**Installation and User Instructions**

**Bore Pump Automation**

**ThingsBoard/The Things Network/LoRaWAN/Feather**

September 2020  
Version 0.1.0

# Document Revisions

|  |  |  |
| --- | --- | --- |
| Date | Version Number | Document Changes |
| 17 Sept 2020 | 0.1 | Initial draft |
| 9 Oct 2020 | 0.2 | Updated troubleshooting section. |
|  |  |  |
|  |  |  |







# Table of Contents

1 PREFACE 5

1.1 Description of the User/s 5

1.2 Obtaining Documentation and Information 6

1.2.1 Internet 6

1.2.2 Documentation Feedback 6

2 Description of the product 7

2.1 Purpose of the Product 7

2.2 Process Overview 7

2.3 Understanding the User Interface 8

2.4 Operating Panels 9

2.4.1 Manual Pump Control Panel 9

2.4.2 Automatic Pump Control 10

2.4.3 Alarm Panel 11

2.4.4 Depth Indicator 12

2.4.5 Pump Controller Event Log 12

2.4.6 Pump Controller Alarm Log 13

3 End User System Operation 14

3.1 Manual Operation 14

3.1.1 Turn Pump On and Off Manually 14

3.1.2 Turn Pump On Manually With Estimated Fill Timer 15

3.1.3 Turn Pump On Manually With Custom Timer 16

3.2 Automatic Mode 17

3.2.1 Turn On Pump Automation 18

3.2.2 Notes on Automatic Operation 18

3.3 Alarms 19

4 The Pump Controller 20

4.1 Firmware Configuration 20

4.1.1 config.h 20

4.1.2 arduino\_secrets.h 21

4.2 I/O Pins 22

4.3 Operation 23

5 Dashboard and Device Configuration 24

5.1 Pump Controller Devices 24

5.1.1 Details 24

5.1.2 Shared Attributes 24

5.1.3 Server Attributes 24

5.2 Tank Level Sensor Devices 25

5.2.1 Details 25

5.2.2 Server Attributes 25

5.2.3 Relations 25

5.3 The Dashboard 26

6 Pump Simulator Software 28

7 Troubleshooting 30

7.1 Device Alias 30

7.2 Shared Attributes 30

7.3 ‘Manages’ Relationship 30

7.4 Things Network Configuration 30

# PREFACE

## Description of the User/s

This manual is aimed at two types of user. The first is the end user, farmers on the land utilizing the bore pump control system remotely, and technical users at DPI configuring and administering the system.

The ‘Farmer’ persona is not necessarily a technically minded person. They are looking for a simple way to use and automate their bore pump operation.

The ‘DPI’ user is more technical, able to configure remote systems and understands concepts such as the network stack and can differentiate (or even write code) between different programming languages.

No certification is required to operate this system, however an experienced operator should be tasked with its configuration.

## Obtaining Documentation and Information

### Internet

The latest version of the documentation is available at the following address: <https://github.com/DPIclimate/LoRaWAN-Arduino-Thingsboard-Bore-Pump-Controller/tree/master/docs>

Supplemental documentation for system components can be found:

#### ThingsBoard

ThingsBoard documentation is located at <https://thingsboard.io/docs/>

#### The Things Network

Documentation regarding The Things Network is located here: <https://www.thethingsnetwork.org/docs/>

#### Arduino Boards

Documentation for the Adafruit Feather M0 board is located: https://cdn-learn.adafruit.com/downloads/pdf/adafruit-feather-m0-basic-proto.pdf

#### LoRaWAN

LoRaWAN Documentation is located: <https://lora-alliance.org/lorawan-for-developers>

#### Simulator

A .Net-based pump controller simulator has been written for testing and training purposes. Instructions for its use is contained in this document.

### Documentation Feedback

All feedback on this documentation is welcome. Please contact the I.T. Crowd via email with any comments, questions or concerns.

# Description of the product

## Purpose of the Product

The Bore Pump Control System is designed around providing a simple interface for allowing the manual and automatic control of bore pumps on farms. The systems aims to manage the operations of bore pumps remotely via several coordinated systems and components:

***ThingsBoard*** – **ThingsBoard** is an IoT platform already in use with the project sponsor. **ThingsBoard** is the main hub for managing operations. Please see section 2.3 – Understanding the User Interface, for more details.

***LoRaWAN*** – **LoRaWAN** is the method in use at DPI for communicating with devices outside the boundary of more well-known methods such as cellular or Wi-Fi.

***Pump Controller***: The **Pump Controller** is built using an Adafruit Feather, the preferred hardware of DPI for embedded projects. The Pump Controller is a computer attached to the pump control electronics that communicates with the dashboard. It sends a pump status message to the dashboard every 10 minutes, receives commands from the dashboard, and is responsible for driving the pump control electronics in response to the commands.

Individual pins control input and output for various Pump features. Please see section 4.2 - I/O Pins for further information.

***The Things Network***: **The Things Network** allows communication via LoRaWAN.

## Process Overview

The high-level process for managing the pump is simple. Using a dashboard, the farmer end user can operate their pump remotely, using either manual commands (as simple as pump on or pump off), or setting the pump into automatic mode, which utilizes a rules engine to maintain the water level in the bore, keeping it above and below certain level thresholds which can be configured in the dashboard. The automatic mode is driven by tank level readings supplied by a sensor on the tank. *Technical Data can be found in related documentation: Architecture.docx*







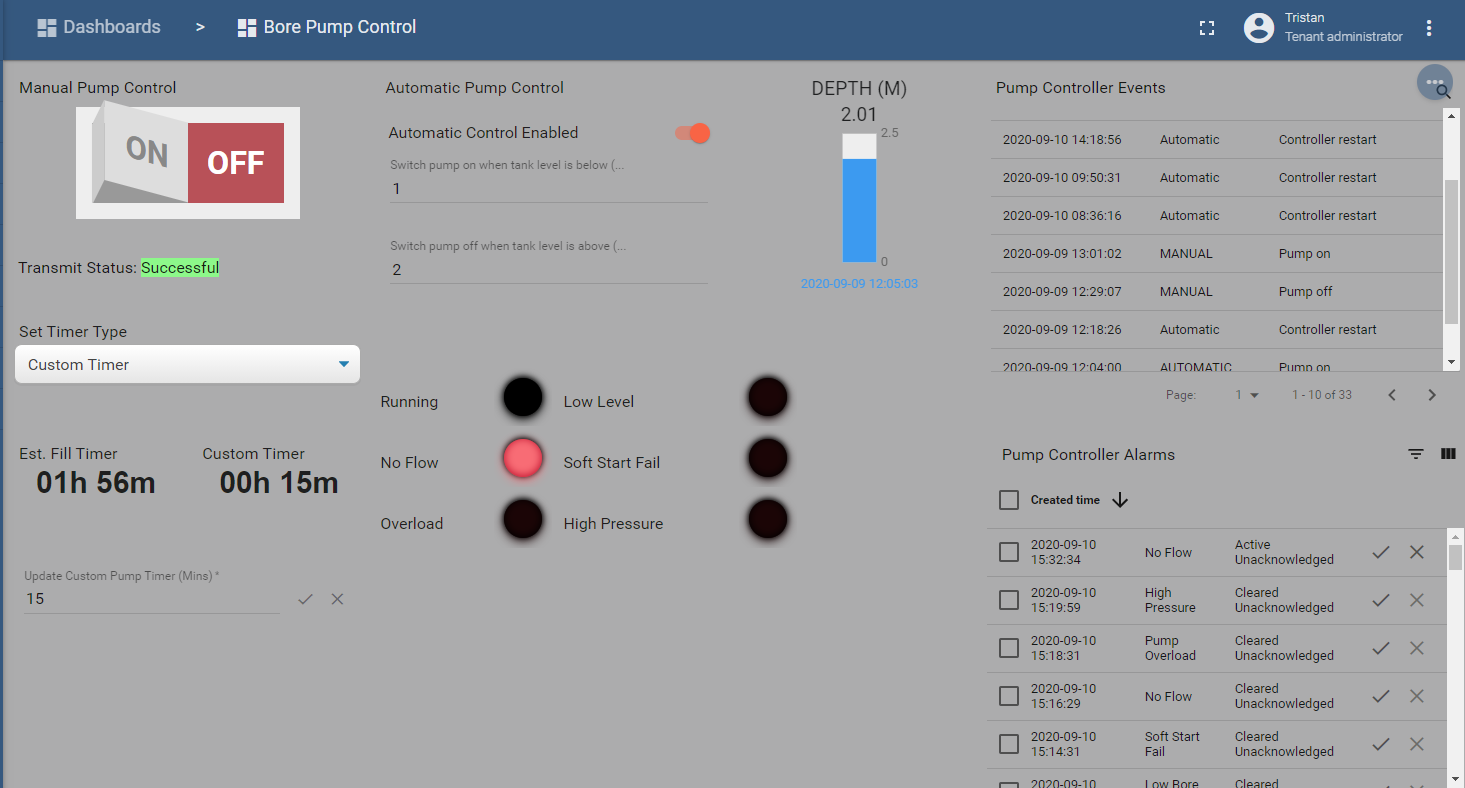




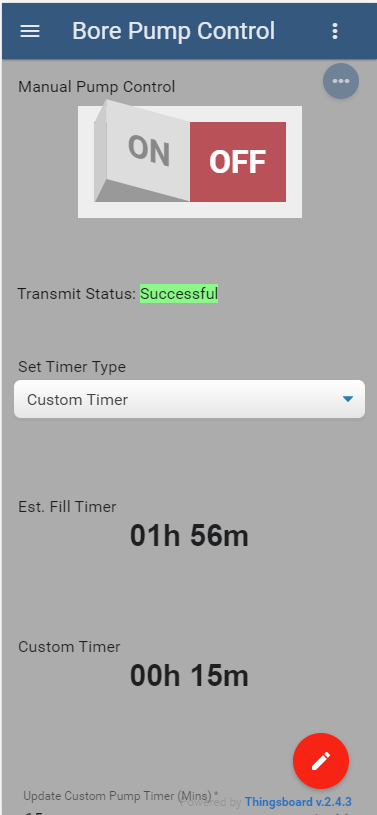
## Understanding the User Interface

The main dashboard interface is designed to be responsive and usable on any device.

Desktop:



Or mobile:



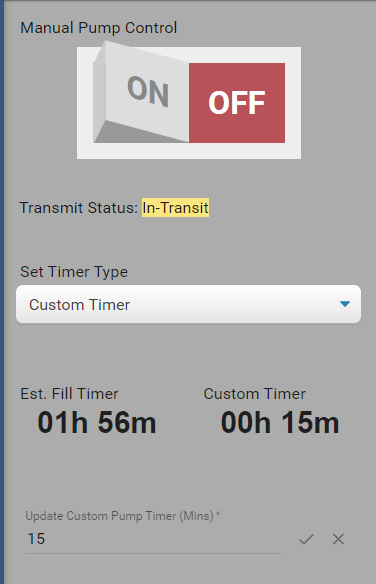
All functions on the dashboard are available in either orientation. Privileged users can change the ordering of Dashboard Widgets (see ThingsBoard documentation for more information).

The main components are the Manual Pump Control switch, the Automated Pump Control toggle, and their respective configuration options. (Custom timer, Estimated Fill Timer, etc.) All UI operations will be detailed later in this document.

Other sections include Alarms, a current depth gauge, and a log of events.

## Operating Panels

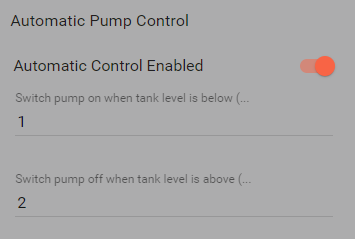
### Manual Pump Control Panel



The manual pump control panel includes the following controls:

* The Manual Pump Control Switch
  + Turns manual control on or off.
* The Transmit Status
  + Transmit status shows the current state of the most recent command to the pump controller.
* Set Timer Type dropdown
  + Allows user to select a different type of timer, options are:
    - Custom – Run pump for a discrete period of time.
    - Estimated Fill Timer – Run pump until pump is full (calculated based on pump flow rate and current depth of tank).
    - No timer – Pump will run until switched off or alarm raised
* Update Custom Pump Timer
  + Allows user to set the time in minutes for the custom pump timer.

### Automatic Pump Control



The Automatic Pump Control is quite simple. It includes the following controls:

* The Automatic Pump Enabled Toggle
  + Turns automatic control on or off.
* Low Level Threshold
  + Depth level at which the logic engine will switch the pump on in order to fill the tank.
* High Level Threshold
  + Depth level at which the logic engine will switch the pump off.

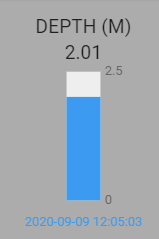
### Alarm Panel



The Alarm Panel shows the status of the following metrics within the system:

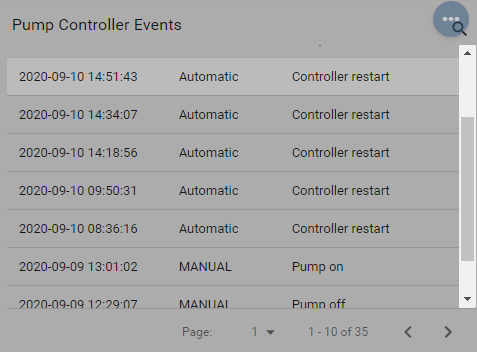
* Running
  + Indicates whether or not the pump is running.
* No Flow
  + Indicates the pump is not flowing.
* Overload
  + Indicates the pump is in an overloaded state.
* Low Level
  + Indicates the bore is at a low level.
* Soft Start Fail
  + Indicates a problem starting the pump.
* High Pressure
  + Indicates the pump is at too high a pressure.

### Depth Indicator



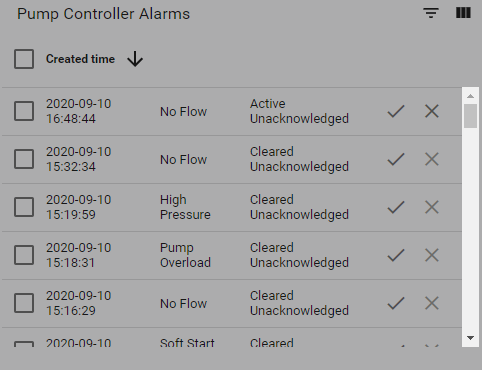
Shows the most recent depth report from the pump controller, including date and time it was recorded.

### Pump Controller Event Log



Shows log of Pump Controller Events and overall system status at the time (Automatic or Manual control).

### Pump Controller Alarm Log



Shows log of alarms and whether they have been acknowledged/cleared, etc.

# End User System Operation

This section is aimed at End Users operating their pumps via a mobile device or PC. It covers operations able to be carried out after configuration and testing has been done. Configuration of end-to-end systems (Firmware, dashboards, ThingsBoard, etc.) is covered in sections 4 and 5.

## Manual Operation

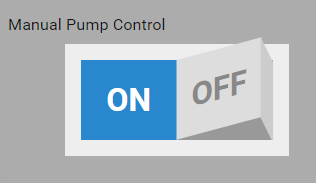
Manual pump operation is simply turning the pump on and off via the Manual Pump Control switch. Manual operation can also be set to run for a specific length of time, either a custom length set by the user, or by enabling the ‘Estimated Fill Timer’ option in the Timer Type dropdown (see below for instructions). The use of timers allows a level of automation, whereby the pump will switch off upon the timer reaching zero.

Due to the limitations of sending messages to the pump controller using the LoRaWAN protocol, pump commands are not received by the pump controller until it sends the next status message. This means commands may not be acted upon until up to 10 minutes after they have been issued from the dashboard.

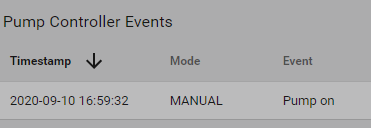
If a pump command is sent before a previous command has been received by the pump controller, the new command *overwrites* the previous one. For example, if you sent a pump ON command with no timer value and realized you wanted to use the estimated fill timer, you could toggle the manual control switch to OFF, change the timer selection, then toggle the manual control switch back to ON. Only the ON command with estimated fill timer would be received by the pump controller.

### Turn Pump On and Off Manually

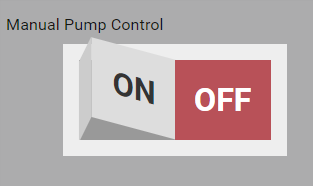
1. Ensure Pump is not in operation (Switched OFF).
2. Toggle Manual Pump Control Switch to ON



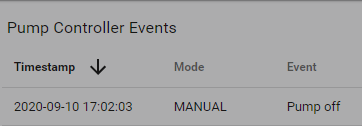
1. Observe Pump On/Manual Mode event in Pump Controller Events



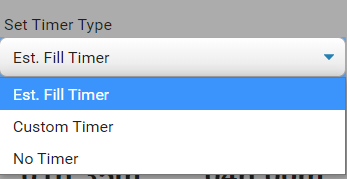
1. When complete, toggle Manual Pump Control Switch to OFF

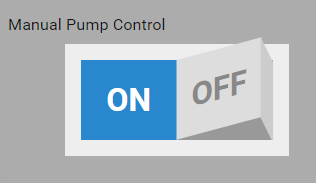


1. Observe Pump Off/Manual Mode event in Pump Controller Events



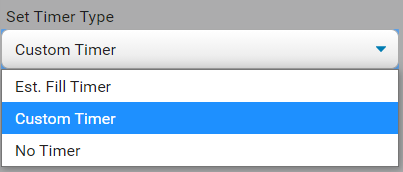
### Turn Pump On Manually With Estimated Fill Timer

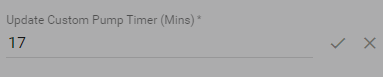
1. Ensure pump is not in operation (Switched OFF).
2. Select Est. Fill Timer from Timer Type Dropdown menu: 
3. The estimated length will be displayed in the dashboard:  
   
4. Toggle Manual Pump Control Switch to ON

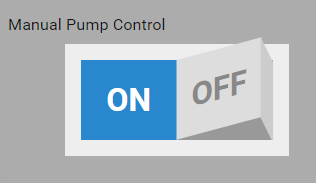


1. Observe Pump on/Manual Mode event in Pump Controller Event Log.
2. At conclusion of Estimated Fill Timer length, the pump will stop running, and the Manual Pump Control will switch to OFF, and be reflected in the Pump Controller Event Log.

### Turn Pump On Manually With Custom Timer

1. Ensure pump is not in operation (Switched OFF).
2. Select Custom Timer from the Timer Type dropdown menu: 
3. Enter a custom time (in minutes) in the Update Custom Timer (Mins) text field and click the tick button:



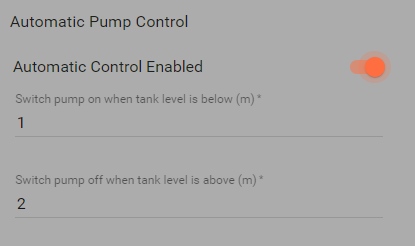
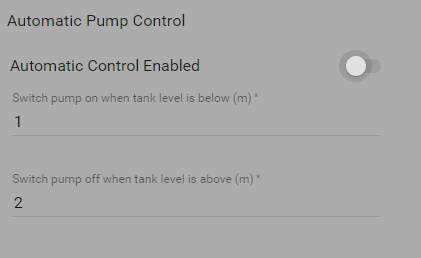
1. Toggle Manual Pump Control Switch to On: 
2. Observe Pump on/Manual Mode event in Pump Controller Event Log.
3. At the conclusion of the Custom Timer length, the pump will stop running, and the Manual Pump Control will switch to OFF, and be reflected in the Pump Controller Event Log.

## Automatic Mode

Automatic mode is designed to switch on automated control of the bore pump in response to water level sensor readings, and other inputs. This mode automates all pump operations by monitoring depth of the tank and sending control messages in response to various inputs, with configurable fields updated by the user as required.

Due to the limitations of sending messages to the pump controller using the LoRaWAN protocol, pump commands are not received by the pump controller until it sends the next status message. This means commands may not be acted upon until up to 10 minutes after they have been issued by the automatic control business logic.

### Turn On Pump Automation

1. Toggle the Automatic Control Enabled switch to ON: 
2. The system is now in Automatic mode. The lower and upper thresholds can be updated while the pump is in Automatic mode and changes will be picked up by the system.
3. To switch off Automatic mode, toggle the Automatic Control Enabled switch to OFF:   
   

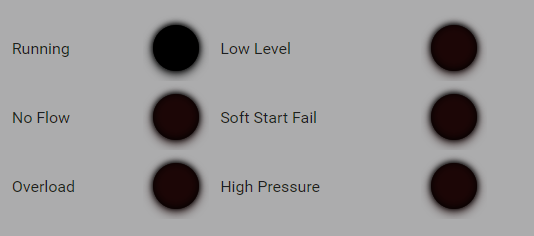
### Notes on Automatic Operation

The logic that is implemented in Automatic mode is as follows:

1. Tank level message comes in to system.
2. ThingsBoard logic checks if pump is in Automatic Mode
3. If in Automatic Mode, the logic checks if the tank level is above or below the thresholds.
4. If below the lower threshold, the pump turns on (assuming pump is not already in this state).
5. If above the upper threshold, the pump turns off (assuming pump is not already in this state).

## Alarms

Alarms are a ThingsBoard method of conveying status, warnings, or errors. It is more accurate to consider them indicators of status or state, as shown in the alarm panel:



During the course of operation, some issues may arise with the pump or tank, causing an alarm to be raised. These are no cause for concern and will often be resolved of their own volition as signals are received and processed by the Pump Controller hardware.

In the case that alarms are continually raised, manual inspection and/or maintenance of the pump, pump controller, or level sensor may be required.

ThingsBoard documentation for Alarms can be found here: <https://thingsboard.io/docs/user-guide/alarms/>

# The Pump Controller

## Firmware Configuration

The firmware must be configured to use the LoRaWAN protocol and to have the correct ThingsNetwork identifiers. The firmware configuration must be done in the source code. The appropriate files are detailed below:

### config.h

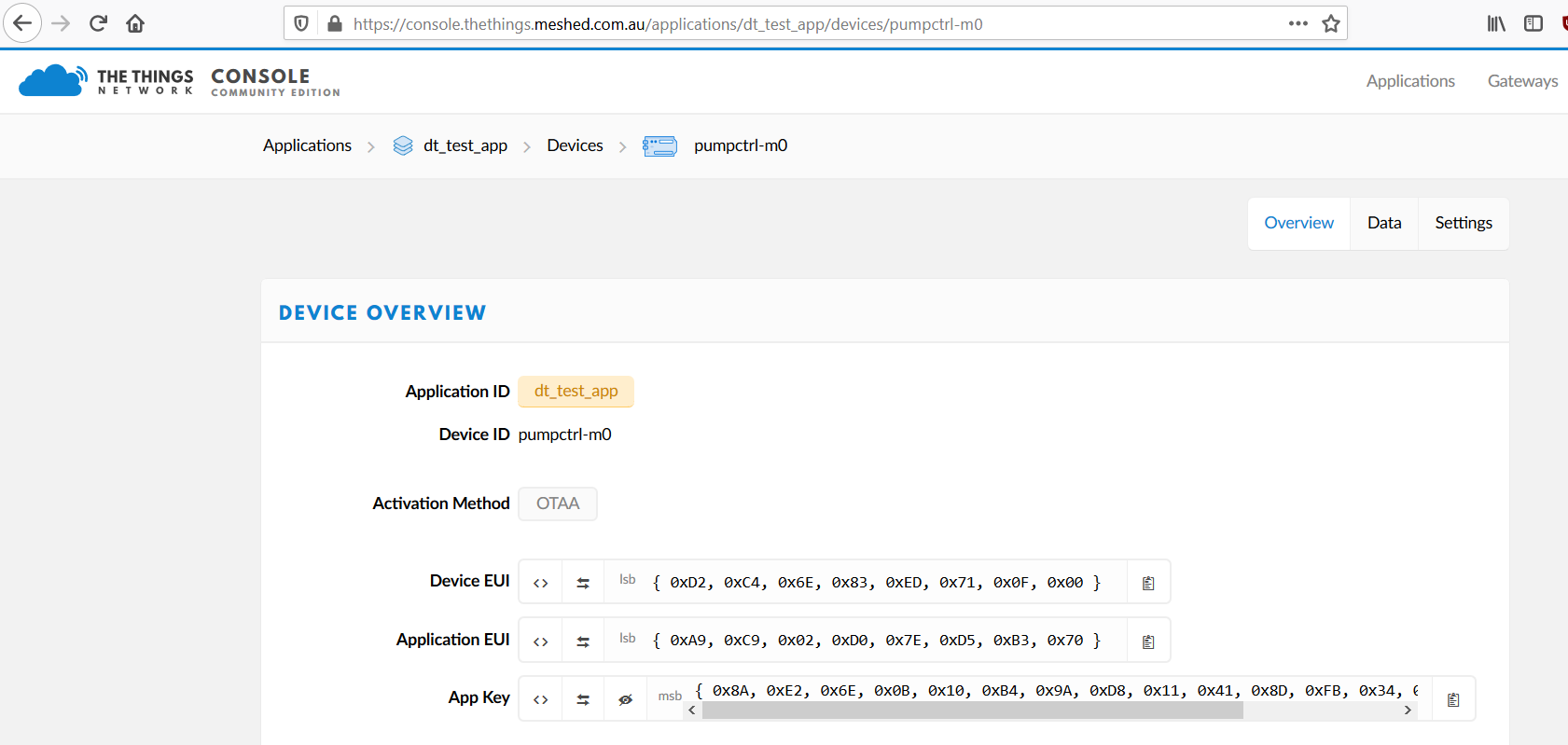
The macro to choose between the WiFi and LoRaWAN protocols is defined in config.h. It must be set to USE\_LORAWAN, and the USE\_WIFI macro must not be defined.

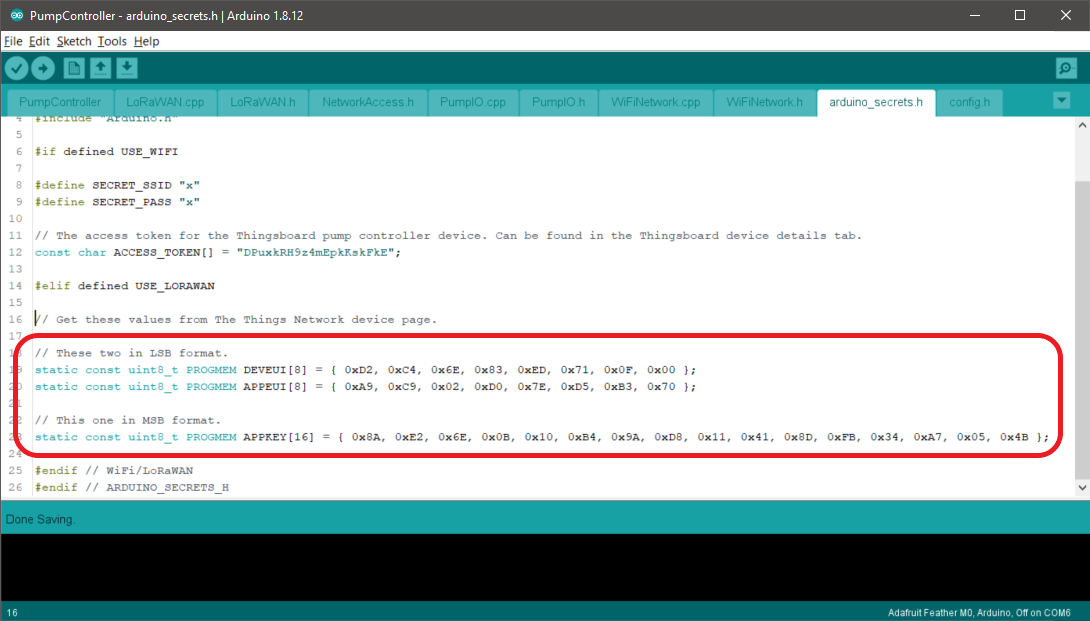
For the breadboard tests, set PUMP\_CONTROL\_PIN = 13 to use the build-in LED. This will light the LED when the pump control signal is high – i.e. the pump should be running and put the LED out when the pump is not running.



### arduino\_secrets.h

The Things Network application and device keys must be set in arduino\_secrets.h. Note the byte ordering requirements and note on the ThingsNetwork screenshot how to get them in the correct order.





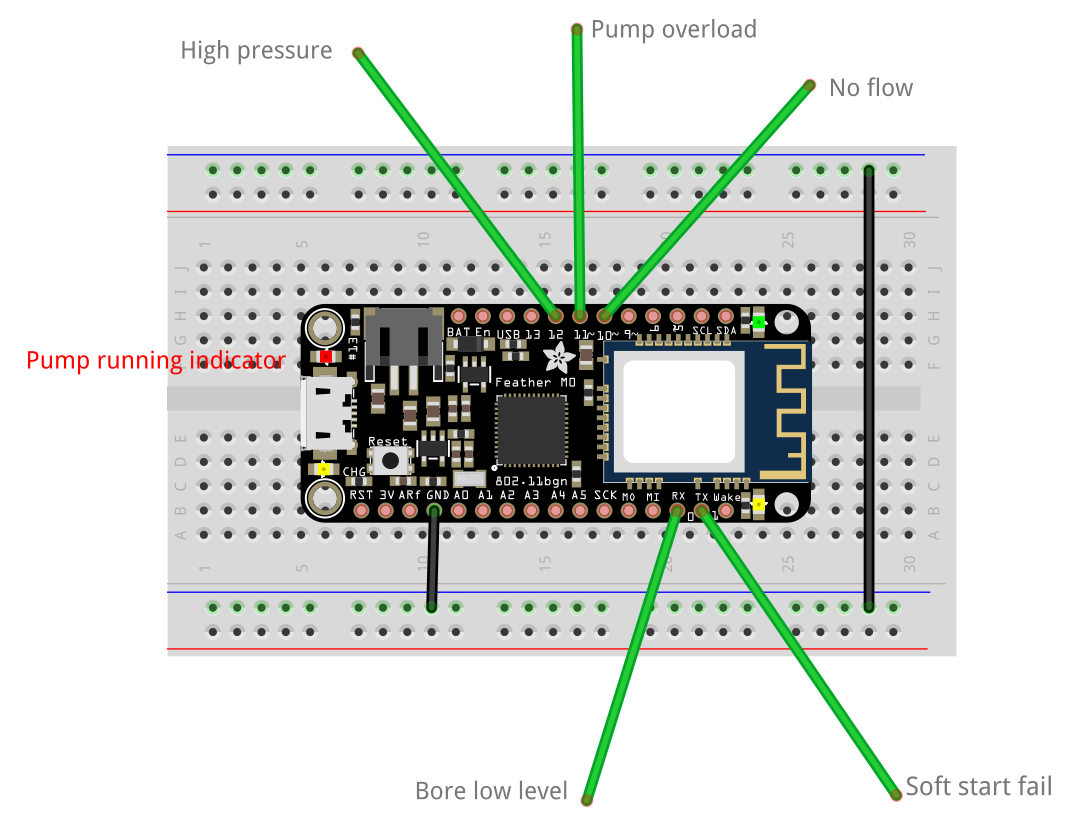
The firmware must be built and uploaded to the WiFi Feather after these changes have been made.

## I/O Pins

The following diagram shows the pump controller I/O pin assignments as delivered. These can be modified if required by editing config.h as explained in Section 4.1.1.

Note the pump control output pin is assigned to the on-board LED by default. A high level will be put onto this pin if the pump is meant to be running, and the pin will be low if the pump is not meant to be running.

The pins marked with green wires are input pins and are expected to be driven low if the appropriate alarm condition is in effect. The internal pull-up resistors are enabled for these pins so no external wiring is required for a signal if it is not in use.



## Operation

The pump controller will boot as soon as power is applied via the USB port or battery connector.

The pump controller will sleep for 10 seconds and then attempt to join The Things Network using OTAA. This means joining time is variable but should be under a minute. The initial sleep time is to allow time to connect with a serial console before the first log messages are written if desired.

Once the pump controller has joined The Things Network it will send an initial status message and then send another status message every 10 minutes, as long as it is connected to The Things Network.

Commands from the dashboard will only be received after the next status message has been sent by the pump controller. This means commands may take up to 10 minutes to take effect.

A change on the pump control output pin or any of the input pins causes a status message to be sent as soon as possible.

Commands to run the pump will be ignored if any of the input lines are low when the command is received.

The pump control output pin will be brought low (that is, switch the pump off) if any of the input lines go low while the pump is running.

If connection to the Things Network is lost the controller will attempt to join again when it sends the next status message.

Below is an example of the pump controller boot sequence.

|  |
| --- |
| 15:35:06.220 -> 0.1.lora  15:35:06.220 -> setup() complete.  15:35:06.220 -> EV\_JOINING  15:36:07.756 -> EV\_JOIN\_FAILED  15:37:16.165 -> EV\_JOINED  15:37:16.165 -> Send initial status message.  15:37:16.165 -> LoRaWAN will encode and send this message: {'pumpRunning':0,'boreLowLevel':0,'softStartFail':0,'pumpOverload':0,'controllerRestart':1,'highPressure':0,'noFlow':0}  15:37:16.165 -> Sending status byte: 10  15:37:18.527 -> EV\_TXCOMPLETE (includes waiting for RX windows) |

# Dashboard and Device Configuration

This section explains the configuration required for the tank level sensor and pump controller devices, and how to attach them to the dashboard.

**NOTICE:** These steps should only be carried out by someone with experience in ThingsBoard Configuration. End Users (i.e.: Farmers) should consult with DPI for setting up Dashboards and Devices.

## Pump Controller Devices

A pump controller device is the ThingsBoard representation of the Feather controlling the bore pump.

The following tables describe the configuration required for a pump controller. If an attribute is listed here it must be given a value before the device can be used on the dashboard.

### Details

|  |  |  |
| --- | --- | --- |
| Field | Value | Notes |
| Name | <Any value> | You can use any name for the pump controller device. |
| Device Type | “Pump Controller” | This field must have the value “Pump Controller”. |

### Shared Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Value | Notes |
| protocol | String | “lora” or “wifi” | The value is used to select the mechanism used to send commands to the pump controller. |

### Server Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Value | Notes |
| inactivityTimeout | Integer | 1,500,000 | 25 minutes in milliseconds.  This gives 2.5x the expected status message interval before the pump controller is flagged as offline and an alarm is raised. |
| lowLevel | Double | 1 | The tank level in metres below which automatic control will switch the bore pump on. This does not have to be 1, but must be less than highLevel. |
| highLevel | Double | 2 | The tank level in metres above which automatic control will switch the bore pump off. This does not have to be 2, but must be greater than lowLevel. |

## Tank Level Sensor Devices

A tank level sensor device is the ThingsBoard representation of the water level sensor in a water tank. Readings from this sensor are used for automatically controlling the bore pump.

The following tables describe the configuration required for a tank level sensor. If an attribute is listed here it must be given a value before the device can be used on the dashboard.

### Details

|  |  |  |
| --- | --- | --- |
| Field | Value | Notes |
| Name | <Any value> | You can use any name for the tank level sensor device. |
| Device Type | “Water/Fuel Depth Level” | This field must have the value “Water/Fuel Depth Level”. |

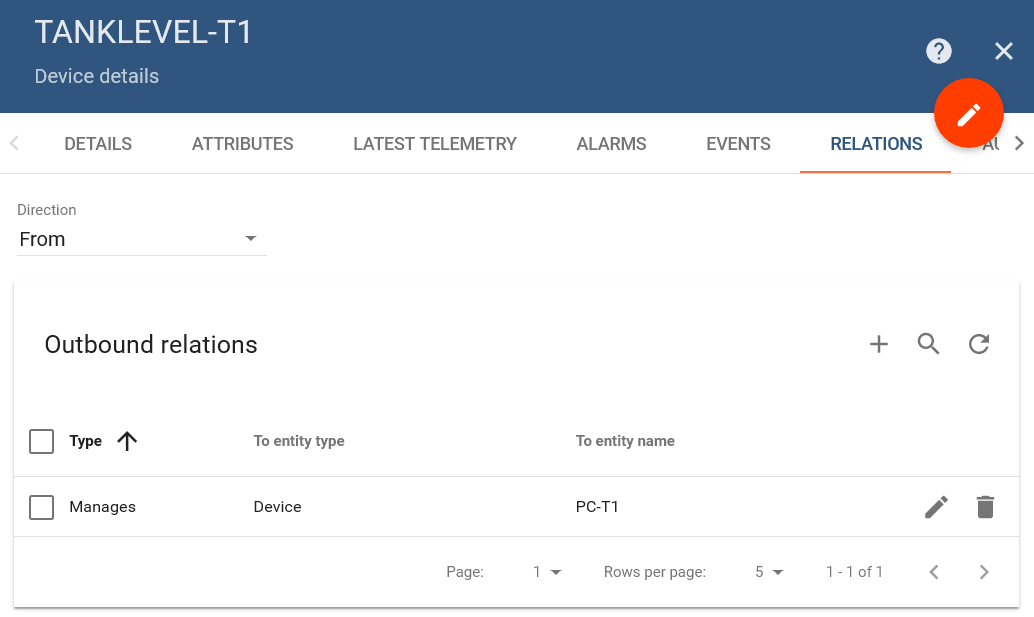
### Server Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Value | Notes |
| tankFillRate | Double | 0.0042 | This value is in metres/min and represents how quickly the bore pump can raise the tank water level.  It is used to calculate the estimate fill time. An approximate value is sufficient. |

### Relations

A relationship between the tank level sensor and the pump controller must be defined. The relationship is created in the tank level sensor device and is of type “Manages” to a pump controller device.

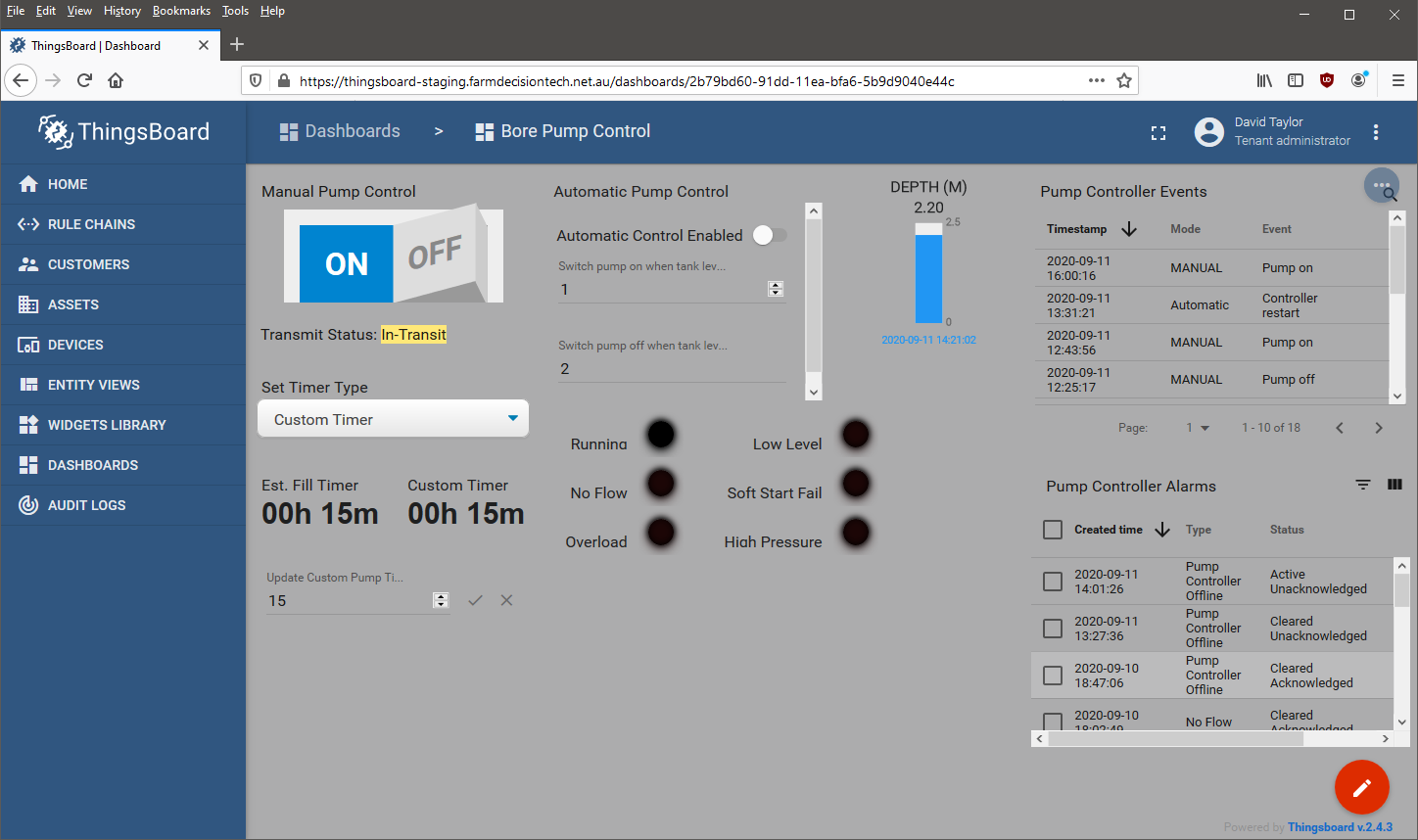
Each tank level sensor must only manage one pump controller, and each pump controller must only be managed by one tank level sensor. So, each tank level sensor must have exactly one “Manages” relationship and two tank level sensors cannot manage the same pump controller.



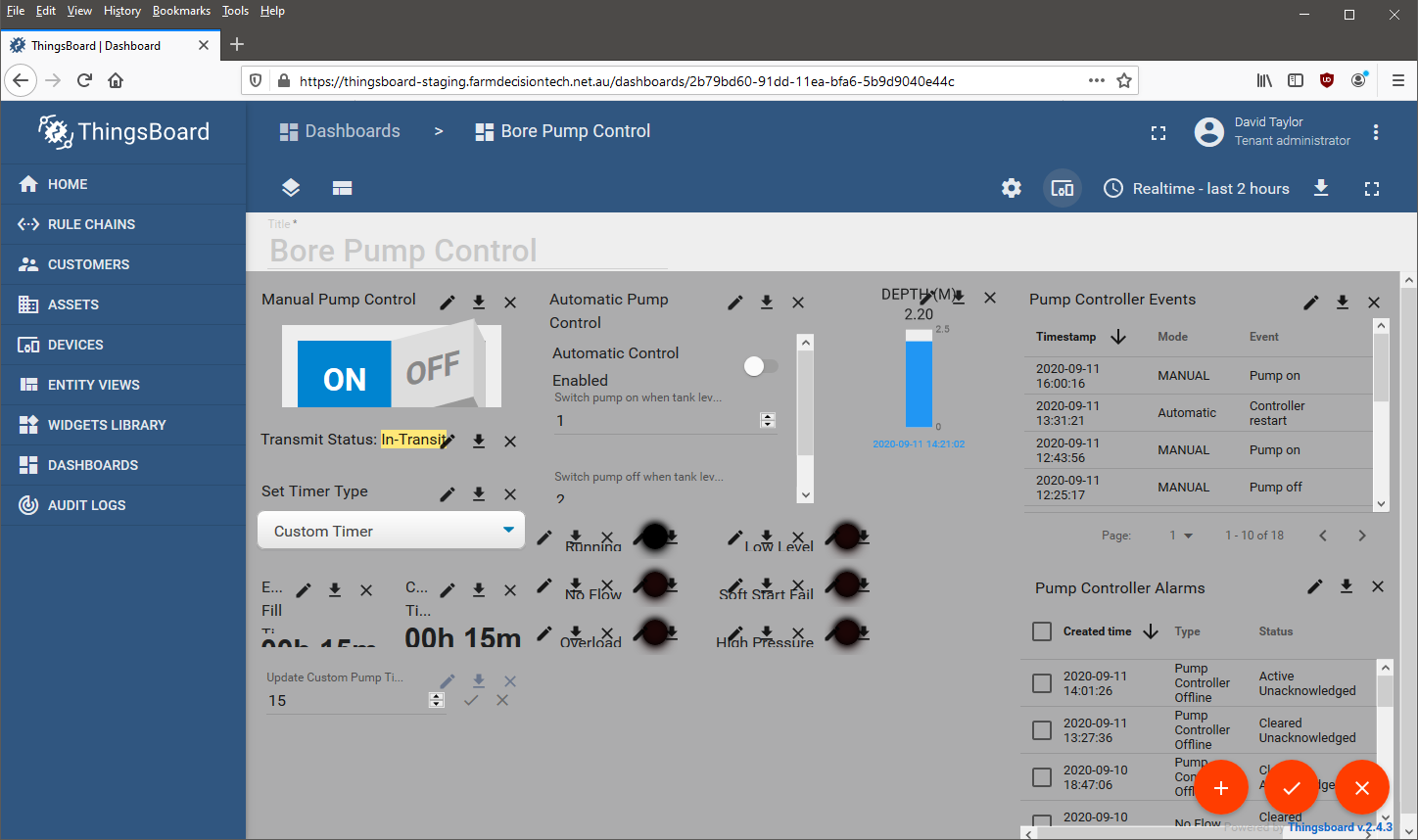
## The Dashboard

With the devices configured they can be associated with the dashboard using the dashboard aliases. This is standard ThingsBoard functionality.

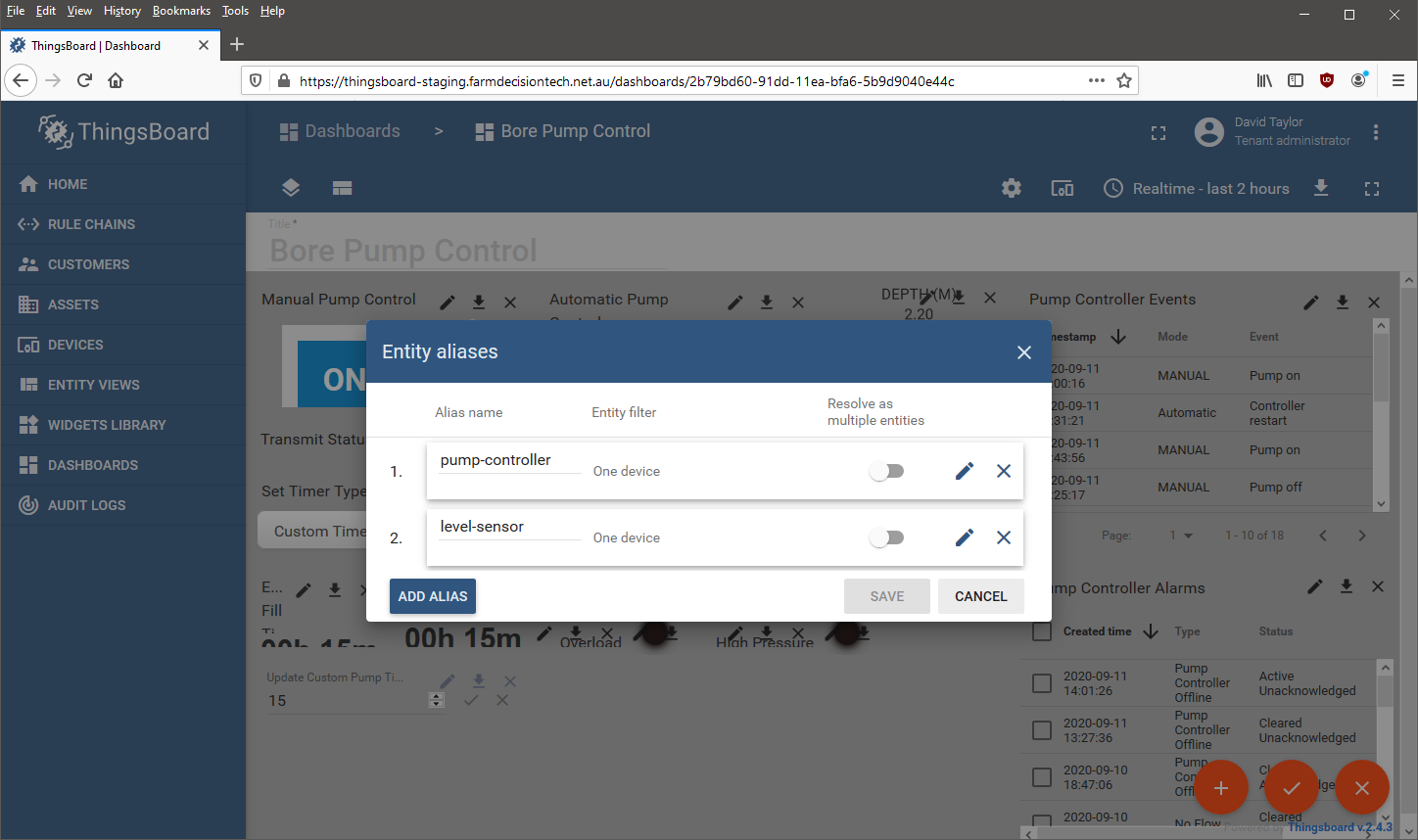
Place the dashboard into edit mode by clicking on the pencil icon in the lower right corner.



Open the device aliases dialog by clicking the aliases button in the dashboard toolbar.



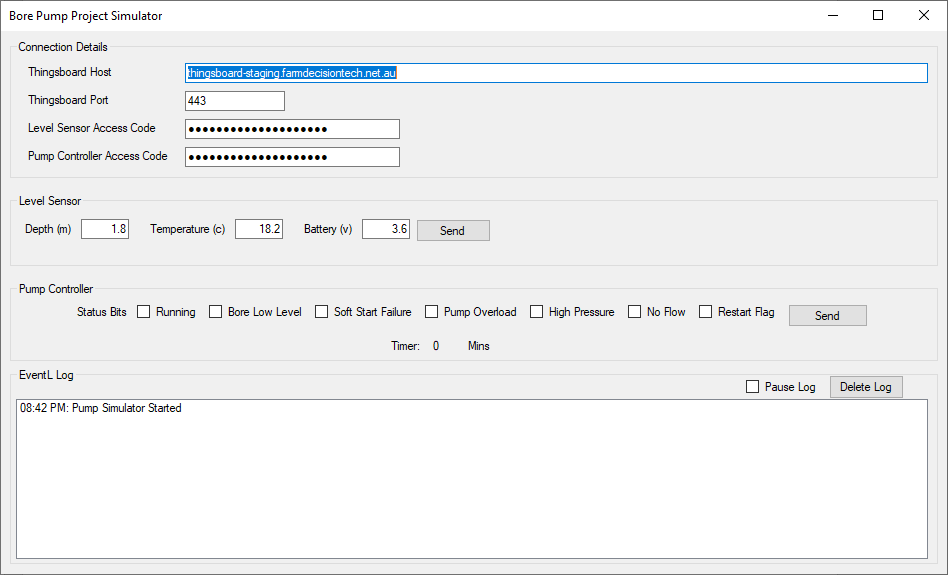
Create or modify two aliases. The “pump-controller” alias must reference the pump controller device and the “level-sensor” alias must reference the tank level sensor device. The alias names must be as written here. The aliases must be for “one device” and the resolve as multiple entities feature must be disabled.



# Pump Simulator Software

To aid in development, testing and training, the IT Crowd have developed a Tank Level Sensor and Pump Controller simulator to emulate the Tank Level Sensor and Pump Controller in circumstances where a full end-to-end environment is not practical or feasible. The source for this solution can be found in the Bitbucket repository.

To build and run the simulator:

1. Open the solution in Visual Studio and select Build Solution from the Build menu.
2. Run the solution: 
3. Ensure the URL and port number are correct. The Level Sensor Access and Pump Controller Access codes can be copied to the clipboard from the Thingsboard device details page and pasted into the fields here. Changes to these fields are saved automatically.  
     
   By default, the devices controlled by the Pump Simulator are:
   1. PUMPCTRL-WIFI (Pump Controller)
   2. DUMMY LEVEL SENSOR (Level Sensor)

These devices are configured via the instructions in Section 5 of this manual, and can be updated as required by an experienced DPI operator.

As the Pump Simulator emulates existing DPI hardware, its operations are not in the scope of this document, however further instructions are available by contacting the I.T. Crowd if required.



































# Troubleshooting

Occasionally, during the operation of the system, some issues will occur. This section aims to outline some of the more common problems, and some steps to remediate. If an issue occurs that isn’t covered, or cannot be resolved, please contact DPI or the I.T. Crowd.

## Dashboard not showing latest telemetry

If the dashboard is not updating with the latest telemetry data, the dashboard device aliases may be referencing the wrong devices. See section 5.3 for information on configuring the dashboard device aliases.

## Pump commands are in the event list but not Things Network console

If pump control commands are being shown in the event list on the dashboard but are not being seen in the Things Network console it could be the rules engine is trying to send them to a wifi Pump Controller. During set up of the Pump Controller device, it is essential to ensure the ‘protocol’ shared attribute is accurate, i.e.: the right protocol for the device being set up, either Wifi or LoRaWAN. Please see section 5.1.2 for relevant information.

## Automatic pump control not working

If automatic pump control is not working it could be due to a misconfigured device relationship. The rules engine must be able to find the Pump Controller associated with a Tank Level Sensor. A bad device relationship configuration means the associated device might not be found or the pump control messages are sent to the wrong device.

See section 5.2.3 in this document for details on setting this relationship. Ensure there is a 1:1 relationship between Tank Level Sensor and Pump Controller, no Pump Controller should be managed by more than one Tank Level Sensor, and no Tank Level Sensor should manage more than one Pump Controller.

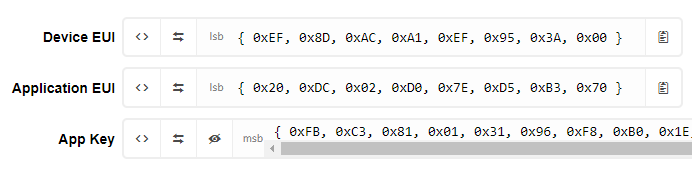
## Pump controller cannot join Things Network

If the Pump Controller cannot join the Things Network it could be due to bad line of sight between the Pump Controller and the LoRaWAN gateway or incorrect values in the firmware that are used to identify the Pump Controller to the Things Network.

When configuring devices in The Things Network console and Pump Controller firmware (specifically the file arduino\_secrets.h), it is easy to use the wrong values, or the wrong formatting of the values. Ensure that:

1. The values are in the correct format, i.e.:
   1. Firmware Variable ‘DEVEUI’ matches Things Network field ‘Device EUI,’ and that the field in Things Network is in LSB format.
   2. Firmware Variable ‘APPEUI’ matches Things Network field ‘Application EUI,’ and that the field in Things Network is in LSB format.
   3. Firmware Variable ‘APPKEY’ matches Things Network field ‘App Key’ and that the field in Things Network is in MSB format.

Correct formats are shown here:



Also see section 4.1.

## Transmit Status says Failed

If a pump on or off command has been sent to the Pump Controller and the expected status is not reflected in the messages coming back from the Pump Controller after 10 minutes the Transmit Status will change to Failed. This may happen because the downlinks have not been received properly or there is an alarm condition active that is stopping the Pump Controller from starting the pump as requested.

If no alarm conditions are active the situation may resolve itself if the downlinks were not getting through and one finally makes it onto the Pump Controller.

If an alarm condition is active and the command was to switch the pump on, then change the dashboard switch to off to bring the dashboard state back into sync with the Pump Controller. This will change the status to yellow in-transit until the next status message arrives from the pump controller.

In any case, changing the dashboard switch will send another command and give the dashboard and Pump Controller another get in sync. Changing the switch twice is ok too, for example from on to off, and back to on. This would send a pump off message, then overwrite that with a pump on message – only the latest downlink command is delivered to the Pump Controller.