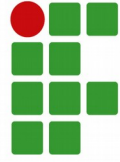
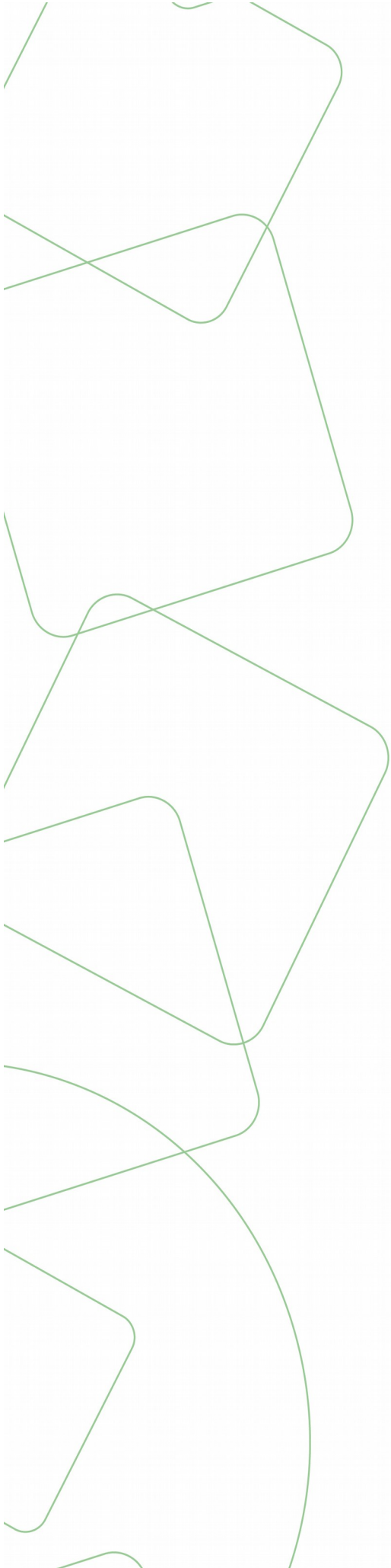


**INSTITUTO  
FEDERAL**  
Santa Catarina

**SENSORIAMENTO REMOTO DAS CONDIÇÕES AMBIENTAIS DE COLMEIAS  
DE ABELHAS UTILIZANDO RF:**

**Projeto RFabelhas  
Manual de Instruções**

**Julho/2018**



**INSTITUTO  
FEDERAL**  
Santa Catarina



**MINISTÉRIO DA EDUCAÇÃO**  
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA  
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA

# **Projeto RFabelhas**

## **Manual de Instruções**

São José – Julho 2018

**Coordenador(a)**

Prof. Marcos Moecke

**Membros da Equipe**

prof. Ramon Mayor Martins

prof. Cleber Jorge Amaral

Bolsista: Kristhine Schaeffer Fertig

Bolsista: Tamara Arrigoni

Bolsista: Natalia Miranda

Bolsista: Yan Lucas Martins

Colaborador voluntário: João Pedro Menegali Salvan Bittencourt

**Edital:**

CHAMADA PÚBLICA FAPESC No 08/2016

**Financiamento:**

FAPESC (TO 2017TR000466)

**Apoio:**

EPAGRI CIRAM  
IFSC Campus São José

## Projeto Rfabelhas

### MANUAL DE INSTRUÇÕES

---

Criando um nó LoRa:.....	6
Criando um Gateway LoRa na TTN:.....	7
Sensores utilizados nas colmeias.....	8
Leitura dos dados na TTN e a publicação deles nos dois bancos de dados do InfluxDB e na EPAGRI.....	10
Visualização dos dados usando Grafana.....	11

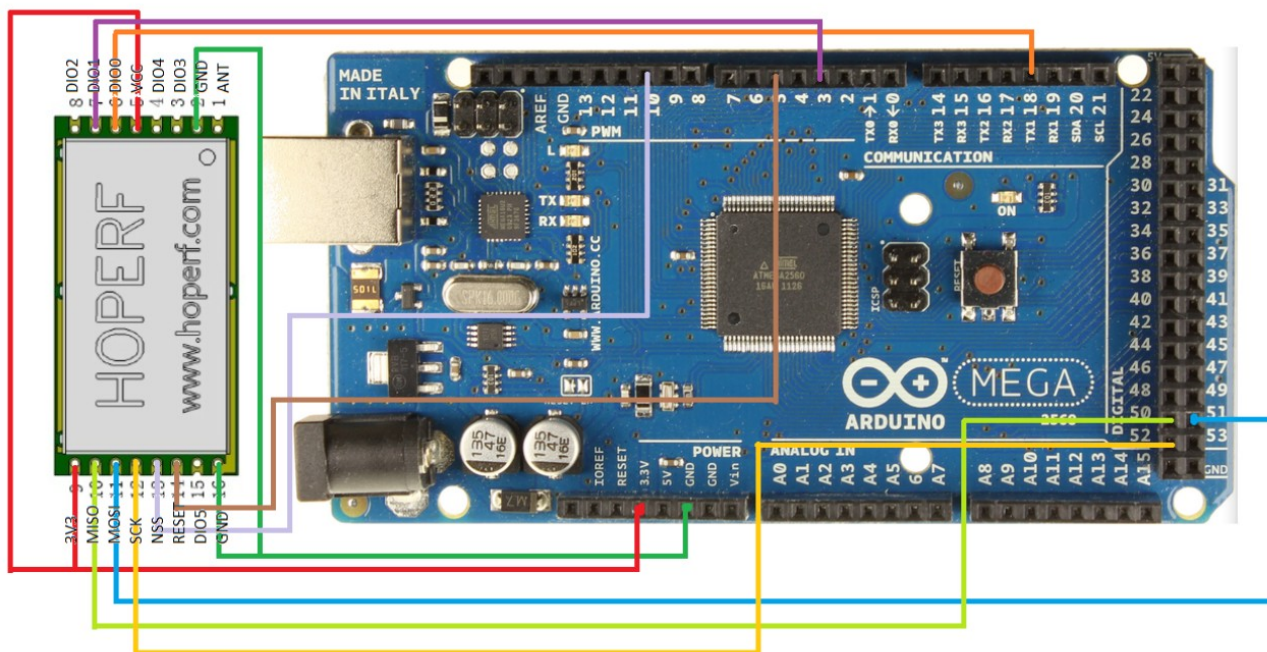
---

# Criando um nó LoRa:

Utilizando um arduino Mega + RFM95PW



Seguir o esquema de ligação ou a tabela seguinte:



	RFM95PW	ARDUINO MEGA 2560
<b>3v3</b>	<b>9</b>	<b>OPEN</b>
<b>VCC</b>	<b>5</b>	<b>VCC (5Vd)dc</b>
<b>GND</b>	16 e 2	GND
<b>MOSI</b>	11	51
<b>MISO</b>	10	50
<b>SCK</b>	12	52
<b>NSS</b>	<b>13</b>	<b>6</b>
<b>RESE T</b>	14	5
<b>DIO0</b>	6	18
<b>DIO1</b>	7	3
<b>ANT</b>	<b>1</b>	<b>ANTENA</b>

# Criando um Gateway LoRa na TTN:

- Em <https://account.thethingsnetwork.org/register> e cadastre um usuário e senha
- Depois de criada a conta, entre em <https://console.thethingsnetwork.org/gateways> e em “Register gateway”

Gateways > Register

## REGISTER GATEWAY

**Gateway EUI**  
The EUI of the gateway as read from the LoRa module

AA AA AA FF FF BB BB BB 8 bytes

**I'm using the legacy packet forwarder**  
Select this if you are using the legacy [Semtech packet forwarder](#).

**Description**  
A human-readable description of the gateway

Gateway Teste

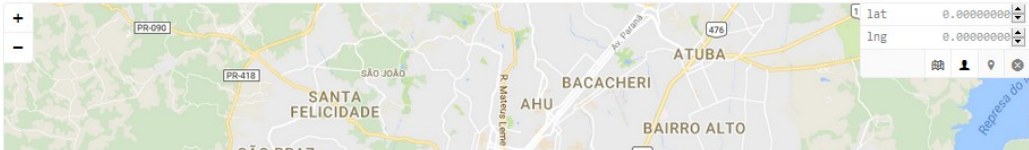
**Frequency Plan**  
The [frequency plan](#) this gateway will use

United States 915MHz

**Router**  
The router this gateway will connect to. To reduce latency, pick a router that is in a region which is close to the location of the router itself.

ttn-router-brazil

**Location**  
The exact location of you gateway. This will be used if your gateway cannot determine its location by itself. Set a location by clicking on the map.



lat 0.00000000  
lng 0.00000000

# Sensores utilizados nas colmeias

## Sensor de temperatura e umidade relativa

Está sendo utilizado o [Sensor de Umidade e Temperatura AM2302 DHT22](#). Este sensor tem uma resolução de 0.1 % e acurácia (exatidão) de  $\pm 2\%$  no sensor de humidade relativa e uma resolução de 0.1 % e acurácia (exatidão) de  $\pm 0.5\%$  no sensor de temperatura. Esse sensor utiliza os pinos (ver AM2302 Pin assignments):

- 1 - VDD - Power (3.3V-5.5V)
- 2 - SDA - Serial data, bidirectional port
- 3 - NC - Empty
- 4 - GND - Ground

## sensor de temperatura e umidade relativa interna

Atualmente o pino 2 desse sensor está conectado a porta digital PD8 do Arduino.

## sensor de temperatura e umidade relativa externa

Atualmente o pino 2 desse sensor está conectado a porta digital PD7 do Arduino.

## Sensor de iluminância

Está sendo utilizado o [Sensor de Luz BH1750FVI Lux](#). Este sensor utiliza uma interface I2C para a comunicações, usando os seguintes pinos:

- 1 - VDD - Power (3.3V-5.5V)
- 2 - GND - Ground
- 3 - SCL - Clock line
- 4 - SDA - Serial data, bidirectional port
- 5 - ADDR - Slave address selector

Atualmente os pinos da comunicação I2C ([Wire.h](#)) utiliza os pinos 20 (SDA) e 21 (SCL) no caso do Arduino Mega.

A definição do endereço do dispositivo é feito pelo terminal ADDR.

ADDR = 'H' =>  $\text{bx}1011100 = 0x5C$

ADDR = 'L' =>  $\text{bx}0100011 = 0x23$

## Dúvida?

Qual é o endereço atualmente utilizado?

Para detectar o endereço do dispositivo na I2C pode ser usado o programa [I2C Scanner](#).

## Sensor de peso

Para a medição do peso da colmeia estão sendo usadas 4 [células de carga de até 50kg](#) em conjunto com o [Módulo Conversor HX711](#). As células de carga estão ligadas em ponte de Wheatstone. A precisão do conversor analógico-digital (ADC) é de 24 bits. Esse sensor utiliza os pinos:

- 1 - VDD - Power (2.7 ~ 5.5V)
- 2 - PD\_SCK - Power down control (high active) and serial clock input



- 3 - DOUT - Serial data output
- 4 - GND - Ground

Atualmente o pino 2 desse sensor está conectado a PD9 e o pino 3 a PD10 do Arduino.

## Real Time Clock

Para permitir a ativação sincronizada das medições e transmissões, usamos o [Real Time Clock RTC DS3231](#). Ele dispõe de um sensor de temperatura e do RTC. Este RTC utiliza uma interface I2C para a comunicações, usando os seguintes pinos:

- 1 - GND - Ground
- 2 - VDD - Power (3.3V-5.5V)
- 3 - SDA - Serial data, bidirectional port
- 4 - SCL - Clock line
- 5 - SQW -
- 6 - 32k -

<https://howtomechatronics.com/tutorials/arduino/arduino-ds3231-real-time-clock-tutorial/>

## Bibliotecas necessárias

Para compilar o código do Arduino será necessário incluir as bibliotecas correspondentes. Um bom repositório para as bibliotecas é <https://www.arduinolibraries.info/> Será necessário importar:

- [DHT sensor library](#) DHT\_sensor\_library-1.3.0.zip
- [BH1750FVI](#) BH1750FVI-1.0.0.zip
- [BH1750 master github](#)
- [Queuetue HX711 Library](#) Queuetue\_HX711\_Library-1.0.1.zip
- [HX711\\_ADC](#) HX711\_ADC-1.0.2.zip
- [HX711 master github](#)

## **Leitura dos dados na TTN e a publicação deles nos dois bancos de dados do InfluxDB e na EPAGRI**

Os dados decodificados na TTN devem ser recuperados por um servidor que esteja na NUVEM para evitar o tráfego desnecessário no link GPRS. Assim, os serviços e timer que estão sendo executados na Raspberry Pi devem ser movidos para um servidor na nuvem. Para os estudos iniciais foram criados dois servidores (containers) na nuvem. Um sem acesso ao systemd, e portanto deve ser programado com a crontab. O acesso é feito através das portas 711, usando o usuário e a senha padrão do projeto.

---

# Visualização dos dados usando Grafana

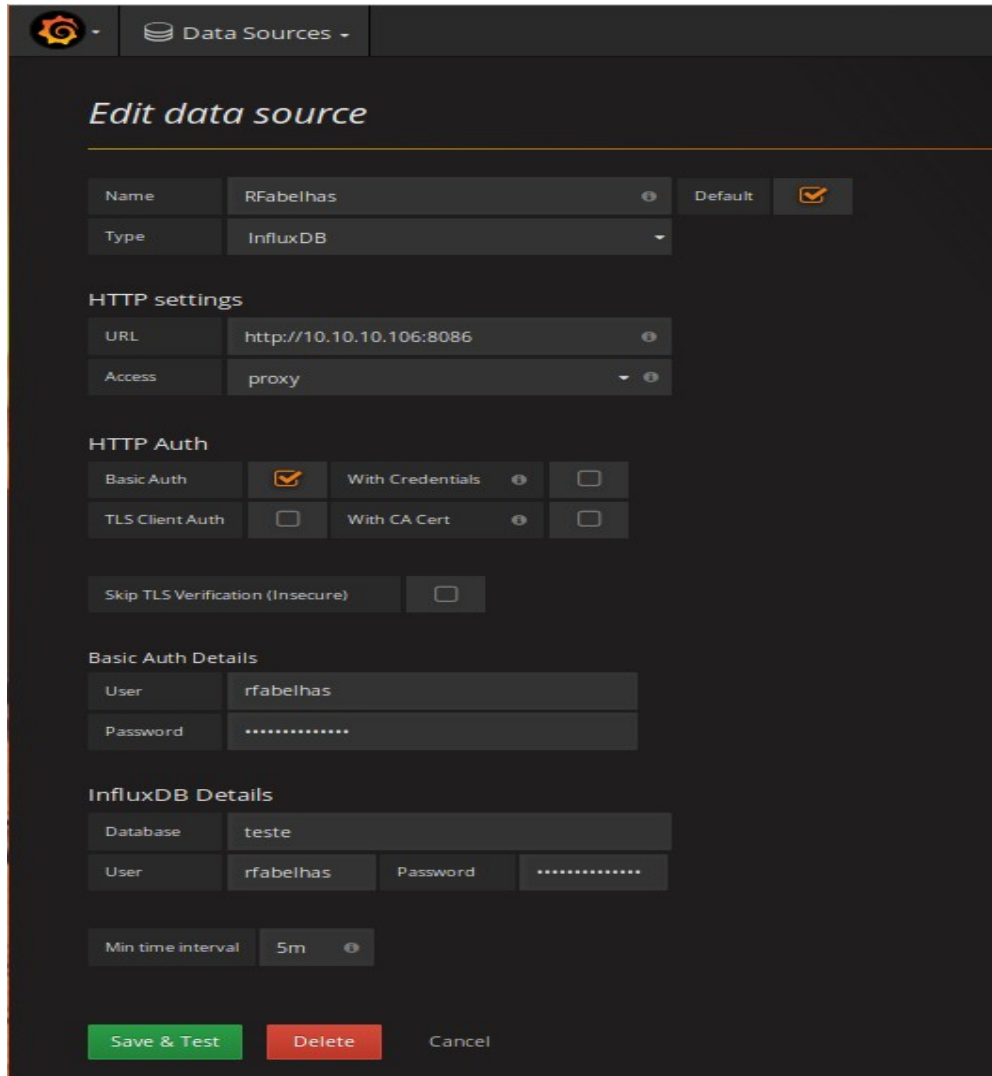
## Instalações necessárias

[Store and visualize data using Node-RED, InfluxDB and Grafana](#)

A instalação e uso do Node-RED foi substituída por um simples comando de acesso ao banco de dados da TTN. Talvez haja problema de segurança como senha podendo ser capturada. Na situação atual do projeto, vamos manter assim e investigar isso numa próxima etapa.

[Install InfluxDB and create a database](#)

[Install Grafana](#)



The screenshot shows the 'Edit data source' interface in Grafana. The data source is named 'RFabelhas' and is of type 'InfluxDB'. It is set as the default data source. The configuration includes the following details:

- Name:** RFabelhas
- Type:** InfluxDB
- Default:**
- HTTP settings:**
  - URL:** http://10.10.10.106:8086
  - Access:** proxy
- HTTP Auth:**
  - Basic Auth:**  **With Credentials:**
  - TLS Client Auth:**  **With CA Cert:**
  - Skip TLS Verification (Insecure):**
- Basic Auth Details:**
  - User:** rfabelhas
  - Password:** [Redacted]
- InfluxDB Details:**
  - Database:** teste
  - User:** rfabelhas
  - Password:** [Redacted]
- Min time interval:** 5m

At the bottom, there are three buttons: 'Save & Test' (green), 'Delete' (red), and 'Cancel' (grey).

Configuração do Dashboard



**Graph** | General | **Metrics** | Axes | Legend | Display | Alert | Time range

Data Source: rlabelfhas

```
FROM arlogon rlabelfhas WHERE column
```

SELECT field (speed) insert

GROUP BY time (5m) fill (null)

FORMAT AS Time series

ALIAS BY

[Add Query](#)