Pigeon IMU User’s Guide

Rev 1.0

Cross The Road Electronics

www.ctr-electronics.com

Table of Contents

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your CTRE products. To this end, we will continue to improve our publications, examples, and support to better suit your needs.

If you have any questions or comments regarding this document, or any CTRE product, please contact support@crosstheroadelectronics.com

To obtain the most recent version of this document, please visit www.ctr-electronics.com.

1. What is the Pigeon IMU?

Pigeon IMU is an inertial measurement unit that can sense acceleration, angular velocity, and Earth's magnetic field. With this information, Pigeon can be used to sense a mobile platform's pose, which then can be used for a variety of applications.

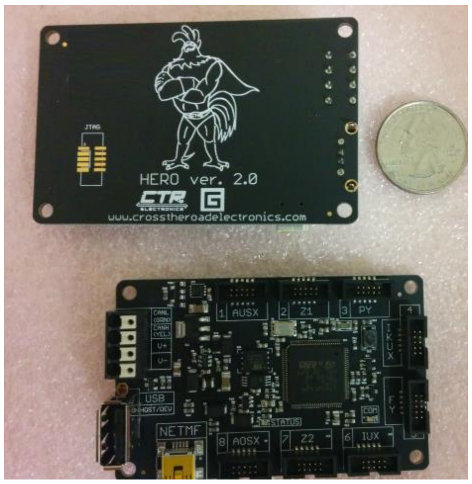
Pigeon IMU features include…

- 9 Degrees of freedom (3 Axis accelerator, 3 Axis Gyroscope, 3 Axis Magnetometer).
- Full AHRS (Attitude and heading reference system): Yaw, Pitch, and Roll.
- Gyro automatically re-biased after eight seconds of no-motion.
- Optional Temperature Compensation for further reduced gyro drift and accelerometer error.
- Boot Time: ~ 5 seconds
- Optional Compass fusing for zero heading drift.
- Data connection can be CAN bus, or Gadgeteer-data-cabled to Talon SRX (UART).
- Connection strategy allows for multiple IMUs (multiple arm/manipulator servos, drivetrain heading).
- Invensense MPU-9250 MotionTracking™ Device
- Robot heading and Yaw are continuous, ideal for robot heading servos, and motion plotting. No need to modulo divide anything.
- Talon SRX Motor controller and Pigeon have a direct line of communication (allows for advanced future motion profiled features).
- Conformal-coated
- Robot-level API implementation is simple, no exceptions and few error conditions to service.
- Reverse Battery Protection

2. Supported Hardware Platforms

The Pigeon IMU is design to integrate across multiple platforms. Listed below are the common control systems the Pigeon is natively supported.

2.1. Cross The Road Electronics HERO Control System



The CTR HERO Control System board allows developers to utilize all features of the Pigeon IMU. It is meant for education, custom development, and integration of Pigeon IMU into existing applications.
The HERO also provides a method for field upgrading Pigeon to latest firmware. It is the ideal development kit for learning and integrating the Pigeon IMU into custom applications!

Applications are developed in Visual Studio 2019 (C#, VB, managed-C++) using .NETMF framework.

Be sure to look for the for HERO related tips.

2.2. roboRIO FRC Control System

The only legal robot controller for FRC competition. The roboRIO supports CAN bus and Phoenix Tuner provides a GUI for re-flashing and diagnostics.

Be sure to look for the for FRC related tips.

2.3. Custom UART Client

The Pigeon can be used as a simple UART device. Although implementing the entire serial protocol is complex, a simple example can be found at the CTR-Electronics GitHub account for reading basic signals. The example uses a HERO and Gadgeteer UART port, however the software can be ported to any platform that uses 3.3V UART.

3. Specifications

3.1. General Specifications

Outside Dimensions	1.5" x 1.5" x 0.25"
Weight	~6g (0.2 oz.) (excluding wiring)
Supported Communication Protocols	DWCAN bus (1Mbps), UART
Nominal Battery Voltage	12V
Min/Max Voltage	5.5 – 28V
Max Current	50mA
Boot Calibration	~5 seconds (1)
Yaw (6D) angle drift (motion)	~1 degree per minute (2)
Yaw (6D) angle drift (no motion)	0.23 -0.28 degrees per minute (2)
Pitch, Yaw, Roll (6D)	100 Hz (10ms)
Gyroscope resolution	±2000 degrees per second (16 bit)
Accelerometer resolution	±2 g (16 bit)
Magnetometer (Compass) resolution	±4800 µT (14bit)
Compass Accuracy	5 degrees (3) 1-2 degrees (3,4)
Compass Update Rate	100 Hz (10ms)
Fused Heading drift	1 - 2 degrees (2,3,4)

NOTE 1: Platform must be still for this period. Otherwise boot calibration will require more time.
2: Assumes temperature to be reasonably constant. Though not required, drift may be further improved after Temperature Calibration Procedure. The improvement will range from one individual Pigeon to another.
3: Requires Level 1 Compass Calibration and proper compass placement.
4: Requires Level 2 Compass Calibration and proper compass placement.

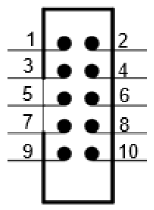
3.2. Electrical Specifications

Symbol	Parameter	Condition	Min	Typ.	Max	Unit
Pigeon power though V+ pads						
V+	Supply voltage		5.5(2)	12.0	28.0	V
I_{supp}	Supply Current		13	26	50	mA
I_{supp}	Supply Current	DC supply 28V DC supply 12.0V DC supply 6V		13 26 46		mA
Pigeon powered through Gadgeteer data cable						
$V_{TalonSupp}$	Supply Voltage from Talon			5.0		V
$I_{TalonSupp}$	Supply Current from Talon			50		mA

NOTE 1: All testing done with orange LEDs on Pigeon (for maximum current draw) unless otherwise specified.
2: CAN will stop functioning at this threshold, however the Pigeon micro controller and IMU will continue to sense motion. CAN communication will be restored once V+ reaches/exceeds 5.7V.

3.3. Gadgeteer Pinout

Pinout	
1	Not Connected
2	5V
3	Not Connected
4	Data RX
5	Data TX
6	RESET
7	Not Used
8	Not Connected
9	Index Output (*future features)
10	Ground



Data Port Pinout

4. Power and Data Connectivity

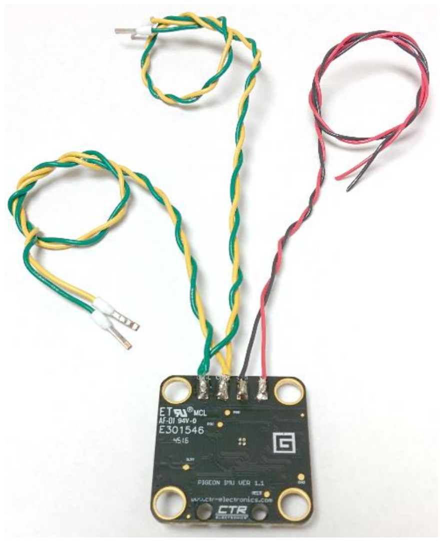
There are multiple ways to power/connect the Pigeon IMU to a robot control system. The two primary methods are…

- Soldering CAN Bus and V^+/Ground (12V battery) to Pigeon PCB.
- Connecting Gadgeteer data cable between Pigeon and the device port of a Talon SRX (on CAN bus). Talon SRX will power Pigeon, and gateway communication between CAN Bus and Pigeon.

Only **one** of these methods should be used at any given time.

The details of each connection strategy are explained below.

4.1. Connecting to CAN and ~12V power



Pigeon can be powered directly using a ~12V power source and wired directly to CAN Bus. When used in this manner, wires can be soldered directly to the solder pads on the underside of the Pigeon.

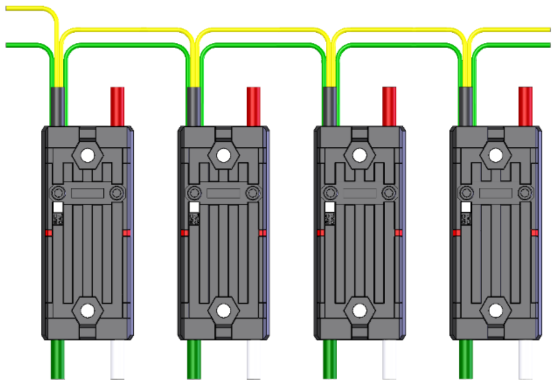
TIP: Using **green for CANL** and **yellow for CANH** is common practice in the HERO and FRC-roboRIO control system.

FRC: The roboRIO and PDP typically should be placed at the two ends of the CAN bus. This ensures proper CAN bus termination.

HERO: HERO has internal CAN bus termination. When used with HERO, only one 120Ω termination resistor needs to be placed at the far end (away from the HERO).

When Pigeon is first powered up, Pigeon may blink red or blink orange depending on initial setup.

4.1.2. Daisy chaining CAN devices



CAN bus daisy-chaining is common in the HERO and the FRC-roboRIO control system. The right diagram demonstrates how multiple Talon SRX devices can be connected in a daisy chain fashion.

This can also be done with Pigeon by soldering two sets of green/yellow wire pairs to the CAN pads (see picture above).

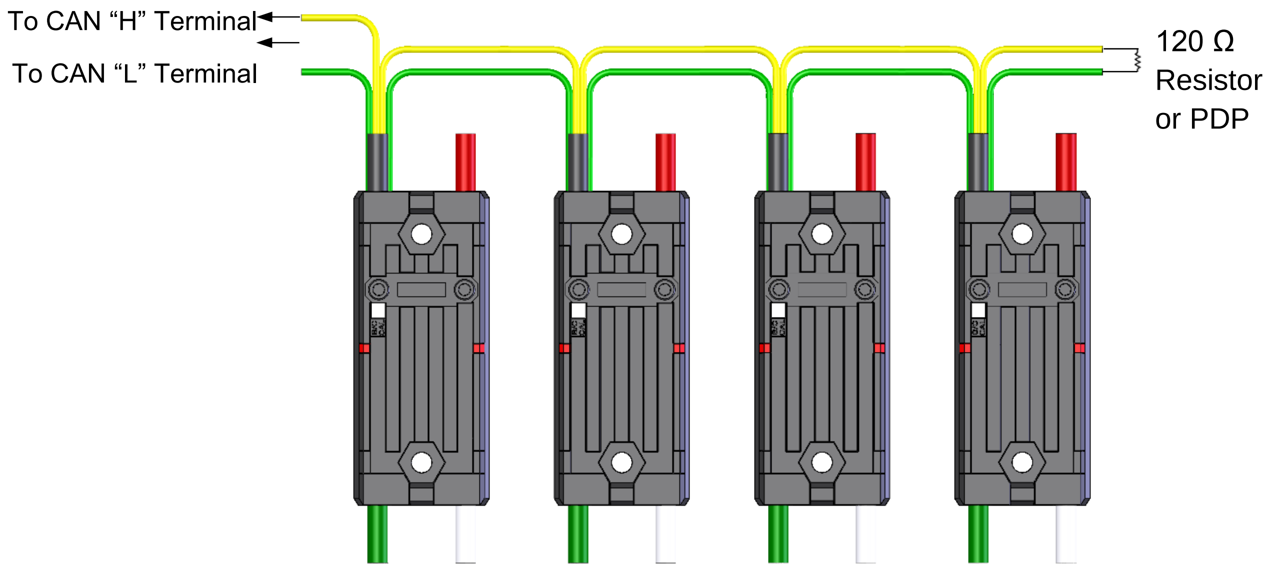
The recommended method is to first strip two yellow wires, twist the stripped portion together, then tin with solder. Next solder the twisted tinned wires to the Pigeon pad. Repeat for green pair of wires. Finally twist the wires of each green/yellow pair (strongly recommended for CAN bus).

4.1.2.1. Wiring Pigeons/Talon SRXs for use with CAN bus

To wire Pigeon and/or Talon SRXs, connect a **yellow** signal wire to the CAN terminal marked "**CANH**" on the robot-controller and connect a **green** signal wire to the CAN terminal marked "**CANL**" on the robot-controller.

To connect additional Pigeons/Talon SRXs, use tightly crimped connectors to connect the signal wires **green**-to-**green** & **yellow**-to-**yellow** as shown below. For the best connection, it is recommended that each connector is crimped **and** soldered. However, CAN connectors are available to provide reliable connections that are not permanent (see Section 4.1.1.2).

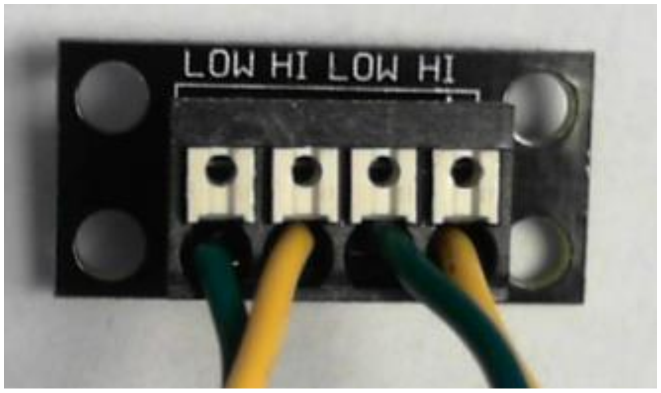
The **yellow** and **green** wires should also be wrapped in a twisted pair fashion (not illustrated) to maximize tolerance to electrical noise.
Note: Signal wires of the same color are electrically identical – it does not matter which wire is used as long as the color is correct.



After all Pigeons/Talon SRXs have been wired, there will be 2 remaining signal wires – connect these two wires using a 120 Ω resistor or to the CAN interface on the Power Distribution Panel (PDP) to properly terminate the cable end.

4.1.2.2. CAN Connector

The CAN Connector can be used to chain Pigeons/Talon SRXs together without crimping connectors or soldering. Each CAN Connector contains a four channel Weidmuller terminal block with two CAN pairs (similar to other FRC CAN devices such as the Power Distribution Panel and Pneumatic Control Module). The four holes can be used to provide strain relief to the CAN wires. Additionally, the holes can be used for mounting to the robot frame. Spacers or electrical tape may be used to prevent shorting to the robot frame.



These are available for purchase at ctr-electronics.com.

4.1.2.3. CAN Bus Wire Selection

It is recommended to use **yellow** for CANH and **green** for CANL for the following reasons…

- Makes inspection and troubleshooting easier.
- The colors match what is labeled on the HERO/roboRIO, Power Distribution Panel, and Pneumatic Control Module.
- The colors match the Talon SRX cable harness.

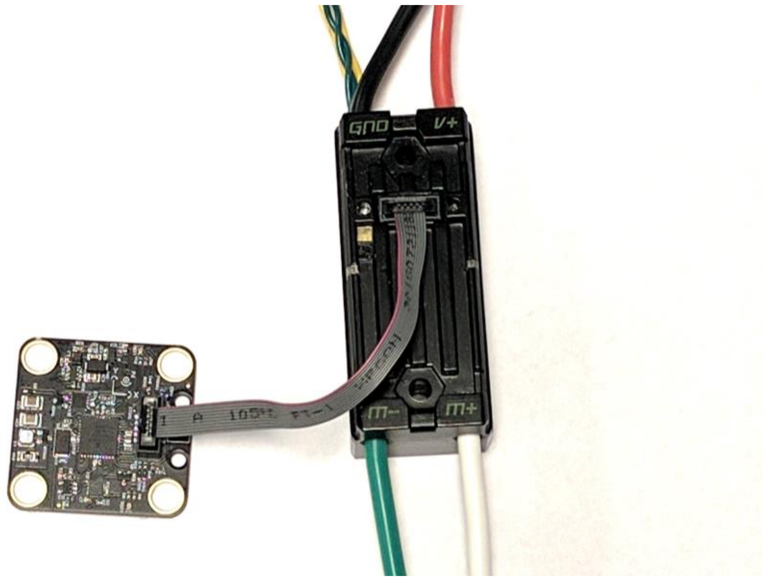
AWG 22 or similar gauge wiring can be used. An electric drill can be used to twist the CANH/CANL wire pair. Pre-twisted wire is also available at ctr-electronics.com.

4.2. Connecting to Talon SRX Gadgeteer data cable.

Alternatively, Pigeon can be powered and data-connected to a Talon SRX. The Talon SRX must…

- Be present on CAN bus. Talons only wired for PWM cannot be used to integrate Pigeon.
- Must not have a sensor plugged into the data port as this is where Pigeon connects.
- Talon firmware must be reasonably up to date. Section 5 includes the procedure for this.

Additional Gadgeteer cables, cable kits, and extenders are available at ctr-electronics.com.



When Pigeon is first powered up, Pigeon may blink red or blink orange on the initial Talon firmware version.

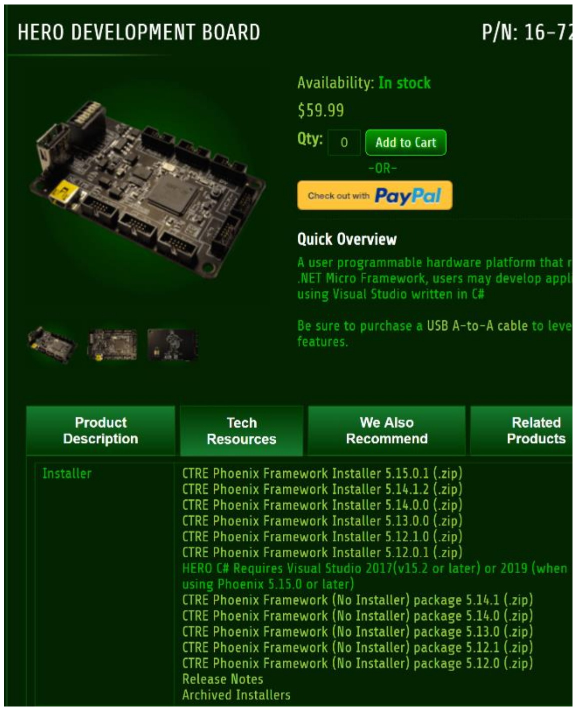
5. Pigeon Walk-though

This section describes a top-to-bottom procedure for a basic installation and confirmation of a Pigeon IMU on the HERO and roboRIO control system.

5.1. Power and data connectivity

The first step is to provide power and data connectivity to the Pigeon. Pigeon can be placed on CAN bus, or connected to a Talon’s device port. Follow either Section 4.1. Connecting to CAN… or Section 4.2. Connecting to Talon SRX… depending on user preference.

5.2. Software/API Installation



Availability: In stock
$59.99
Qty: 0

Quick Overview
A user programmable hardware platform that r
.NET Micro Framework, users may develop appl
using Visual Studio written in C#

Be sure to purchase a USB A-to-A cable to leve
features.

Installer	CTRE Phoenix Framework Installer 5.15.0.1 (.zip) CTRE Phoenix Framework Installer 5.14.1.2 (.zip) CTRE Phoenix Framework Installer 5.14.0.0 (.zip) CTRE Phoenix Framework Installer 5.13.0.0 (.zip) CTRE Phoenix Framework Installer 5.12.1.0 (.zip) CTRE Phoenix Framework Installer 5.12.0.1 (.zip) HERO C# Requires Visual Studio 2017(v15.2 or later) or 2019 (when using Phoenix 5.15.0 or later) CTRE Phoenix Framework (No Installer) package 5.14.1 (.zip) CTRE Phoenix Framework (No Installer) package 5.14.0 (.zip) CTRE Phoenix Framework (No Installer) package 5.13.0 (.zip) CTRE Phoenix Framework (No Installer) package 5.12.1 (.zip) CTRE Phoenix Framework (No Installer) package 5.12.0 (.zip) Release Notes Archived Installers

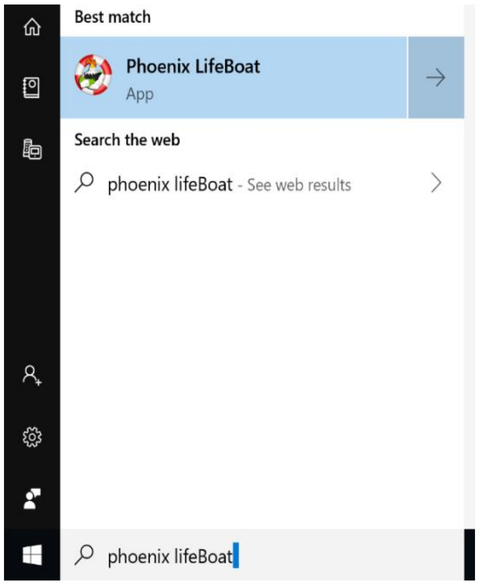
CTR-Electronics distributes support for CAN devices via the HERO Tool Suite Installer.
This includes installation of…

- Robot API for HERO (C#/VB) and roboRIO(LabVIEW/Java/C++)
- Code Examples for devices (including Talon SRX and Pigeon)
- HERO LifeBoat for general device configuration.
- Visual Studio NETMF (if selected and Visual Studio is discovered)
- Copy of latest firmware files for various CTRE devices.

The installer can be found at…
www.ctr-electronics.com/control-system/hro.html

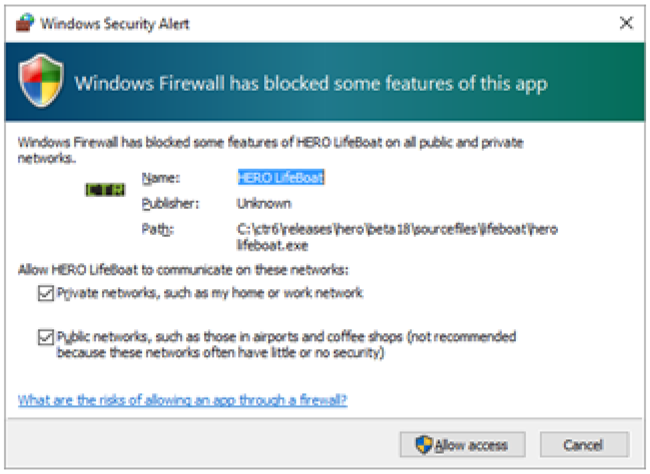
5.2.1. Software/API Installation – Open HERO LifeBoat

Below are some sanity checks the user can perform to check if software is installed. Additionally, these steps are helpful when diagnosing another person's setup.



Phoenix Lifeboat should now appear in the start menu.

When LifeBoat runs the first time, you may get a prompt to allow LifeBoat to communicate over the network**. Check "private networks…" and then "Allow access". If using HERO Control system, you may need to select "public" as well.**



Windows Security Alert

Windows Firewall has blocked some features of this app

Windows Firewall has blocked some features of HERO LifeBoat on all public and private networks.

Name: HERO LifeBoat
Publisher: Unknown
Path: C:\ctr6\releases\hero\beta18\sourcefiles\lifeboat\hero lifeboat.exe

Allow HERO LifeBoat to communicate on these networks:

[✓] Private networks, such as my home or work network

[✓] Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)

What are the risks of allowing an app through a firewall?

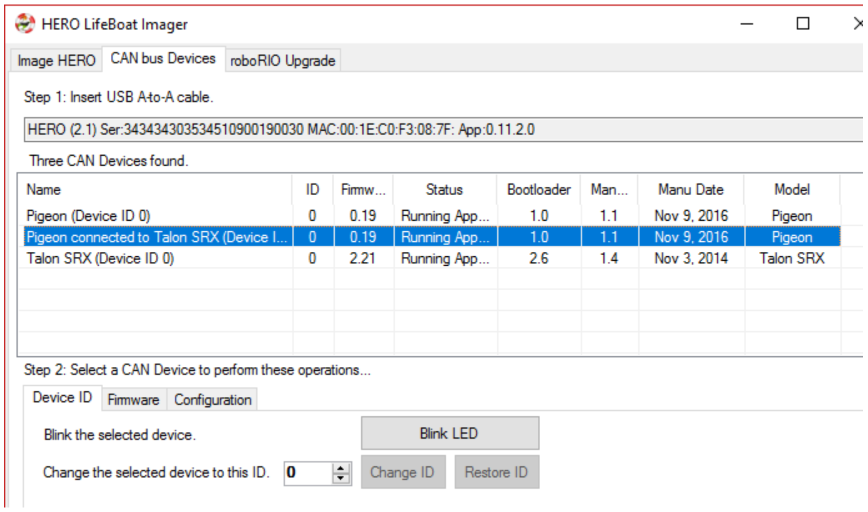
5.3. Device Configuration

After installing the CTRE Tool Suite, the user will be able to open HERO LifeBoat. This step is necessary for both HERO and roboRIO based control systems.

HERO developers should follow Section 5.3.1. Those using the roboRIO should follow Section 5.3.2.

5.3.1. HERO – Device Configuration

This section is applicable when using the HERO Control System. More detailed information can be found in the HERO's User's Guide. All that is required at this point is connecting the HERO to the PC using a USB A-to-A cable, and wiring HERO to the CAN bus shared with Pigeon and/or Talon SRX.



Step 1: Insert USB A-to-A cable.

HERO (2.1) Ser:343434303534510900190030 MAC:00:1E:C0:F3:08:7F: App:0.11.2.0

Three CAN Devices found.

Name	ID	Firmw...	Status	Bootloader	Man...	Manu Date	Model
Pigeon (Device ID 0)	0	0.19	Running App...	1.0	1.1	Nov 9, 2016	Pigeon
Pigeon connected to Talon SRX (Device I...	0	0.19	Running App...	1.0	1.1	Nov 9, 2016	Pigeon
Talon SRX (Device ID 0)	0	2.21	Running App...	2.6	1.4	Nov 3, 2014	Talon SRX

Step 2: Select a CAN Device to perform these operations...

Blink the selected device.

Change the selected device to this ID. 0

HERO LifeBoat will discover the Pigeon regardless of which connection strategy is used.

The top Pigeon is present on CAN bus.

The bottom two lines shows a Pigeon connected to Talon SRX via the gadgeteer cable, and the corresponding Talon.

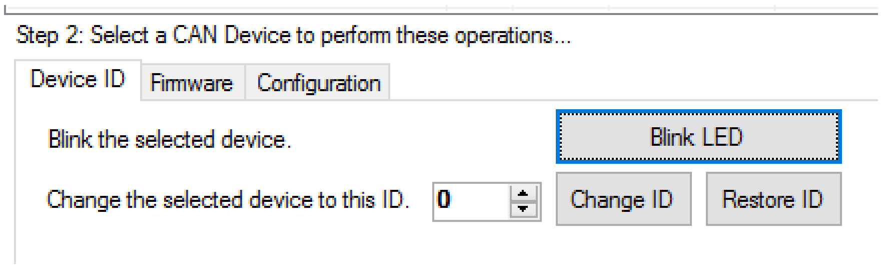
5.3.1.1. HERO – Device Configuration – Talon is present but Pigeon is missing?

If using Gadgeteer cable strategy, ensure the Talon firmware is up to date. If the Talon firmware is old, it will not understand how to automatically detect and communicate with the Pigeon using the Gadgeteer data cable. To ensure Talon can support Pigeons plugged into the data port, check the Talon firmware versions against these milestones…

- **FRC** Talon firmware be equal or newer than **2.20**
- **HERO** Talon firmware be equal or newer than **10.20**

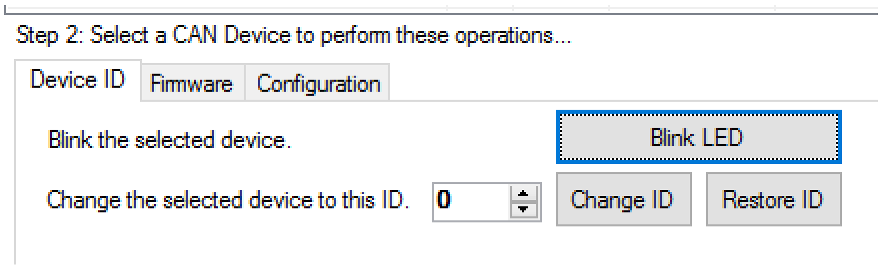
5.3.1.2. HERO – Device Configuration – Blink Test

Confirm the progress made so far by pressing the "Blink LED" button. Pigeon should be fast-blinking both LEDs orange to confirm that it received the command. This is a useful feature to identifying/confirming that the correct line-item corresponds to the correct physical device. This is particularly helpful with systems that have more than Talon SRX/Pigeon/PCM/etc.


Step 2: Select a CAN Device to perform these operations...
Device ID
Firmware
Configuration
Blink the selected device.
Blink LED
Change the selected device to this ID.
0
Change ID
Restore ID

5.3.1.3. HERO – Device Configuration – Setting Device IDs

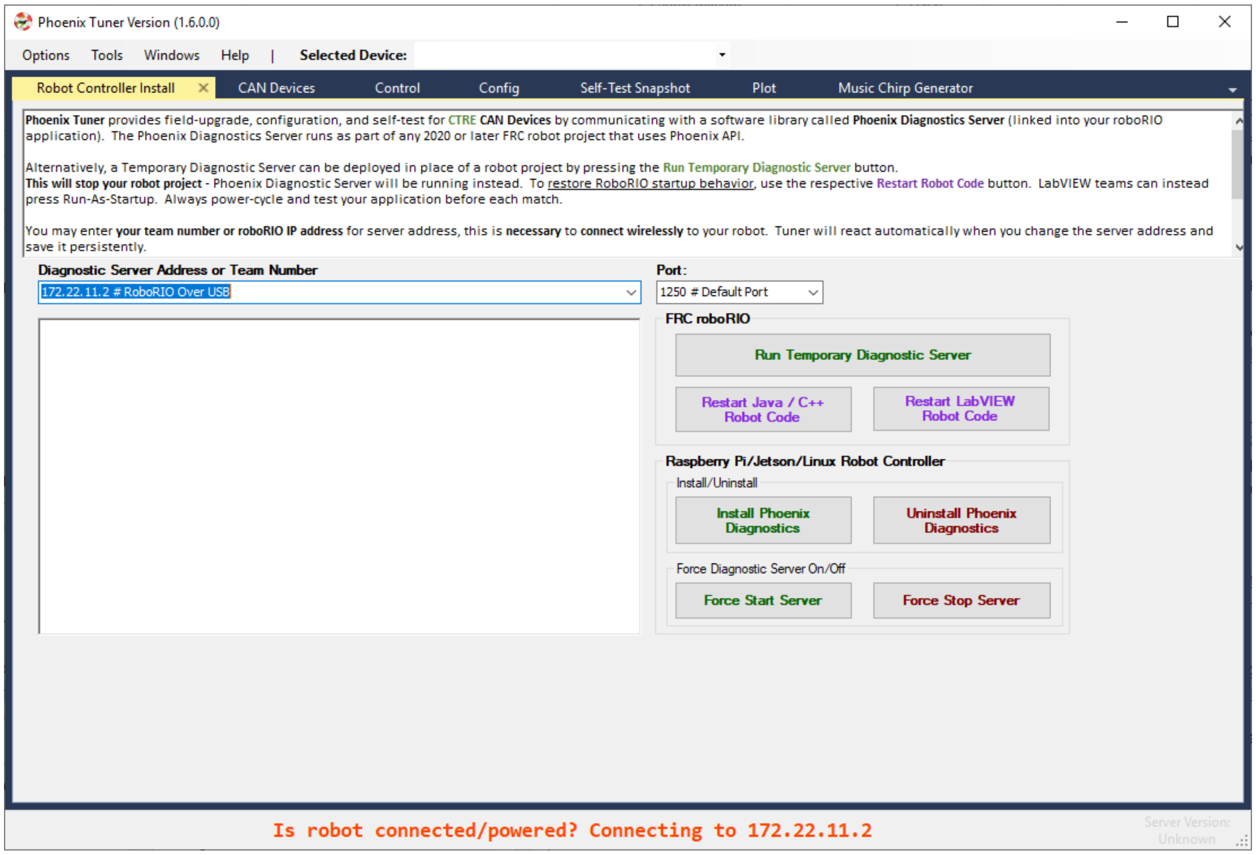
Devices of the same model must have unique CAN device IDs. For example, if there is more than one Pigeon wired to the CAN Bus, then each Pigeon must be given a unique device ID. This can be done with the device ID number entry and the "Change ID" button.


Step 2: Select a CAN Device to perform these operations...
Device ID
Firmware
Configuration
Blink the selected device.
Blink LED
Change the selected device to this ID.
0
Change ID
Restore ID

5.3.2. FRC roboRIO – Device Configuration

Phoenix Tuner can be used to configure the Pigeon IMU with the FRC roboRIO.

See instructions in our software documentation here:
https://phoenix-documentation.readthedocs.io/en/latest/index.html



Phoenix Tuner provides field-upgrade, configuration, and self-test for CTRE **CAN Devices** by communicating with a software library called **Phoenix Diagnostics Server** (linked into your roboRIO application). The Phoenix Diagnostics Server runs as part of any 2020 or later FRC robot project that uses Phoenix API.

Alternatively, a Temporary Diagnostic Server can be deployed in place of a robot project by pressing the Run Temporary Diagnostic Server button.
This will stop your robot project - Phoenix Diagnostic Server will be running instead. To restore RoboRIO startup behavior, use the respective Restart Robot Code button. LabVIEW teams can instead press Run-As-Startup. Always power-cycle and test your application before each match.

You may enter **your team number or roboRIO IP address** for server address, this is **necessary** to **connect wirelessly** to your robot. Tuner will react automatically when you change the server address and save it persistently.

Diagnostic Server Address or Team Number: 172.22.11.2 # RoboRIO Over USB

Port: 1250 # Default Port

FRC roboRIO
Run Temporary Diagnostic Server
Restart Java / C++ Robot Code
Restart LabVIEW Robot Code
Raspberry Pi/Jetson/Linux Robot Controller
Install/Uninstall
Install Phoenix Diagnostics
Uninstall Phoenix Diagnostics
Force Diagnostic Server On/Off
Force Start Server
Force Stop Server

```
Is robot connected/powered? Connecting to 172.22.11.2
```

Server Version: Unknown

5.4. Updating the firmware of a Pigeon

Updating the firmware on Pigeon is similar to updating the firmware on Talon SRX.
The two methods of updating firmware is with the HERO development board (LifeBoat) and the FRC roboRIO (Phoenix Tuner).

5.4.1. Where is the latest firmware?

Firmware files are typically installed in two locations…

- **C:\users\Public\Documents\FRC**
- **C:\users\Public\Documents\Cross The Road Electronics\LifeBoat\HERO Firmware Files**

…and are also available at www.ctr-electronics.com under the various product pages.

5.4.2. What happens if the wrong product's CRF is selected?

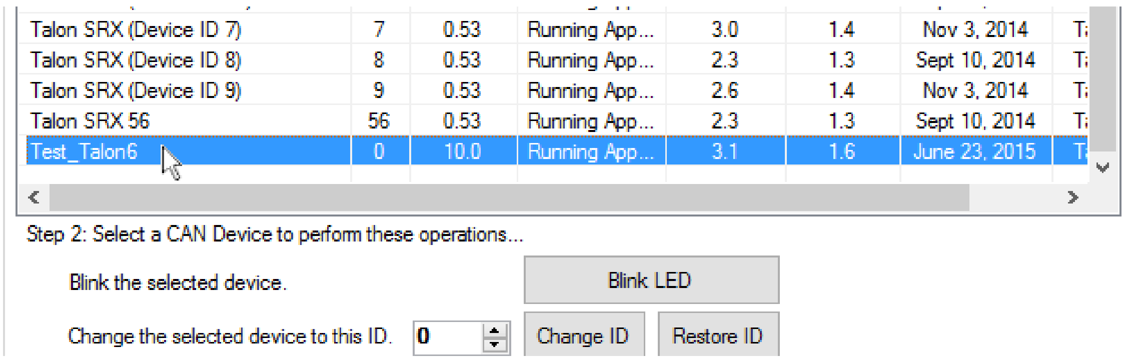
If a CRF is selected that is meant for an entirely different product, User will receive an error describing this. There is no possibility of permanently damaging a CTRE product by selecting mismatched firmware.

5.4.3. Updating Firmware – HERO LifeBoat

Detailed instructions on how to use HERO LifeBoat to field-upgrade a CTRE CAN Device can be found in the HERO User's Guide. Essentially open HERO LifeBoat and connect to the HERO via a USB-A-to-A cable.

Select the CAN device to update. The device can be a …

- CAN bus Pigeon
- CAN Talon SRX
- Pigeon connected to a Talon via the gadgeteer cable
- PCM
- PDP
- Future devices

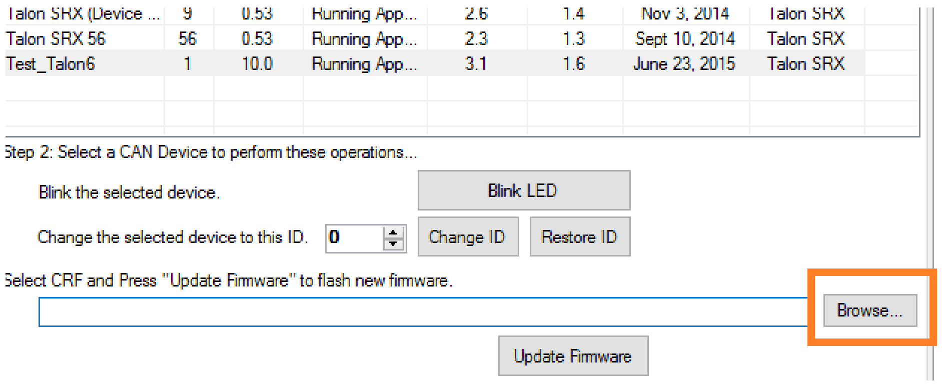


Talon SRX (Device ID 7)	7	0.53	Running App...	3.0	1.4	Nov 3, 2014	T
Talon SRX (Device ID 8)	8	0.53	Running App...	2.3	1.3	Sept 10, 2014	T
Talon SRX (Device ID 9)	9	0.53	Running App...	2.6	1.4	Nov 3, 2014	T
Talon SRX 56	56	0.53	Running App...	2.3	1.3	Sept 10, 2014	T
Test_Talon6	0	10.0	Running App...	3.1	1.6	June 23, 2015	T

Step 2: Select a CAN Device to perform these operations...

Blink the selected device.

Change the selected device to this ID. 0



Talon SRX (Device ...	9	0.53	Running App...	2.6	1.4	Nov 3, 2014	Talon SRX
Talon SRX 56	56	0.53	Running App...	2.3	1.3	Sept 10, 2014	Talon SRX
Test_Talon6	1	10.0	Running App...	3.1	1.6	June 23, 2015	Talon SRX

Step 2: Select a CAN Device to perform these operations...

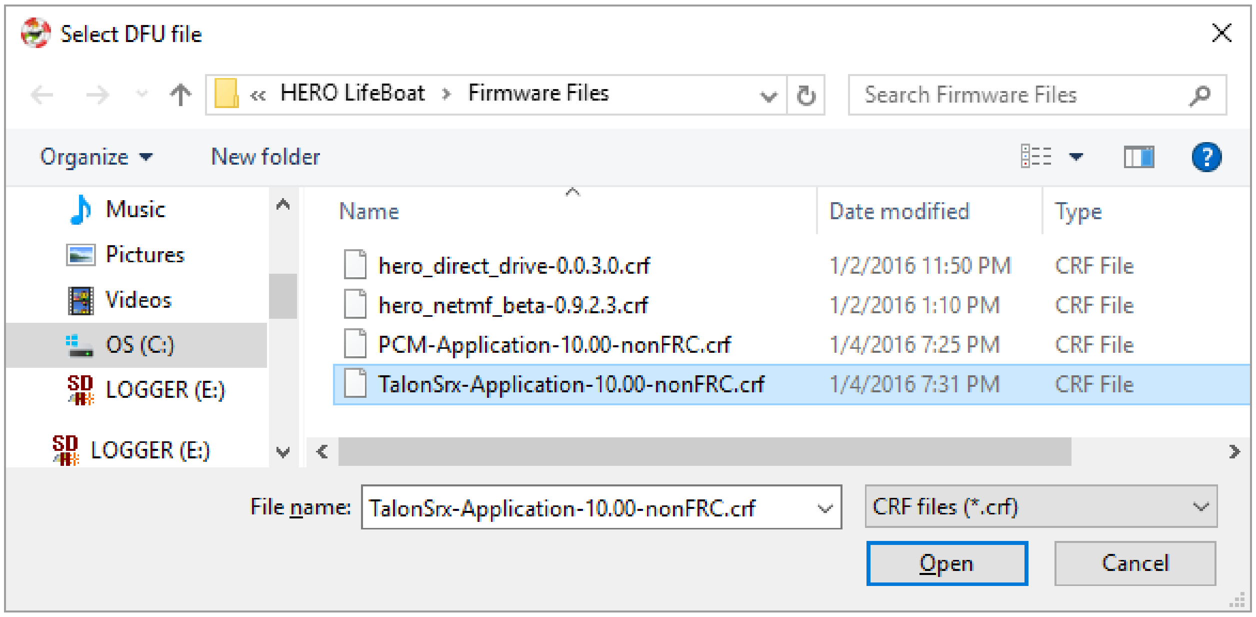
Blink the selected device.

Change the selected device to this ID. 0

Select CRF and Press "Update Firmware" to flash new firmware.

Navigate to the firmware tab and press the browse button to select the latest firmware for your device. See Section 5.4.1 above for where to find latest firmware.

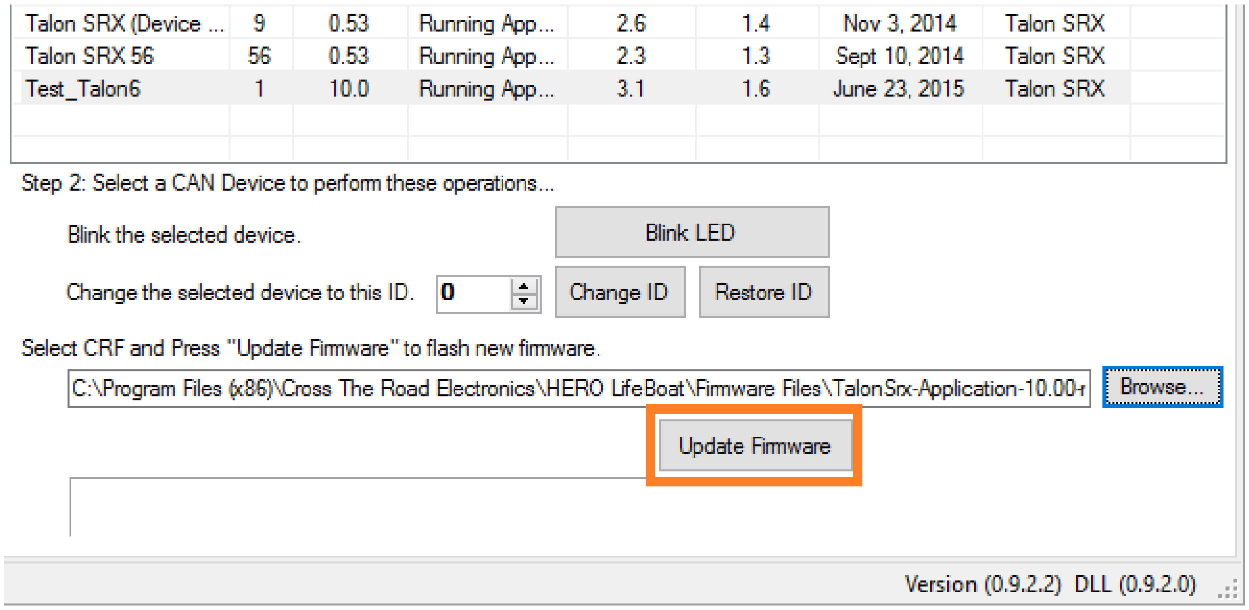
For example, if re-flashing a Talon SRX, choose the Talon SRX crf file. Then press "Open".



Name	Date modified	Type
hero_direct_drive-0.0.3.0.crf	1/2/2016 11:50 PM	CRF File
hero_netmf_beta-0.9.2.3.crf	1/2/2016 1:10 PM	CRF File
PCM-Application-10.00-nonFRC.crf	1/4/2016 7:25 PM	CRF File
TalonSrx-Application-10.00-nonFRC.crf	1/4/2016 7:31 PM	CRF File

File name: TalonSrx-Application-10.00-nonFRC.crf
CRF files (*.crf)

Next press "Update Firmware" to begin flashing the CAN device.



Talon SRX (Device ...	9	0.53	Running App...	2.6	1.4	Nov 3, 2014	Talon SRX
Talon SRX 56	56	0.53	Running App...	2.3	1.3	Sept 10, 2014	Talon SRX
Test_Talon6	1	10.0	Running App...	3.1	1.6	June 23, 2015	Talon SRX

Step 2: Select a CAN Device to perform these operations...

Blink the selected device.

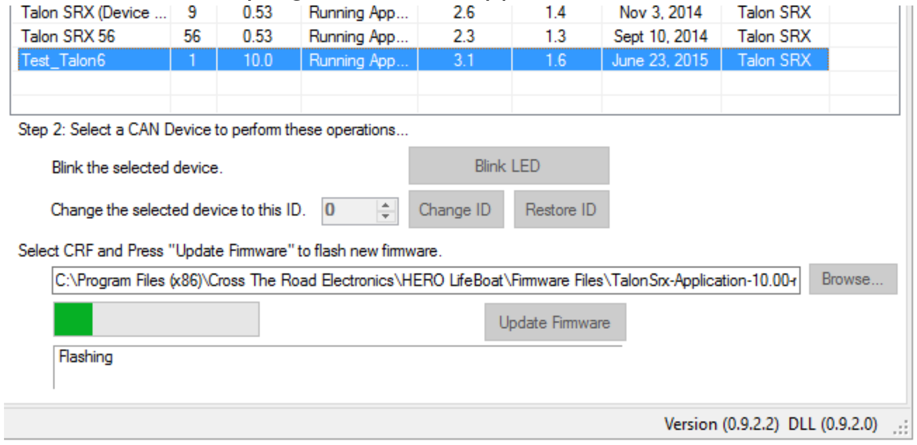
Change the selected device to this ID. 0

Select CRF and Press "Update Firmware" to flash new firmware.

C:\Program Files (x86)\Cross The Road Electronics\HERO LifeBoat\Firmware Files\TalonSrx-Application-10.00-

Version (0.9.2.2) DLL (0.9.2.0)

The interface will disable and a progress bar will appear.



Talon SRX (Device ...	9	0.53	Running App...	2.6	1.4	Nov 3, 2014	Talon SRX
Talon SRX 56	56	0.53	Running App...	2.3	1.3	Sept 10, 2014	Talon SRX
Test_Talon6	1	10.0	Running App...	3.1	1.6	June 23, 2015	Talon SRX

Step 2: Select a CAN Device to perform these operations...

Blink the selected device. Blink LED

Change the selected device to this ID. 0 Change ID Restore ID

Select CRF and Press "Update Firmware" to flash new firmware.

C:\Program Files (x86)\Cross The Road Electronics\HERO LifeBoat\Firmware Files\TalonSrx-Application-10.00 Browse...

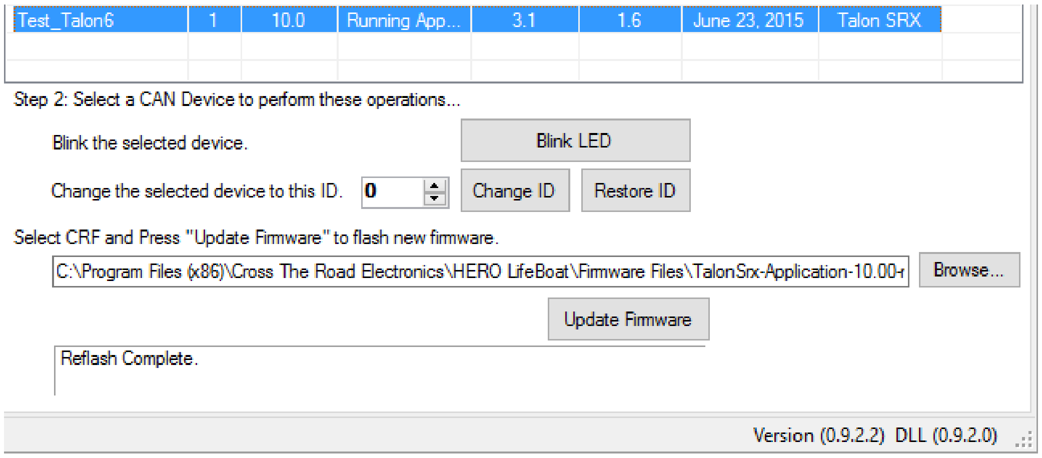
Update Firmware

Flashing

Version (0.9.2.2) DLL (0.9.2.0)

When the field-upgrade is complete, the following success message will show.
The entire flash session should take…

- ~10 seconds if device is directly on CAN bus.
- ~40 seconds if flashing Pigeon connected to Talon SRX via Gadgeteer data cable.



Test_Talon6	1	10.0	Running App...	3.1	1.6	June 23, 2015	Talon SRX

Step 2: Select a CAN Device to perform these operations...

Blink the selected device. Blink LED

Change the selected device to this ID. 0 Change ID Restore ID

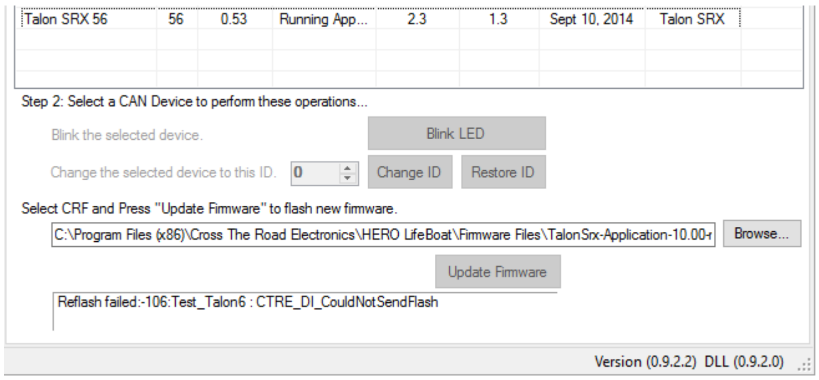
Select CRF and Press "Update Firmware" to flash new firmware.

C:\Program Files (x86)\Cross The Road Electronics\HERO LifeBoat\Firmware Files\TalonSrx-Application-10.00 Browse...

Update Firmware

Reflash Complete.

Version (0.9.2.2) DLL (0.9.2.0)



Talon SRX 56	56	0.53	Running App...	2.3	1.3	Sept 10, 2014	Talon SRX

Step 2: Select a CAN Device to perform these operations...

Blink the selected device. Blink LED

Change the selected device to this ID. 0 Change ID Restore ID

Select CRF and Press "Update Firmware" to flash new firmware.

C:\Program Files (x86)\Cross The Road Electronics\HERO LifeBoat\Firmware Files\TalonSrx-Application-10.00 Browse...

Update Firmware

Reflash failed:-106:Test_Talon6 : CTRE_DI_CouldNotSendFlash

Version (0.9.2.2) DLL (0.9.2.0)

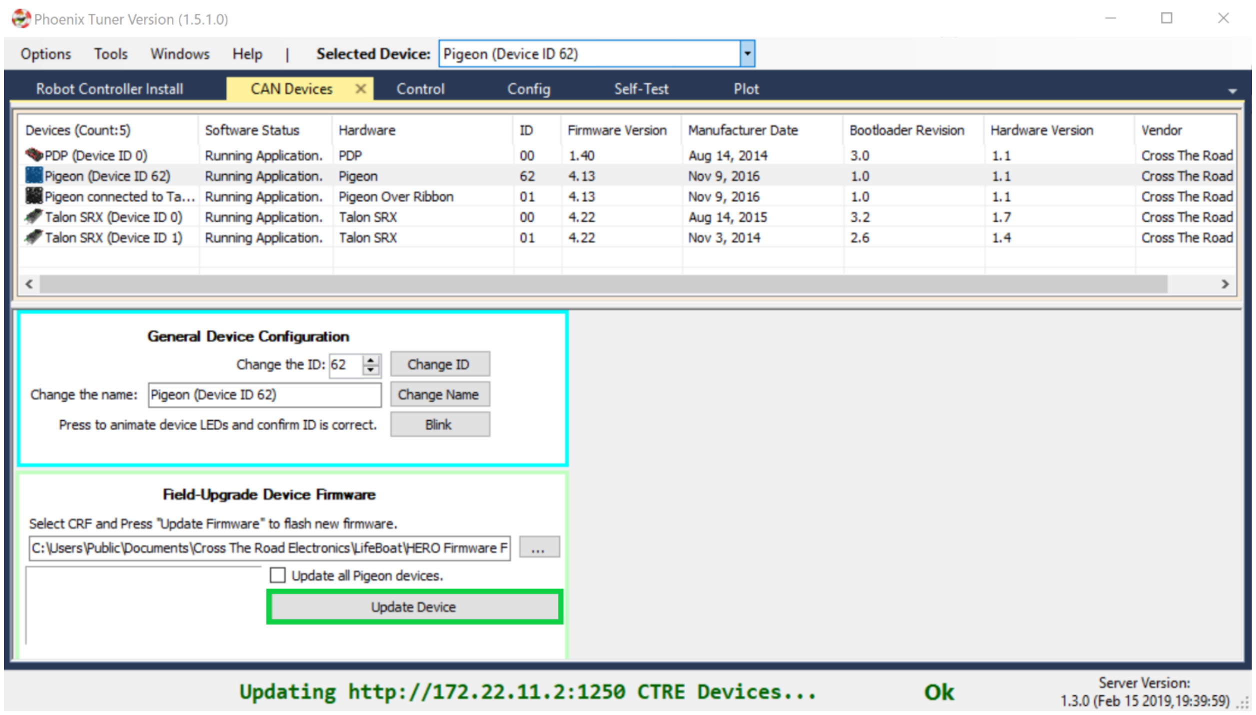
If power or communications is disrupted during the field-upgrade, LifeBoat will report the error and the CAN device will be left in bootloader mode (which is harmless). At which point you can simply try again after resolving the intermittent power/communication issue. In other words, the CAN devices are not susceptible to "bricking" due to incomplete flashes.

Above is an example of what to expect if CAN bus/Gadgeteer cable is disconnect mid-flash.

5.4.4. Updating Firmware – FRC roboRIO Phoenix Tuner

Open Phoenix Tuner. Select the Pigeon on the left and press the Update Firmware button. See the Section 5.4.1 above for firmware file locations.

Field-upgrade will take ~10 seconds if device is directly wired to CAN bus, ~20 seconds if device is a Pigeon connected via the Gadgeteer data cable.



Selected Device: Pigeon (Device ID 62)

Devices (Count:5)	Software Status	Hardware	ID	Firmware Version	Manufacturer Date	Bootloader Revision	Hardware Version	Vendor
PDP (Device ID 0)	Running Application.	PDP	00	1.40	Aug 14, 2014	3.0	1.1	Cross The Road
Pigeon (Device ID 62)	Running Application.	Pigeon	62	4.13	Nov 9, 2016	1.0	1.1	Cross The Road
Pigeon connected to Ta…	Running Application.	Pigeon Over Ribbon	01	4.13	Nov 9, 2016	1.0	1.1	Cross The Road
Talon SRX (Device ID 0)	Running Application.	Talon SRX	00	4.22	Aug 14, 2015	3.2	1.7	Cross The Road
Talon SRX (Device ID 1)	Running Application.	Talon SRX	01	4.22	Nov 3, 2014	2.6	1.4	Cross The Road

Change the ID: 62

Change the name: Pigeon (Device ID 62)

Press to animate device LEDs and confirm ID is correct.

Select CRF and Press "Update Firmware" to flash new firmware.

C:\Users\Public\Documents\Cross The Road Electronics\LifeBoat\HERO Firmware F

☐ Update all Pigeon devices.

```
Updating http://172.22.11.2:1250 CTRE Devices...        Ok
```

Server Version:
1.3.0 (Feb 15 2019,19:39:59)

5.5. Using robot API to test Yaw

After Pigeon (and possibly connected Talon SRXs) are update, a simple example can be written to confirm Pigeon is functioning correctly. Alternatively, examples can be downloaded from the **CTR-Electronics GitHub account.**

https://github.com/CrossTheRoadElec

5.5.1. Creating first Pigeon

To leverage the features of Pigeon, robot application must create a Pigeon object. One object should be created per Pigeon.

5.5.1.1. Creating first Pigeon – HERO – C#

Below is a C# example for creating a Pigeon object connected to CAN bus.

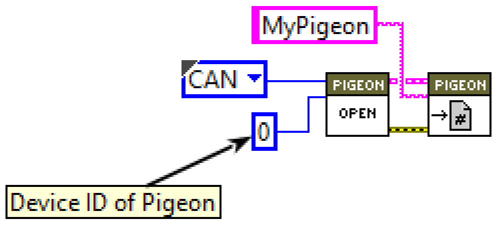
```
        CTRE.Phoenix.Sensors.PigeonIMU _pigeon;
        _pigeon = new CTRE.PigeonIMU(0); /* Pigeon is on CAN Bus with device ID 0 */
```

Below is a C# example for creating a Pigeon connected to a Talon SRX.

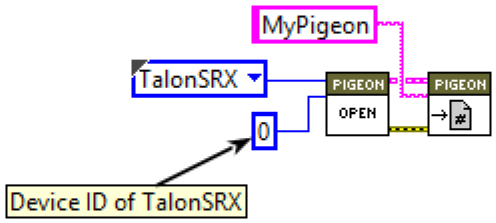
```
CTRE.Phoenix.MotorControl.CAN.TalonSRX _talon = new CTRE.Phoenix.MotorControl.CAN.TalonSRX
(0); /* make a talon with deviceId 0 */
_pigeon = new CTRE.Phoenix.Sensors.PigeonIMU(_talon); /* Pigeon is connected to _talon via
Gadgeteer cable.*/
```

5.5.1.2. Creating first Pigeon – FRC roboRIO – LabVIEW

Below is a LabVIEW example for creating a Pigeon object connected to CAN bus.



Below is a LabVIEW example for creating a Pigeon connected to a Talon SRX.



Note: The Pigeon Open VI may be changed in the future to be a polymorphic selection between CAN bus and Talon SRX connectivity.

5.5.1.3. Creating first Pigeon – FRC roboRIO – Java

Below is a Java example for creating a Pigeon object connected to CAN bus.

```
    PigeonImu _pigeon = new PigeonImu(0); /* example Pigeon with device ID 0 */
```

Below is a Java example for creating a Pigeon object connected to a Talon SRX.

```
    TalonSRX _talon2 = new TalonSRX(2); /* Talon SRX on CAN bus with device ID 2*/
    PigeonIMU _pigeon = new PigeonIMU(_talon2); /* Pigeon is plugged into Talon 2*/
```

5.5.1.4. Creating first Pigeon – FRC roboRIO – C++

```
_pidgey = new PigeonIMU(0); /* Pigeon is on CANBus (powered from ~12V, and has a device ID of zero */
_pidgey = new PigeonIMU(_spareTalon); /* Pigeon is ribbon cabled to the specified TalonSRX. */
```

5.5.2. Test Yaw using Programming API

The next step is to retrieve the Pigeon Yaw. This will indicate the Pigeon is connected and functioning correctly. From here the Software API can be leveraged to utilize Pigeon. Use the code snippet in Section 6.2 to acquire and display the Pigeon's reported Yaw value. Then rotate the Pigeon about the Z-axis to confirm change in heading.

6. Software API

6.1. Getting Status and Signals

Pigeon has several status signals that are reported periodically. These signals can be retrieved using any of the supported programming languages on the supported hardware platforms.

Included in the signals are…

- Pigeon State: Used to determine when Pigeon is finished initializing, and that Pigeon is connected.
- Yaw: Continuous angle (Degrees) of the robot's heading. Value will overflow at ±23040 degrees.
- Pitch: (Degrees: -90 to 90)
- Roll: (Degrees: -90 to 90)
- Compass Heading (Degrees) and Magnitude (µT)
- Fused Heading (Degrees) and Fusion Status (Booleans)
- Quaternion (6D)
- Accelerometer Angles (tilt angles in degrees between two axis, XZ, YZ, XY respectively)
- Magnetometer Axes (µT)
- Gyro Angles (Degrees) (accumulated using gyroscope only)
- Temperature (ºC)
- Uptime (ms): How long the Pigeon has been powered (Capped at 255 seconds).

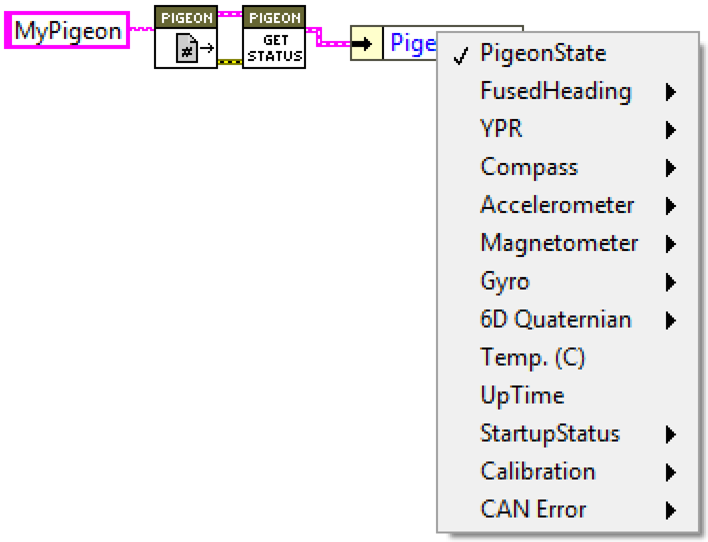
💡 TIP: Anytime critical signal data is retrieved from IMU to control motion, implementation should **check the health of the IMU's connection**. In other words, the robot application should use robot API to determine if Pigeon is physically disconnect/unpowered. Most of the Pigeon API returns integer error codes to determine this, and additional APIs exist to retrieve the "Last Error" and general status.

6.1.1. HERO – C#

```
CTRE.Phoenix.Sensors.GeneralStatus generalStatus = CTRE.Phoenix.Sensors.GeneralStatus();
_pigeon.GetGeneralStatus(generalStatus);
```

6.1.2. FRC roboRIO – LabVIEW

LabVIEW signals are bundled into a single cluster that can be obtained using the Get Status VI. Any of the available signals can then be viewed by unbundling the cluster by name.



6.1.3. FRC roboRIO – Java

```
PigeonIMU.GeneralStatus genStatus = new PigeonIMU.GeneralStatus();
_pigeon.GetGeneralStatus(genStatus);
```

6.1.4. FRC roboRIO – C++

```
PigeonIMU::GeneralStatus genStatus;
genStatus =  PigeonIMU::GeneralStatus();
Cimu->GetGeneralStatus(genStatus);
```

6.2. Retrieve Yaw, Pitch, Roll

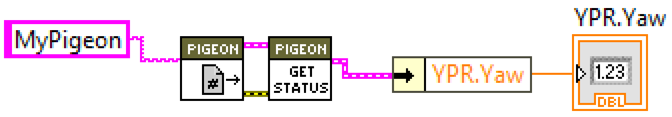
Yaw, Pitch, and Roll are typically represented as an array of three values, where the order of values is Yaw, Pitch, and Roll respectively.

6.2.1. Retrieve Yaw – HERO – C#

```
float [] ypr = new float [3];
_pigeon.GetYawPitchRoll(ypr);
 Debug.Print("Yaw:" + ypr[0] );
```

6.2.2. Retrieve Yaw – FRC roboRIO – LabVIEW

Below is a LabVIEW example for retrieving the Pigeon Yaw measurement.



6.2.3. Retrieve Yaw – FRC roboRIO – Java

```
double [] ypr = new double[3];
 _pigeon.GetYawPitchRoll(ypr);
System.out.println("Yaw:" + ypr[0]);
```

6.3. Set the Yaw/Fused Heading

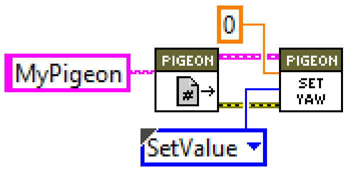
The Yaw and Fused Heading can both be modified – these values can be…

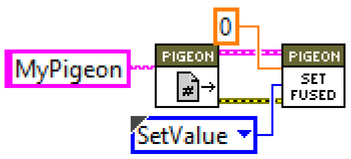
- Tared/zeroed or set to a constant for clean startup.
- Set an offset to the heading (for example declination on the compass).
- Atomically add offset with no accumulated error (add 90 degrees to current heading)
- Match compass

6.3.1. FRC roboRIO – LabVIEW

There are four Tare types that can be used to modify Yaw and Fused Heading. "SetValue" will set the signal to the indicated value. "SetOffset" will create an offset of the indicated magnitude and this offset will be added to the raw signal value in all read operations. "AddOffset" will add to the current offset value. This can be useful when it is necessary to periodically change your offset. "MatchCompass" will reset the signal value to match the current compass measurement.

Yaw and Fused Heading each have their own Set VI in LabVIEW and both function the same way. The Set VI is given the set value and Tare type for the desired operation. If the Tare type is not specified, the default is SetValue.





6.3.2. FRC roboRIO – Java

```
_pigeon.SetYaw(newAngle);
_pigeon.SetFusedHeading(newAngle);
```

6.3.3. HERO C#

```
_pigeon.SetYaw(newAngle);
_pigeon.SetFusedHeading(newAngle);
```

6.4. Entering Calibration Mode(s)

The various calibration modes of the Pigeon can be initiated programmatically.
Care should be taken when doing this to ensure robot application does not inadvertently put Pigeon into a calibration mode during normal use of the robot.

It is strongly recommended to read section 10 before using this function.

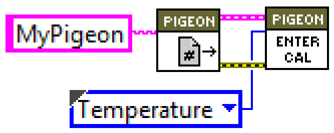
6.4.1. HERO – C#

```
_pigeon.EnterCalibrationMode(CTRE.Phoenix.PigeonImu.CalibrationMode.Temperature);
```

6.4.2. FRC roboRIO – Java

```
_pigeon.EnterCalibrationMode(CalibrationMode.Temperature);
```

6.4.3. FRC roboRIO – LabVIEW



6.5. Enabling/Disabling Temperature Compensation

Although Temperature compensation is enabled by default (at power up), it may helpful for testing to momentarily disable Temperature compensation.

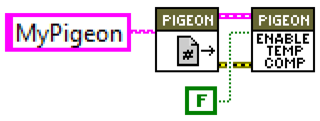
This API has no effect if Temperature Calibration procedure has not been followed. This is because Temperature Compensation is applied if the calibration was not done.

The general status API can be used to track the how often Temperature Compensation is applied.

6.5.1. FRC roboRIO – Java

```
_pigeon.configTemperatureCompensationEnable(false,10);
```

6.5.2. FRC roboRIO – LabVIEW



7. LED Table

Condition	Color	Blink Pattern	Suggestions
Not powered	**Off**	**Off**	Plug in Power
Powered but no communication.	**Red**		If using Talon-Gadgeteer-cable, update Talon to latest firmware. Inspect Gadgeteer ribbon cable. If using CAN bus, check CAN wiring and termination.
CAN Connected	**Orange**		
Change in USB Connectivity	**Green**		
Boot-Calibrating		**Offset Blinking** (1)	
Boot-Calibration Complete		**Level Blinking** (2)	
Pigeon is in Bootloader	**One LED is Green/Orange Other LED is OFF.**		Flash firmware using HERO LifeBoat or Phoenix Tuner.
Hardware failure	**One LED is Red Other LED is Orange**		Hardware is damaged.

NOTE 1: Each LED alternates between on and off. Only one LED is on at any moment.
One LED will stay on for **one second**, then the other LED will illuminate for a **short moment**.
2: Each LED alternates between on and off. Only one LED is on at any moment.
Each LED takes the **same amount of time**.

8. IMU Error Sources

All IMUs are susceptible to accrued error due to the sources of error listed below. However, this can be avoided by following a few basic principles and careful IMU placement. The suggestions below are applicable to many IMUs, including those available in FRC Robotics.

Below are also instructions for how to root-cause each source of error using the features of Pigeon.

8.1. Location – Center of rotation

Due to the fusion between the accelerometer and the gyroscope, excessive and sustained G-force can impact the Yaw. This can be resolved by moving the IMU closer to the COR (center of rotation).

This root cause can be confirmed by printing/plotting the accelerometer magnitude while spinning the robot at max velocity. Ideally the magnitude should be as close to 1G as possible.

Alternatively, the accumulated gyro Z value can be printed/plotted while performing a sustained-high-speed rotation. If the accumulated gyro Z value is correct but the Yaw value is not, then this sufficiently confirms the root-cause.

Procedure…

- First drive the robot against an immoveable flat obstacle. A wall works best against the robot's flattest surface.
- Zero the yaw and accumulated gyro Z register.
- Drive the robot in a zero turn at max speed for ~30 seconds.
- Drive the robot in the opposite direction for ~30 seconds, or until robot approaches zero heading.
- Drive back up against the flat obstacle (wall) and read Yaw and Gyro Z values.
- If root-cause is confirmed (Yaw is incorrect and Gyro-Z is correct), move IMU closer to COR, and repeat procedure.

8.2. Temperature

IMUs (in general) also drift due to subtle changes in temperature. This is primarily due the temperature's impact on the zero-bias values for the gyroscope. However, temperature can also affect accelerometer and magnetometer.

One strategy to reduce this is to continually re-bias the gyroscope whenever a no-motion event is detected (Pigeon does this as well as other market IMUs). This combined with an environment where temperature does not change rapidly can be adequate depending on the application.

Another strategy is to on-the-fly compensate for changes in temperature. Pigeon supports this as well but requires the user to perform (once) a simple temperature calibration (see Section 10.3).

The impact of temperature can be confirmed by taking a source of heat (such as a small lamp) and heating the IMU while printing/plotting the temperature signal and relevant IMU signals. For example, plotting temperature and Yaw will reveal if temperature is affecting the Yaw measurement. See Section 10.3 for an example of this.

8.3. Avoid Magnetic/ferromagnetic materials

Most IMUs have some type of compass calibration procedure, including Pigeon. However, if the IMU is placed in an area with severe magnetic disturbance, then minimally the benefit of compass is reduced, and maximally the robot heading may be severely incorrect. Although many IMUs (including Pigeon) will automatically detect invalid magnitude field strengths, depending on the direction of the distortion, the compass can still be affected. For this reason, compass-features are often utilized in an outdoor setting in a chassis designed to accommodate compass features.

The common disturbances are due to proximity to ferromagnetic materials such as iron/steel and proximity to magnetic materials (motors). Additionally, if an IMU is placed within an extreme magnetic field (e.g. proximity to neodymium, rare-earth magnets), the hard-iron offsets within the magnetometer may permanently change, requiring a new calibration procedure. Care should be taken to ensure such magnets are not placed near IMU PCB (e.g. CTRE Mag Encoder Magnets).

This impact can be root-caused by waving a simple mechanical compass in the presence of where the IMU is intended to be placed. Look for any major changes in the needle position as the compass is waved near the critical areas of the robot, while maintaining consistent orientation between compass and Earth.

Next place the mechanical compass in the location where the IMU is meant to be placed. While the robot is placed on blocks, drive all motors, chains, and articulators while watching the compass. If the compass needle reacts to robot actuation, then IMU should be relocated, or simply avoid relying on compass features if the application will allow it.

Phone compass applications (software) can be used to a lesser degree, however a mechanical compass is best, and are widely available for <$5.

Another means of confirming this source of error is to print/plot the compass magnitude as reported by the IMU while moving the robot into various poses. This can be compared with the

expected magnetic field strength which can be determined with a Phone compass application, or with an online search such as…
http://www.ngdc.noaa.gov/geomag-web/
Any massive dip or rise in the magnetic norm may indicate magnetic disturbance.

However, it should be noted that there can be magnetic disturbance despite the norm not changing by much. For this reason, the mechanical compass testing above is recommended.

8.4. Vibration

Vibration of an IMU can cause errors by contributing noise to the accelerometer, gyro, and magnetometer. Additionally, if the vibration is severe enough, a sensor input may see a saturated value. A common example of this is vibration in the airframe of a UAV/drone (for example, and unbiased propeller).

It is generally recommended to include some form of vibration dampening when mounting any IMU. Rubber grommets are included with Pigeon to be used inside the mounting holes as vibration dampeners.

9. Status Rates

A future update will provide an API to modify the status frames of the various signals provided by Pigeon.

The default update rates are 100 Hz(10ms) for yaw, Pitch, Roll, and all fusion-related signals. All other signals default to 100ms unless overridden with robot API.

10. Calibration Modes

The Pigeon supports several calibration modes. The benefits of, and procedure of, each Calibration mode is documented below.

10.1. Boot Calibration

The Pigeon performs a boot calibration procedure on power up. This includes initial biasing of the gyroscope and correcting for temperature (if temperature calibration procedure was followed once). Typically, this takes four seconds and requires the Pigeon to be reasonably flat at startup. This is done automatically and requires no involvement with the user.

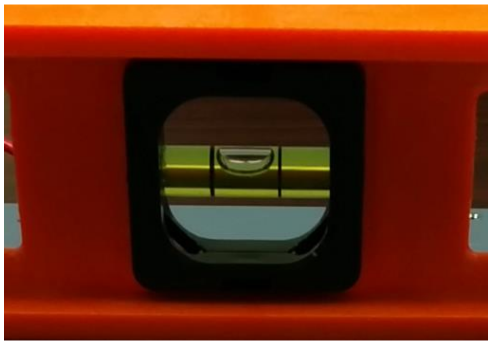
While boot calibrating, the LED pattern on the Pigeon will favor one side's LED over the other side's LED. (See Section 7). When boot calibration is finished, the LED pattern will change to be symmetric (See Section 7).

10.2. Accelerometer Calibration

This calibration procedure finds initial biases to apply to the accelerometer. Although the IMU chip is factory calibrated at the silicon manufacturer, Pigeons are re-factory calibrated with this procedure. The procedure can be redone out in the field, however it is not expected to be necessary.

The procedure is to first find a level surface. This must be scrutinized with a leveler…

… and should be checked at various angles to ensure table is perpendicular to gravity in all orientations.

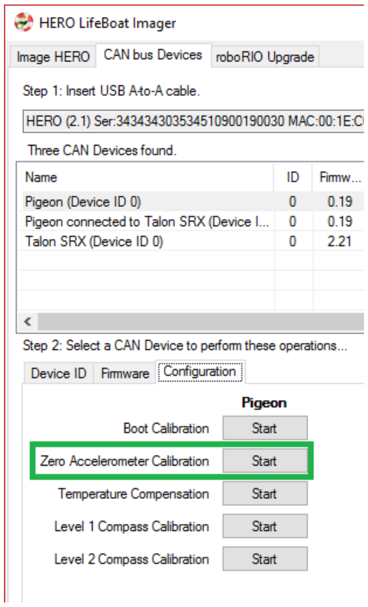


Bubble-based levelers can be used. However, care should be taken to ensure bubble is as centered as reasonably possible.

If the bubble edge is in "contact" with either black line on either side, that will yield typically 0.5 degrees of error in pitch or roll.

Once leveled, Pigeon must be placed flatly against the table. For this reason, using the Gadgeteer cable connection strategy is ideal for this calibration.

Using robot API, the EnterCalibration(Accelerometer) routine can be used to put IMU into this mode. When this routine is called (once) the Pigeon will perform a longer-than-usual boot-calibration, then blink solid-green for a minute. Once finished Pigeon will again boot-cal and the procedure is complete.



HERO LifeBoat Imager

Image HERO | CAN bus Devices | roboRIO Upgrade

Step 1: Insert USB A-to-A cable.

HERO (2.1) Ser:343434303534510900190030 MAC:00:1E:C

Three CAN Devices found.

Name	ID	Firmw...
Pigeon (Device ID 0)	0	0.19
Pigeon connected to Talon SRX (Device I...	0	0.19
Talon SRX (Device ID 0)	0	2.21

Step 2: Select a CAN Device to perform these operations...

Device ID | Firmware | Configuration

Pigeon

Boot Calibration	Start
Zero Accelerometer Calibration	Start
Temperature Compensation	Start
Level 1 Compass Calibration	Start
Level 2 Compass Calibration	Start

Alternatively, the configuration page in HERO LifeBoat can also be used to start calibration when using HERO.

Additionally, Phoenix Tuner Self-Test can be used to get updated detailed information on the progress of the calibration. The entire procedure should take ~20 seconds.

10.3. Temperature Calibration

Temperature calibration generally improves all aspects of the Pigeon. The primary benefit is the additional automatic gyro re-biasing that occurs in the background (though this also occurs after 8 seconds of no-motion). The impact of temperature on IMUs in general is explained in Section 8.2.

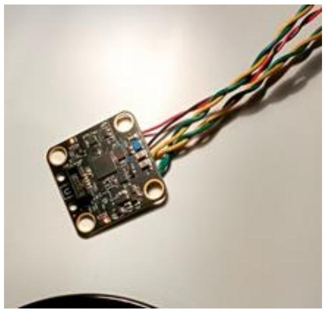
The benefit of temperature calibration also ranges from chip-to-chip. In other words, the temperature calibration may not improve a particular Pigeon measurably. For this reason, it is useful to first confirm how measurable the impact of temperature change is.

10.3.1. Measure the impact of temperature changes

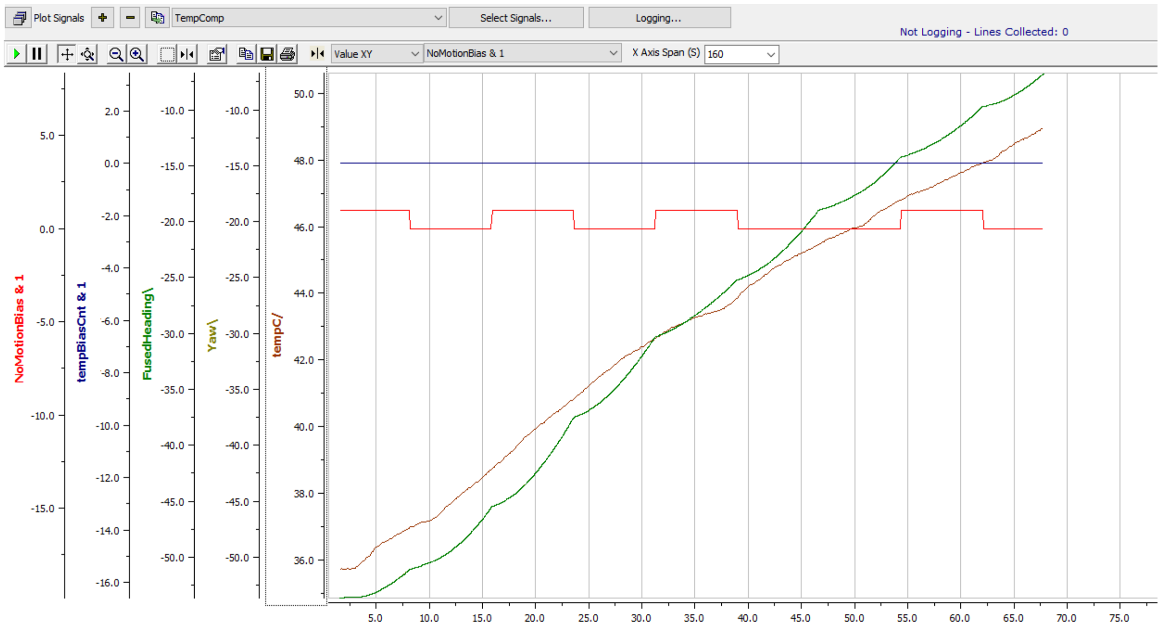
The user can first observe the impact of temperature by cleanly booting up system and observing the critical signals (for example Yaw). This can be done with the Phoenix Tuner self-test, or by plotting/printing the signals.

In the example below a heat lamp was placed on top of the Pigeon for the purpose of raising its temperature.

A simple off the shelf halogen desk lamp is sufficient. This off the shelf lamp was $5 at a local store.



After Pigeon has "settled" (boot calibrated), lamp was placed to focus its light onto the Pigeon directly. After 60 seconds the observed change in temperature is ~14 C' (35'C => 49'C), while the Yaw changes 40 degrees. Note that during this period the no-motion calibration occurred seven times (red edges)



10.3.2. Perform Calibration

To calibrate, first allow the Pigeon to cool. This can be confirmed by monitoring the temperature signal.

Once cool, position the Pigeon so that it can remain still (reasonably flat surface is ideal).

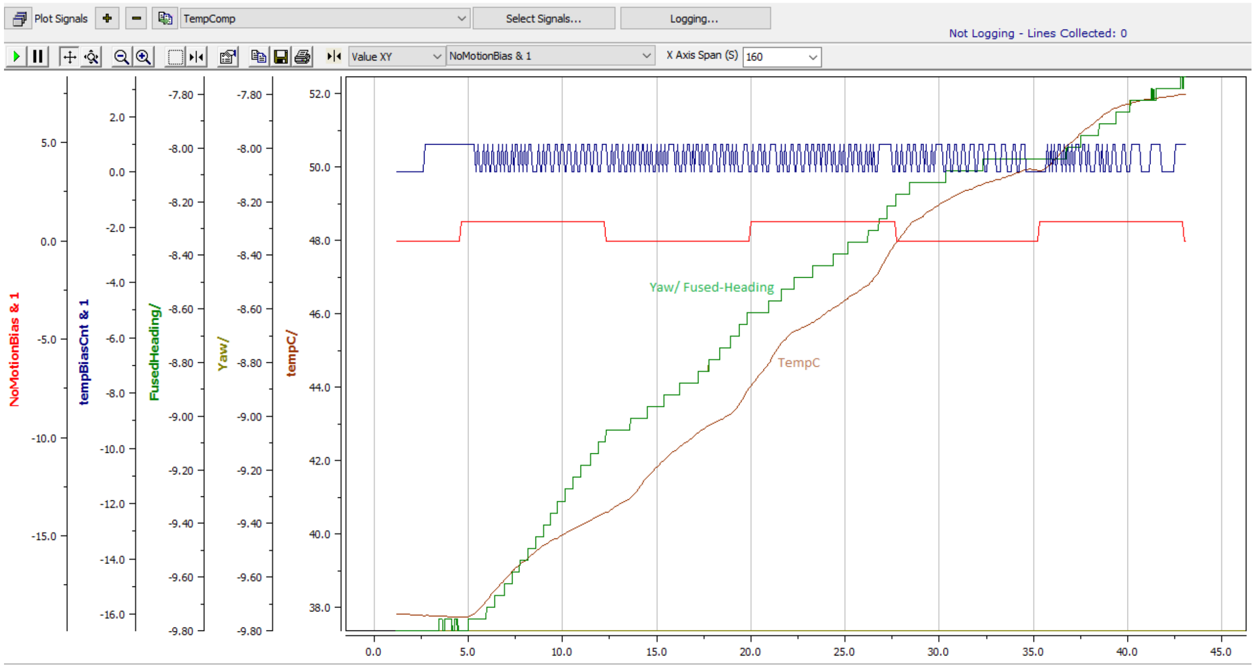
Then use the robot API (or HERO LifeBoat) to enter temperature calibration mode.
Pigeon will perform a longer-than-normal boot-cal. Once complete, the Pigeon LEDs will both turn on at the same time, and will appear bright orange (brighter than typical). In this state Pigeon will generate heat to speed up the calibration.

At this point focus the lamp's light onto the Pigeon without moving/touching the Pigeon (otherwise temperature calibration may take several minutes). Once the Pigeon has seen a sufficient range in temperature it will momentarily blink green, then cleanly boot-calibrate (same as power boot).

10.3.3. Re-Measure the impact of temperature

Once Pigeon has boot-calibrated, repeat the data collection to confirm temperature calibration is being applied.

In the capture below there are many temperature compensation events (blue edges) and six no-motion gyro re-bias events (red edges). After a temperature rise of 14C' (38'C => 52'C), the Yaw changes by 2 deg, a considerable improvement from the pre-calibration capture.



10.4. Compass Calibration

The Pigeon IMU compass calibration procedure is not documented here though it is supported in firmware. In testing the Pigeon IMU and other IMU devices it was found that the magnetometer typically causes more problems than it solves. Placement near magnetic/ferrous materials is common and can wreak havoc on compass readings, particularly in an indoor environment. If you rely on magnetometer, your placement requirements become complex. Finding a spot on a robot that is center-of-rotation and far from steel/magnets can be difficult.

It is far more beneficial to perform the thermal calibration strategy as it is simple to perform, only needs to be done once, and temperature is a far greater contributor of IMU error in FRC than anything else, second to maybe center-of-rotation placement.

Note that the reported compass value will always be 0. This will not affect the FusedHeading value which ignores the compass when the compass is not calibrated.

If using a Pigeon IMU in a non-FRC application where you would like to calibrate the compass, contact CTRE support.

11. Troubleshooting Tips and Common Questions

11.1. Are there any more source examples?

CTR Electronics maintains several FRC and HERO examples at our GitHub account.

https://github.com/CrossTheRoadElec

This includes several projects with Pigeon IMU.

11.2. Why are there colored dots on the Pigeon? Is something wrong with the hardware?

There is nothing wrong with the Pigeon. In production, each Pigeon goes through several test procedures to ensure quality. Each color represents a specific phase that the Pigeon went through.

11.3. What's the difference between FusedHeading and Yaw?

Yaw is calculated using the Gyro and Accelerometer only. In many cases this is adequate. However if Compass-fusing is required, FusedHeading can be used instead.

Additionally, the user can disable compass fusing so that FusedHeading tracks identically with Yaw without having to modify robot application to read Yaw. This is useful if user needs to compare robot performance with and without compass fusing without considerable modification of the robot application.

If Pigeon has never undergone Compass Calibration, then FusedHeading and Yaw are effectively equivalent.

11.4. How much CAN bandwidth does Pigeon require?

Pigeon generally uses ~6% CAN bus utilization. This can be modified by using the SetStatusFrameRate API.

11.5. Pigeon is connected via Gadgeteer cable to Talon SRX, but Pigeon LEDs are red.

Talon SRX has too-old firmware. Follow Section 5.4 to update Talon to latest firmware. The procedure is identical to field-upgrading a Pigeon. Talon firmware requirements are in Section 5.3.1.1.

11.6. When I spin the robot at high speeds, the Yaw error is considerable. But when the robot spins slowly, the Yaw is correct.

See section 8 to rule possible sources of error.

11.7. I want to use Pigeon outdoors. Are there are tips or concerns?

Due to the variety of temperature it is recommend to perform the Temperature Calibration procedure.

11.8. My compass reads 0 all the time.

Compass features require the user to perform Compass Calibration. This is currently not supported (see Section 10.4).
Use the Yaw or FusedHeading signal instead. FusedHeading will properly ignore compass heading in fusion algorithm if compass calibration has never been performed.

12. Functional Limitations

Functional Limitations describe behavior that deviates from what is documented. Feature additions and improvements are always possible thanks to the field-upgrade features of the Pigeon.

12.1. Pigeon Orientation is Z-up

At the time of writing the Pigeon must be orientated Z-up in respect to Earth. Future firmware updates will include the ability to change the base orientation.

12.2. Missing APIs in Examples/Documentation

The following APIs are missing in Section 6. They will be added in future document updates.
-Example for Disabling Compass Fusion if need be.
-Changing Default status frame rates
-Temp Compensation Disable for certain languages
-Missing example for tracking the no-motion gyro re-bias events and temperature-compensation events.
-Missing example for detecting Pigeon loss of connection.
-Missing example for the raw signal getters (Magnetometer-uT, accelerometer-g, gyro-dps).

12.3. No Sticky Fault support in firmware

This will be released in future firmware update.

13. Pigeon CRF Firmware Revision Information

Version	Date	Description
20.0	02-Jan-2020	Re-compiled for the 2020 FRC Season
4.13	31-Jan-2019	Fixed issue where Motion-Profile-Arcing with a Pigeon-over-ribbon-cable would behave as though Pigeon is not connected (neutral drive). Talon SRX / Victor SPX would report missing remote device in self-test even though Pigeon was present.
4.0	19-Dec-2018	Updates to support new API features
0.41	27-Dec-2017	Now supports remote sensors
0.19	21-Nov-2016	Added the ability to change status frame periods. Added a way to disable TempComp for testing.
0.18	20-Nov-2016	Added accumulated Gyro angles for X, Y, Z.
0.17	19-Nov-2016	Initial release

14. Document Revision Information

Version	Date	Description
0.1	29-Nov-2016	Initial Release
1.0	7-Feb-2020	Update code-snippets to Phoenix API Remove references to roboRIO web dash and replace with Phoenix Tuner. Update Section 10.4