC application ([cayenne.mydevice.sb](https://raw.githubusercontent.com/LairdCP/RM1xx-Applications/master/cayenne.mydevice.sb))
* LoRaWAN gateway (e.g. [Laird Sentrius RG1xx](https://www.lairdtech.com/products/rg1xx-lora-gateway))

# Overview

This application note demonstrates that RM1xx captures data on its DVK and sends it over a LoRa network. The gateway will be set up as packet forwarder pointing to TTN as destination, and TTN will be configured to redirect data from end-devices to Cayenne MyDevice so that it can be displayed in widgets on a browser.

# Test Setup

First, DVK needs to be set so that RM1xx can access the temperature sensor and Button1, Button2 and LED5. ([Figure 1](#Figure1) and [Figure 2](#Figure2))

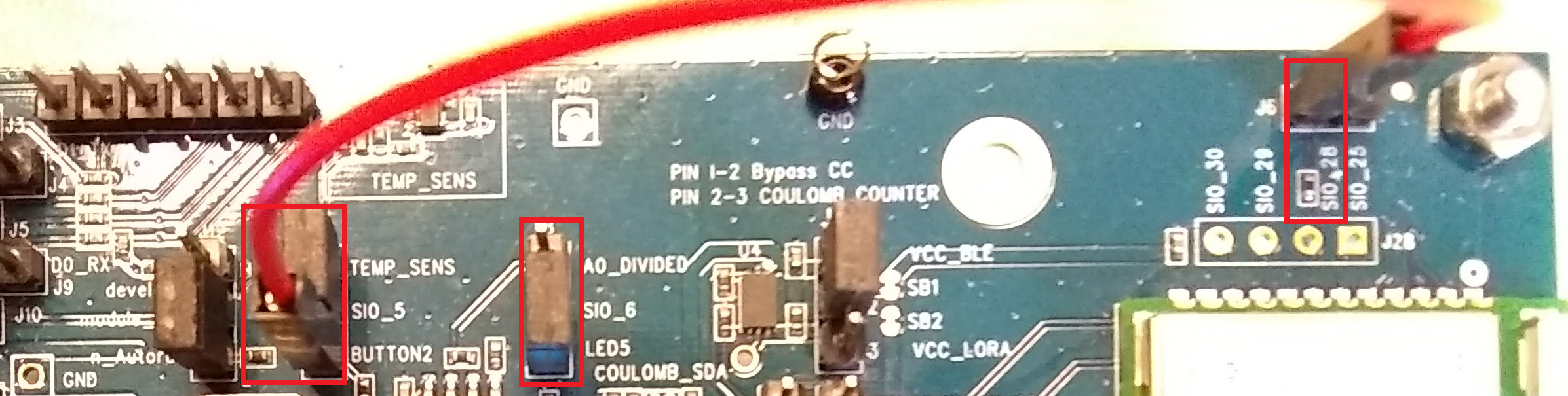


Figure Hardware pin setup (J7pin1-2, J7 pin3 - SIO\_28, J8 pin 1-2)

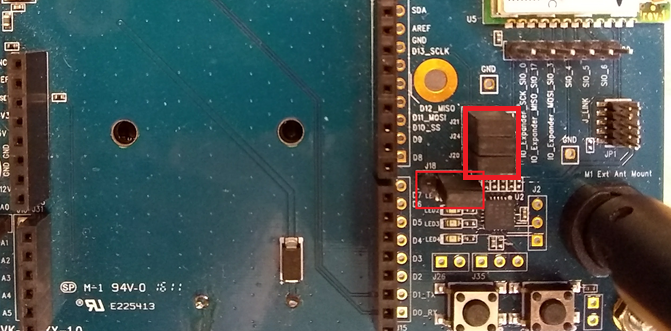


Figure Hardware pin setup (J18 pin 2-3, J20, J21, J24)

Now, TTN and Cayenne need to be set up.

1. Register for an account on cayenne <https://cayenne.mydevices.com/cayenne/dashboard/start> and verify your account
2. Sign up for a TTN account and login on <https://console.thethingsnetwork.org/>
3. Set up your gateway, application, end-devices on TTN

* <https://www.thethingsnetwork.org/docs/gateways/registration.html>
* <https://www.thethingsnetwork.org/docs/applications/add.html>
* <https://www.thethingsnetwork.org/docs/devices/registration.html>

1. Add a new device on Cayenne. LoRa > The Things Network > Cayenne LPP, paste the dev EUI here from TTN and add it
2. On TTN, go back to the application > integrations tab, and then click add integration and select cayenne, then put the process ID you get from the Cayenne page URL after the /lora/ part of the URL (for example, the highlighted part in <https://cayenne.mydevices.com/cayenne/dashboard/lora/3e795080-xxxx-xxxx-xxxx-51a105d3afc2>)
3. Go to TTN’s application payload formats tab and change it to ‘Cayenne LPP’ from custom.

Then, RM1xx needs to be set up.

1. Open up UwTerminalX and configure the AppEUI, DevEUI and AppKey with the values from TTN by using the following command respectively for OTAA. (For using ABP, refer to [LoraWAN Keys/ID document](https://assets.lairdtech.com/home/brandworld/files/LoRaWAN%20Keys%20and%20IDs%20Overview.pdf))

* at+cfgex 1010 "<AppEUI>"
* at+cfgex 1011 "<DevEUI>"
* at+cfgex 1012 "<AppKey>"

1. (RM191-US and RM191-AU only) Set up a sub-band to be used with at+cfg 1001 and at+cfg 1002 For example, use the following to use sub-band 2

* at+cfg 1001 2 (This sets the sub-band 2 among available sub-band options ranging from 1 to 8)
* at+cfg 1002 1 (This decides what key a sub-band can be set up with, as shown in [Table 1](#Table1). Alternatively, at+cfgex 1009 can be used to set a channel mask as shown in [Table 2](#Table2). In this case, at+cfg 1002 2 should be used to go along with it)

Table 1 Channel Map select type

|  |  |
| --- | --- |
| **at+cfg 1002** | **Action** |
| 0 (default) | Stack default – all channels enabled |
| 1 | Use at+cfg 1001 |
| 2 | Use at+cfgex 1009 |

Table 2 ChannelMask commands

| Sub-Band | Frequency Range (MHz) | | Channels | Command |
| --- | --- | --- | --- | --- |
| **US** | **AU** |
| 1 | 902.3–903.7 | 915.2-916.6 | 0-7 | at+cfgex 1009 "000100000000000000ff" |
| 2 | 903.9–905.3 | 916.8-918.2 | 8-15 | at+cfgex 1009 "0002000000000000ff00" |
| 3 | 905.5–906.9 | 918.4-919.8 | 16-23 | at+cfgex 1009 "00040000000000ff0000" |
| 4 | 907.1–908.5 | 920.0-921.4 | 24-31 | at+cfgex 1009 "000800000000ff000000" |
| 5 | 908.7–910.1 | 921.6-923.0 | 32-39 | at+cfgex 1009 "0010000000ff00000000" |
| 6 | 910.3–911.7 | 923.2-924.6 | 40-47 | at+cfgex 1009 "00200000ff0000000000" |
| 7 | 911.9–913.3 | 924.8-926.2 | 48-55 | at+cfgex 1009 "004000ff000000000000" |
| 8 | 915.5–914.9 | 926.4-927.8 | 56-63 | at+cfgex 1009 "0080ff00000000000000" |
| All bands | 902.3–914.9 | 915.9-917.1 | 0-63 | at+cfgex 1009 "00ffffffffffffffffff" |

1. Reset via “atz”
2. If ABP is chosen to be used in step #1, cayenne.mydevice.sb needs to be modified as described in comment below. If running with OTAA, skip this step



1. Right-click on UwTerminalX and click “XCompile + Load + Run”, and choose the smartBASIC application (cayenne.mydevice.sb which can be downloaded from <https://github.com/LairdCP/RM1xx-Applications>)
2. If successfully downloaded, the application starts immediately. RM1xx joins a LoRaWAN network and then transmits data shown on Cayenne periodically

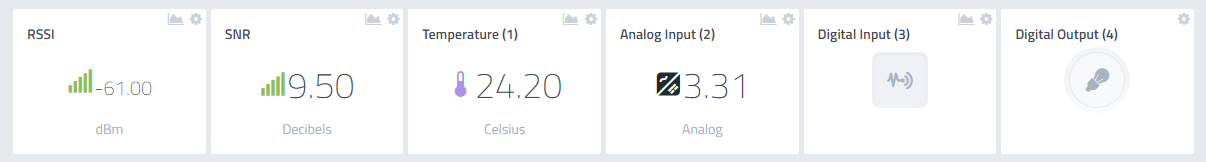


Figure Displayed data on Cayenne

In figure 3, “Analog Input (2)” is the power supply/battery in voltage. “Digital input (3)” is Button1 on the DVK and “Digital Output (4)” is the LED5 status on the board. These names can be changed on setting located at the right upper corner of each widget so that they can look like figure 4, for example.

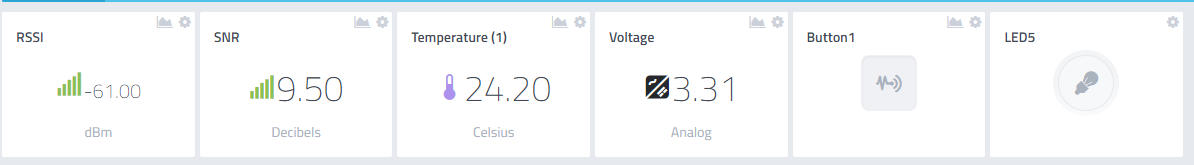


Figure Renamed widgets on Cayenne

If Button2 on DVK is clicked, it will toggle LED5 status and include it in the transmitted data to Cayenne.

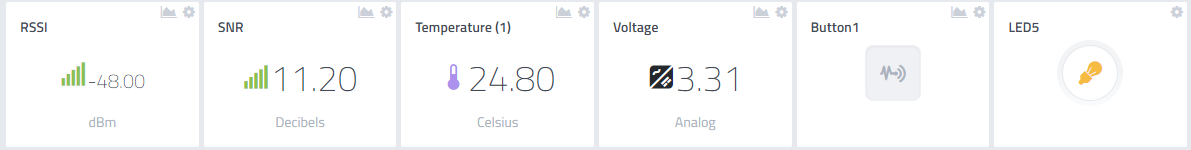


Figure LED5 turned on

If Button1 is being pressed while RM1xx is transmitting data, Button1 status will be updated on Cayenne.

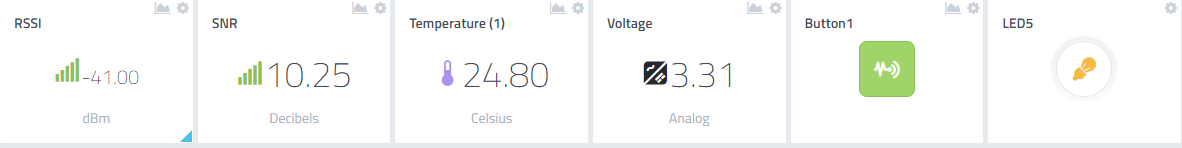


Figure Button1 pressed

You can set notification on “trigger” which is located on right corner of each widget and get an email or text message when certain condition is met. The below is a setup for notification when temperature is above 20oC.

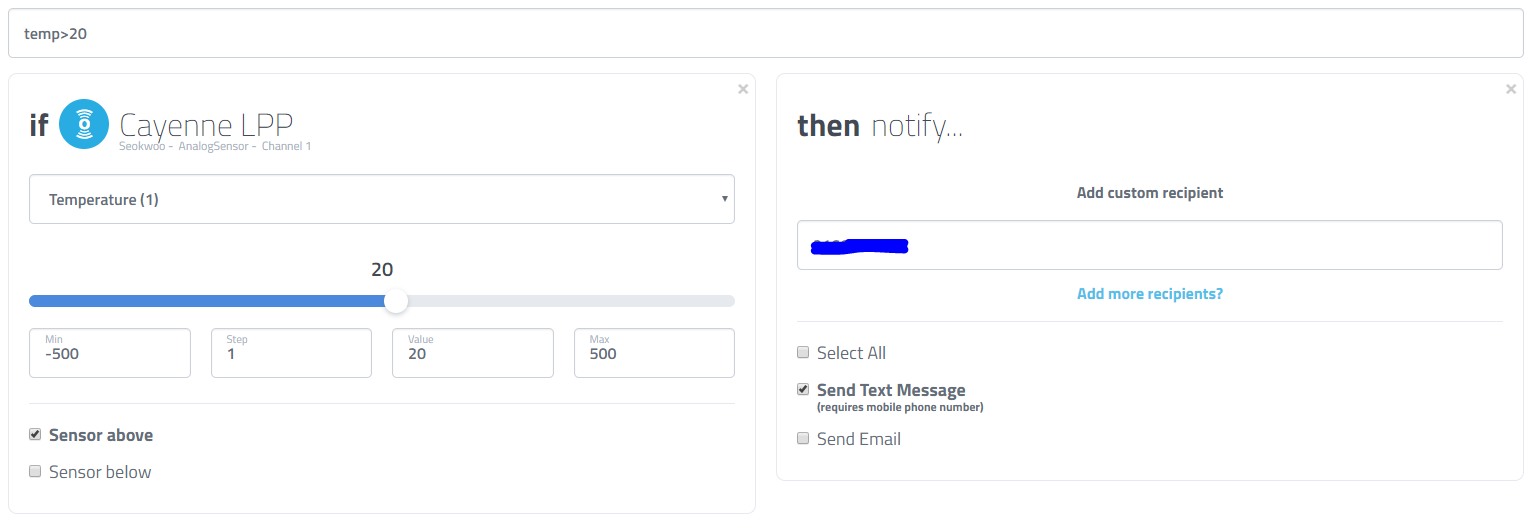


Figure Notification setup for temperature above 20oC

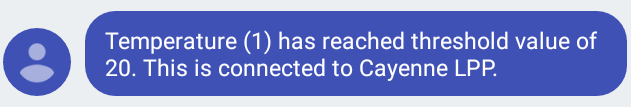


Figure Arrived text message on mobile phone

# Cayenne data format

In cayenne.mydevice.sb, the LoraPost function explains how data should be formatted before it is sent to Cayenne server. Data for each entity (e.g. for sensor) consists of three parts like following.

* First byte: data channel – individually classifies each sensor in a frame
* Second byte: data type (e.g. temperature)
* N bytes after second byte: data value where N (the size of data) varies across different data type

For example, the temperature sensor data is comprised of tempType (“\01\67”) and tempVal (in hex) in the application. Here, 0x01 means that 01 of data channel is used. 0x67 means that temperature sensor is the type of data to be used and the following data is the value for tempVal as formatted in the required Cayenne data-type, as seen in Table 3.

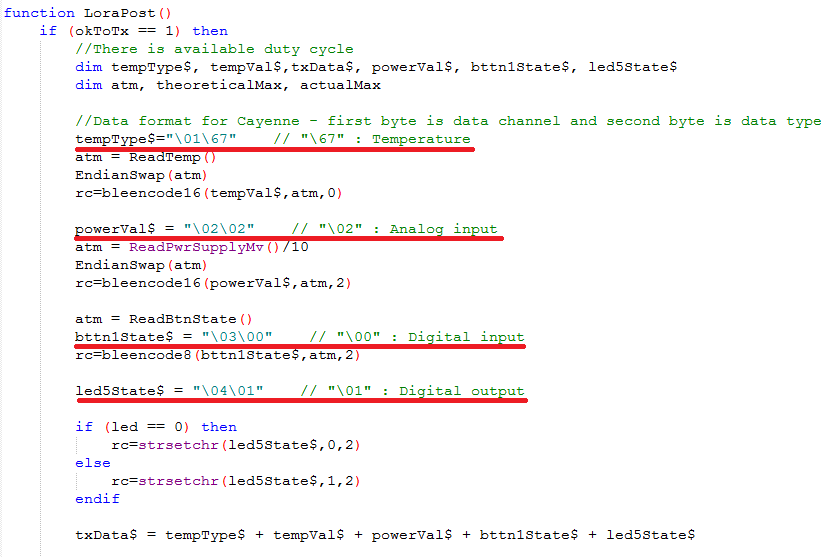


Table 3 Cayenne Data Type

|  |  |  |
| --- | --- | --- |
| Type | Data type (Hex) | Data size(Bytes) |
| Digital Input | 0 | 1 |
| Digital Output | 1 | 1 |
| Analog Input | 2 | 2 |
| Analog Output | 3 | 2 |
| Illuminance Sensor | 65 | 2 |
| Presence Sensor | 66 | 1 |
| Temperature Sensor | 67 | 2 |
| Humidity Sensor | 68 | 1 |
| Accelerometer | 71 | 6 |
| Barometer | 73 | 2 |
| Gyrometer | 86 | 6 |
| GPS Location | 88 | 9 |

[Cayenne Payload Structure](https://mydevices.com/cayenne/docs/lora/" \l "lora-cayenne-low-power-payload) reveals a full table for data format of each type.

# Resources

Cayenne Payload Structure - <https://mydevices.com/cayenne/docs/lora/#lora-cayenne-low-power-payload>

RM1xx Setup Guides – <http://www.lairdtech.com/products/rm1xx-lora-modules#documentation-tab>

RM1xx Sample Applications – <https://github.com/LairdCP/RM1xx-Applications>

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Notes | Approver |
| 1.0 | 12 Oct 2017 | Initial Release | Seokwoo Yoon |
|  |  |  |  |