# Precision GNSS Module
# Evaluation Kit
# User Manual
**v1.2**

## Introduction

This document provides setup and usage reference for the Swift Navigation's Precision GNSS Module (PGM) Evaluation Kit.

The Evaluation Kit - featuring Swift's mPCIe format PGM receiver installed in an IoT computer platform - provides a turn-key demonstration of Swift's precision positioning solution. The PGM provides real-time precision GNSS and IMU measurements, and is designed specifically for Swift's Starling® positioning engine running on a host application processor. When used with Swift's Skylark™ cloud-based precise corrections service, the PGM Evaluation Platform (PEP) delivers even stronger performance and higher accuracy levels. The Evaluation Kit is ideal for customers building industrial, last mile and Internet of Things (IoT) platforms that require the ability to quickly install on a target platform and test performance in the field.

The PGM Evaluation Kit includes everything needed to install this GNSS system in your application and quickly get started with precise RTK positioning. It also comes with a 30-day Skylark trial.

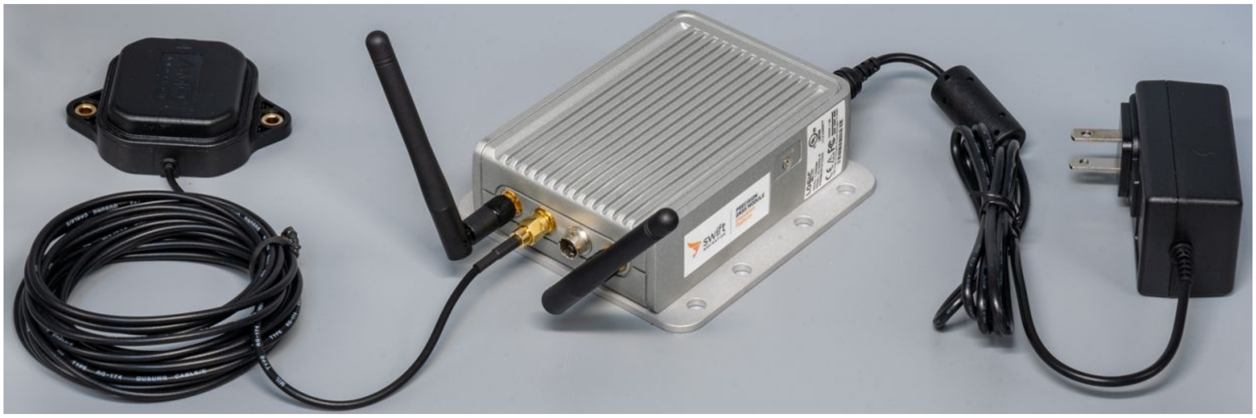


Fig 1. PGM Evaluation Platform

## Table of Contents

# Evaluation Kit Content

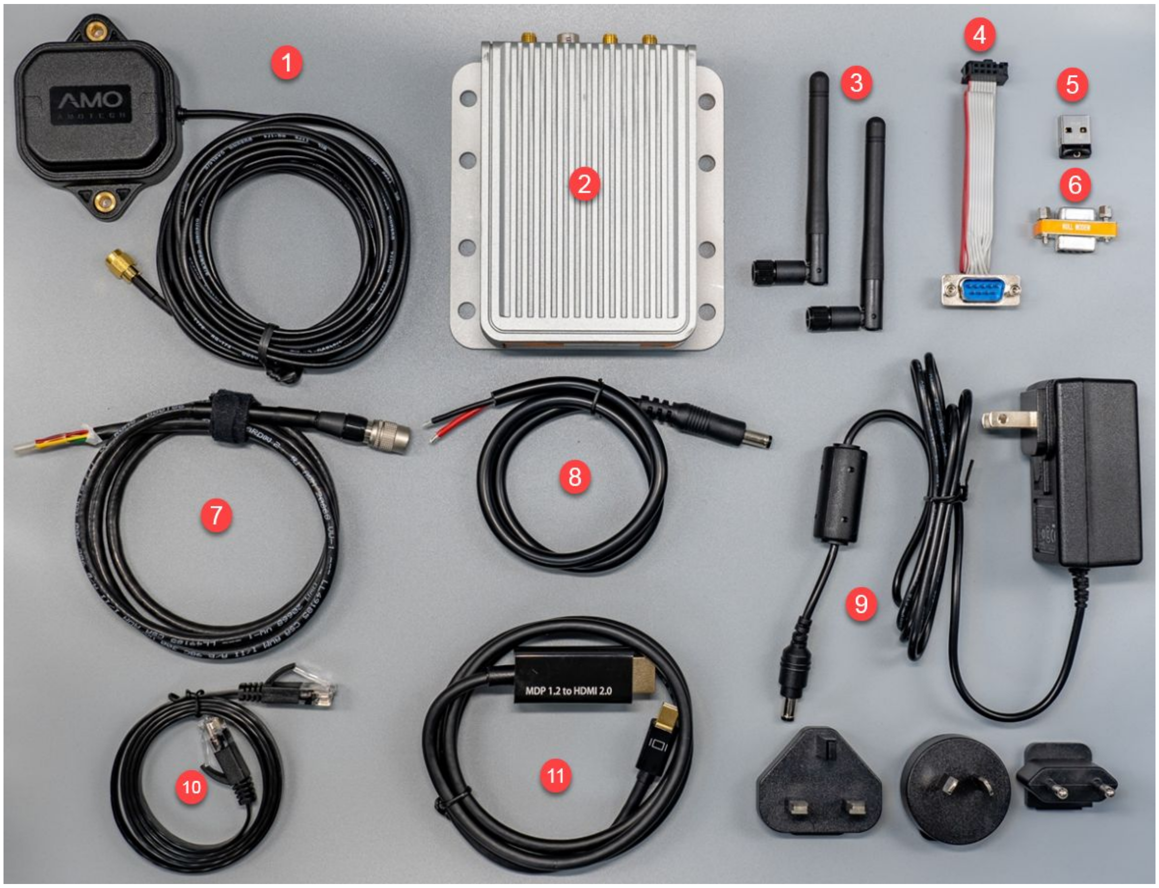


Fig 1. PGM Evaluation Kit Content

1. GNSS L1/L5 magnet-mount antenna
2. PGM Evaluation Platform (PEP)
3. Wi-Fi antennas
4. RS232 serial port adapter cable
5. USB drive for data recording
6. RS232 null modem for serial port connection
7. Vehicle signals cable
8. Power supply cable for direct 12 V supply
9. 12 V DC power supply 100-240 V AC input with international plugs
10. RJ45 Ethernet cable
11. DisplayPort to HDMI adapter cable

## PGM Evaluation Platform

The PGM Evaluation platform is based on the Onlogic CL200 series industrial computer. The Swift Navigation PGM module occupies one of the internal mPCIe slots. Additionally, the platform contains a Wi-Fi / Bluetooth mPCIe module for wireless connectivity, RS232 serial port for GNSS data output, Ethernet port, display port and three USB ports. Platform runs Ubuntu 20.04 Linux server out of the micro SD card.

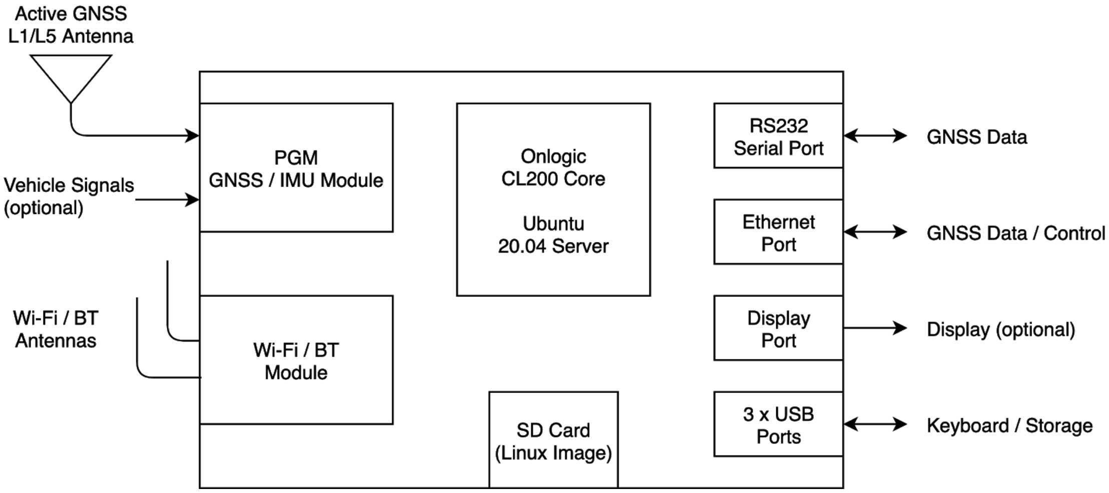


Fig 2. PGM Evaluation Platform Block Diagram

By default, the Wi-Fi module is configured as an access point enabling users to connect to the platform from any modern PC/laptop to configure the system and receive GNSS position data.

The Wi-Fi module can simultaneously work as a client to connect to a cell modem device for Internet access to receive GNSS corrections data. Note: both access point and Wi-Fi client must use the same Wi-Fi channel.

By default, the platform starts after applying power. To stop the platform use the power button or the terminal command for a graceful system shut down or just disconnect the power. Disconnecting power may result in data corruption, especially if recording.

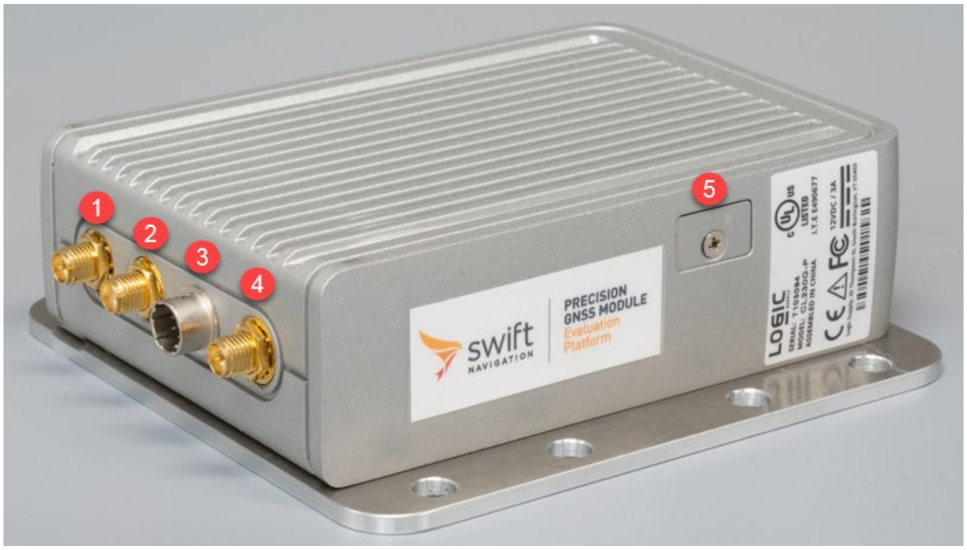


Fig 3. PEP Connectors

1. Wi-Fi antenna RP (Reverse Polarity) SMA connector
2. GNSS antenna SMA connector
3. Vehicle signals receptacle (Hirose HR10A-7R-6P(73))
4. Wi-Fi antenna RP SMA connector
5. Micro SD card slot (Linux image)

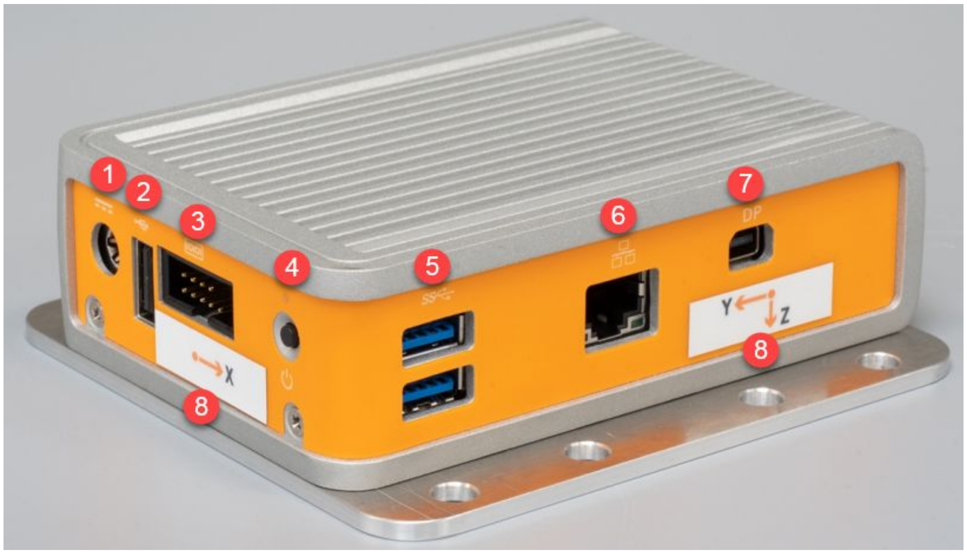


Fig 4. PEP Connectors

1. 12 V power supply 2.5/5.5 mm barrel jack connector
2. USB 2.0 port connector
3. RS232 serial port connector (use adapter cable for DE 9 connector)

4. Power button and power status LED
5. Two USB 3.0 port connectors
6. Ethernet port RJ45 connector
7. Mini DisplayPort connector
8. IMU orientation labels. The orange dot indicates the IMU center.

## Operating Conditions

1. Supply voltage: 12 V DC.
2. Power consumption: about 6 W during normal operation.
3. Operational temperature: 0 to 40°C with a non-condensing relative humidity of 10 - 90%.
4. The device shall not be used outdoors without an additional environmental protection.

## Installation in the Vehicle

For proper system operation it is essential to mount both the PEP and the GNSS antenna securely and firmly to the vehicle body. During operation, the antenna and PEP must remain in the same position relative to each other (i.e., both must be mounted on the same frame).

The GNSS antenna needs to be mounted on the vehicle where there are no obstructions and with a correct ground plane. Typical car’s roof is a good enough ground plane. Place the antenna at the center of the roof as much as possible. Do not place the antenna close to the roof edge. If the ground plane is not available by vehicle construction, place a round or square metal plate of 10 to 15 cm (4” to 6”) radius under the antenna.

To maximize inertial sensors sensitivity, mounting PEP in an orientation orthogonal to the vehicle-body is recommended. This means that PEP should be mounted with all three axes forming angles in multiples of 90° (0°, 90°, 180°, 270°), with respect to the primary direction of vehicle motion.

For easier installation and setup, it is recommended to align the X-axis direction of the PEP with the primary direction of vehicle motion.

PEP must be rigidly mounted to the body of the vehicle. Any vibration which is not directly related to vehicular motion will degrade the quality of the inertial data. Installing PEP on a flexible vehicle rooftop, engine cover, or fender - where the mounting surface can flex and vibrate independently of the vehicle body - should be avoided. Similarly, placing the sensor on plush seating of a vehicle with the antenna on the rigid part of the vehicle will yield poor results.

Record lever arm X, Y and Z vectors after installation. Measure from the orange dots printed on the PEP enclosure to the center of the antenna.

Refer to Appendix A for Starling settings euler angles of typical installations.

# Evaluation Platform Connections

The PEP platform requires connections to power source, GNSS antenna, control/recording computer, and correction data source. Connections to the correction data and control/recording computer may be wired or wireless.

## Power Connections

The platform requires 12 V DC power supply with a rate of 3 A. Use included in the kit wall power adapter or other power supply device with 12 V / 3 A DC output.

## GNSS Antenna Connections

PEP requires an active GPS/Galileo/BeiDou L1/L5 antenna. The device provides 3.3 V antenna bias voltage through the SMA antenna connector (max. 100 mA). For the best results the antenna LNA gain should be between 15 to 25 dB and with NF < 2-3 dB.

## Vehicle Signals

Optionally, the Vehicle Speed Signal (VSS) and Reverse can be connected to the platform for improved Dead Reckoning performance. See Appendix D for connection details.

## Control/Recording Computer Connections

Use one of the following methods to control the platform:

1. Connect your PC/laptop over Wi-Fi to PEP's access point named **SwiftNav-PEP-xxxx**, where xxxx is the last 4 digits of the device serial number (printed on the device side). The Wi-Fi password is `swiftnav`. Upon successful connection open SSH client (like PuTTY on Windows) and connect to PEP using the IP address **10.42.0.1**.
2. Connect your PC via Ethernet. By default, PEP Ethernet is set to the DHCP mode and therefore PEP must be connected to a network router with a DHCP server. Check in the router settings for the assigned IP address or scan the network for the device IP. Once IP is known, open SSH client (like PuTTY on Windows) and connect to PEP using the assigned IP address.
3. Connect a monitor to the DisplayPort and a USB keyboard to control the platform.

## Correction Source Connections

To obtain Skylark corrections data PEP needs Internet access. Use one of the following methods to enable PEP's Internet access:

1. Connect PEP to the Internet access point over Wi-Fi (cell modem with router, cell phone with Personal Hotspot, etc.). The connection must use the same Wi-Fi channel as the PEP's Wi-Fi access point.
2. Connect PEP over the Ethernet to the cell modem / router with Internet access. By default, PEP Ethernet is set to the DHCP mode and therefore PEP must be connected to a network with a DHCP server. PEP Ethernet network can also be set to static IP if required for networks without a DHCP server.

# Evaluation Platform Usage

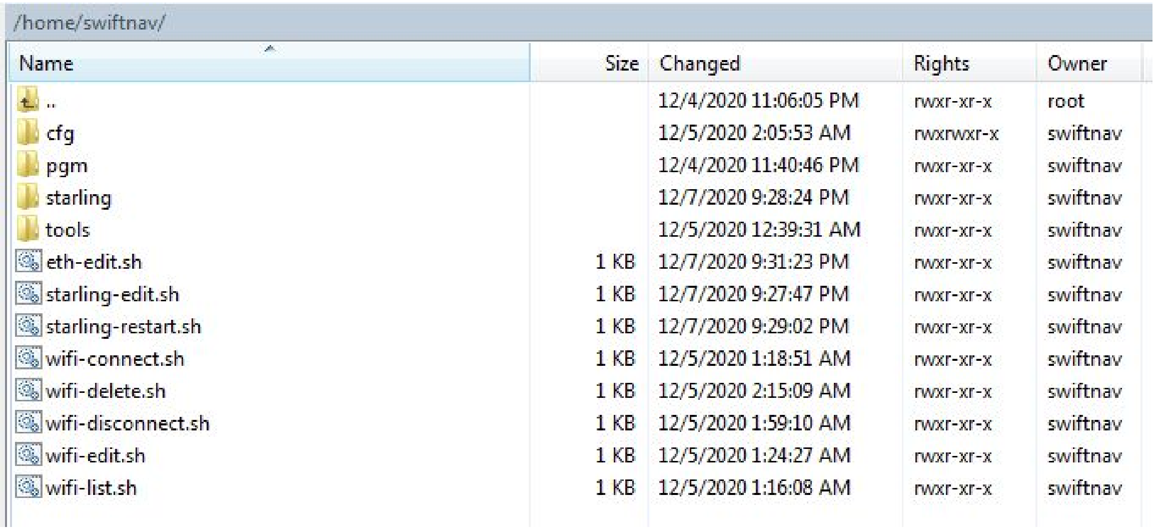
The platform runs Ubuntu 20.04 Linux server. Use SSH to control the platform. Default credentials:

Username: `swiftnav`
Password: `swiftnav`

Starling and networking use text configuration files. Connect to the PEP Linux shell to modify the configuration files.

## Home Directory Files

Configuration files are saved in the home directory as shown below:



/home/swiftnav/

| Name | Size | Changed | Rights | Owner |
|---|---|---|---|---|
| .. | | 12/4/2020 11:06:05 PM | rwxr-xr-x | root |
| cfg | | 12/5/2020 2:05:53 AM | rwxrwxr-x | swiftnav |
| pgm | | 12/4/2020 11:40:46 PM | rwxr-xr-x | swiftnav |
| starling | | 12/7/2020 9:28:24 PM | rwxr-xr-x | swiftnav |
| tools | | 12/5/2020 12:39:31 AM | rwxr-xr-x | swiftnav |
| eth-edit.sh | 1 KB | 12/7/2020 9:31:23 PM | rwxr-xr-x | swiftnav |
| starling-edit.sh | 1 KB | 12/7/2020 9:27:47 PM | rwxr-xr-x | swiftnav |
| starling-restart.sh | 1 KB | 12/7/2020 9:29:02 PM | rwxr-xr-x | swiftnav |
| wifi-connect.sh | 1 KB | 12/5/2020 1:18:51 AM | rwxr-xr-x | swiftnav |
| wifi-delete.sh | 1 KB | 12/5/2020 2:15:09 AM | rwxr-xr-x | swiftnav |
| wifi-disconnect.sh | 1 KB | 12/5/2020 1:59:10 AM | rwxr-xr-x | swiftnav |
| wifi-edit.sh | 1 KB | 12/5/2020 1:24:27 AM | rwxr-xr-x | swiftnav |
| wifi-list.sh | 1 KB | 12/5/2020 1:16:08 AM | rwxr-xr-x | swiftnav |

Fig 5. Files in home directory

Note: Instead of using the command line SSH interface you can also use a program like WinSCP (on Windows) to edit configuration files using GUI interface.

Files Description:

Directory: `cfg/`

| | |
|---|---|
| `swiftnav-ap.sh` | - PEP Wi-Fi access point configuration |
| `swiftnav-pgm.sh` | - PGM module serial port to TCP server routing configuration |

Directory: `pgm/`

| | |
|---|---|
| `pgm-update.py` | - PGM firmware updater Python script |
| `pgm-v1.1.5.4.zip` | - PGM firmware bundle |
| `t.sh` | - script to launch mini terminal |
| `u.sh` | - script to launch PGM firmware updater |

Directory: `starling/`

| | |
|---|---|
| `logs/` | - directory for Starling system logs |
| `config.yaml` | - Starling configuration file |
| `starling` | - Starling program (binary) |
| `start.sh` | - Starling start script (called from starling service) |

Directory: `tools/`

| | |
|---|---|
| `str2str` | - stream to stream routing program (binary) |

## Control Scripts

Following shell scripts are provided for convenience for common operations over SSH:

| | |
|---|---|
| `eth-edit.sh` | - Opens Ethernet configuration file for editing |
| `starling-edit.sh` | - Opens Starling configuration file for editing |
| `starling-restart.sh` | - Restarts Starling service (required after changing configuration) |
| `wifi-connect.sh` | - Connects to a Wi-Fi access point. Requires SSID and password Parameters. Usage: `./wifi-connect.sh <SSID> <PASSWORD>` |
| `wifi-delete.sh` | - Deletes (forgets) Wi-Fi network. Requires SSID parameter. Usage: `./wifi-delete.sh <SSID>` |
| `wifi-disconnect.sh` | - Disconnects from Wi-Fi access point |
| `wifi-edit.sh` | - Opens Wi-Fi access point configuration file |
| `wifi-list.sh` | - Scans and lists nearby Wi-Fi networks |

Refer to Appendix E for scripts content.

## Platform Setup Example with Wi-Fi Connections

This section provides step-by-step instructions on how to setup the PEP system using Wi-Fi connections to a control laptop and a cellular modem.

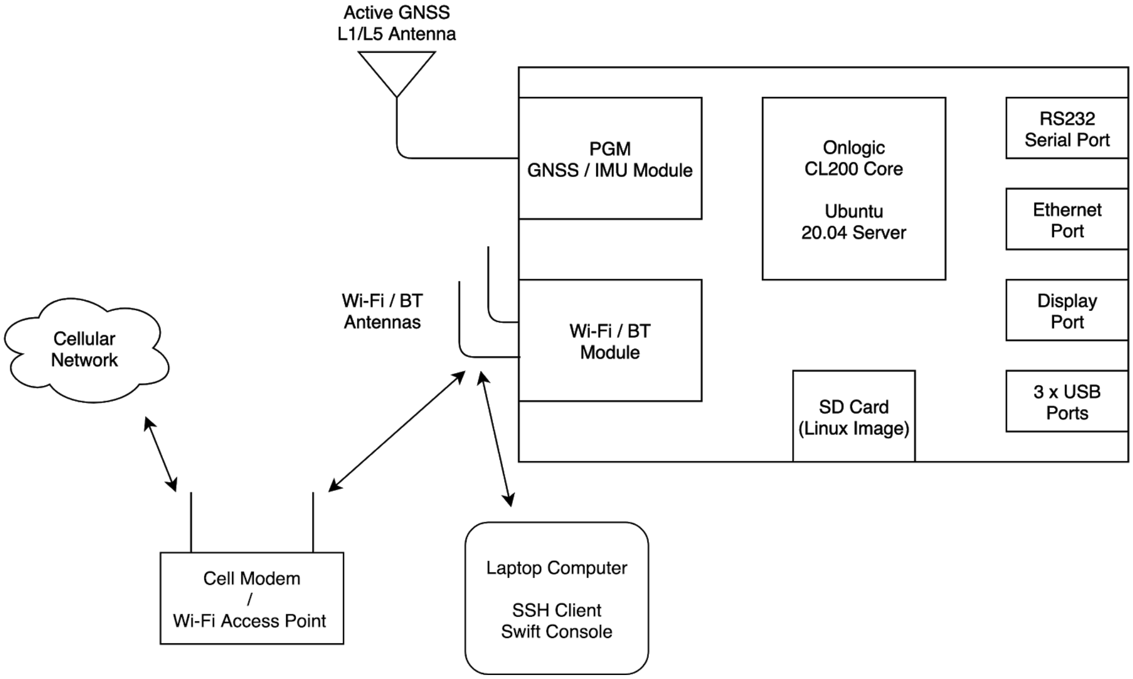


Fig 6. Platform Setup Example with Wi-Fi Connections

Required equipment:

1. PEP device with GNSS and Wi-Fi antennas.
2. Laptop computer (Windows, Linux or macOS) with Wi-Fi networking.
3. Cellular modem with Wi-Fi access point. It can be a dedicated device or a cellphone with a personal hotspot.

Example presented below uses:

- PEP with serial number T102795
- Cell modem / Wi-Fi access point with SSID “SwiftNav Mobile 4” and password “centimeter”
- Windows laptop for control

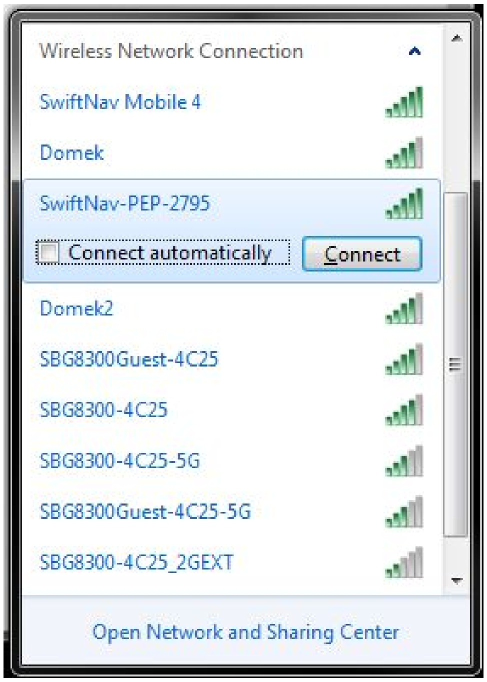
Follow the steps described below to setup and configure the system.

### Connecting and bringing up the PEP system

- Connect GNSS and Wi-Fi antennas to PEP.
- The GNSS antenna needs to be outdoors with a good sky view.
- Power up PEP and let it boot up (about 2 minutes). White LED shows power status.

### Connecting computer to PEP over Wi-Fi for the system control

- Open Wi-Fi network selection window and select **SwiftNav-PEP-xxxx** network:



Wireless Network Connection

SwiftNav Mobile 4

Domek

SwiftNav-PEP-2795

□ Connect automatically    Connect

Domek2

SBG8300Guest-4C25

SBG8300-4C25

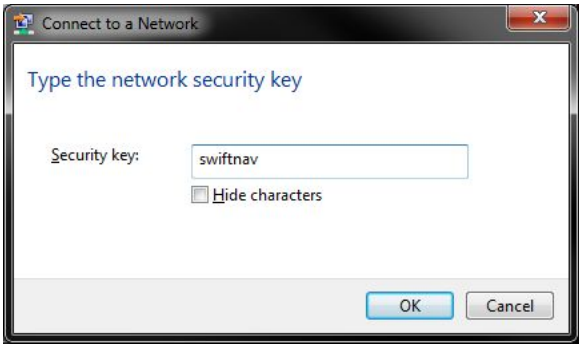
SBG8300-4C25-5G

SBG8300Guest-4C25-5G

SBG8300-4C25_2GEXT

Open Network and Sharing Center

- Enter password (`swiftnav` by default):



Connect to a Network

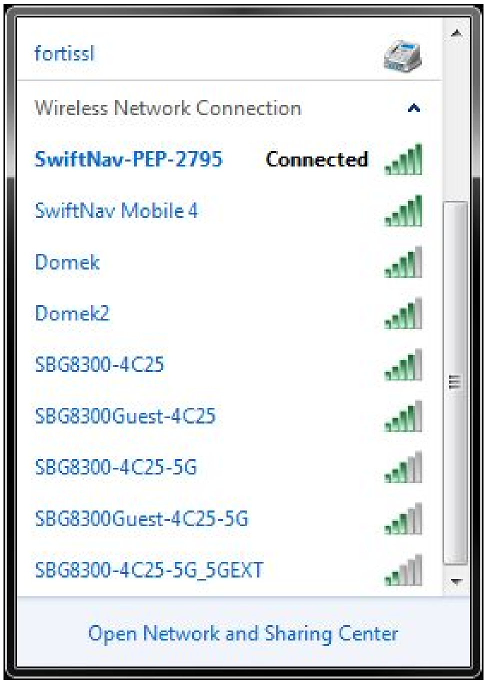
Type the network security key

Security key: swiftnav

□ Hide characters

OK    Cancel

- Network is connected:



fortissl

Wireless Network Connection

**SwiftNav-PEP-2795** **Connected**

SwiftNav Mobile 4

Domek

Domek2

SBG8300-4C25

SBG8300Guest-4C25

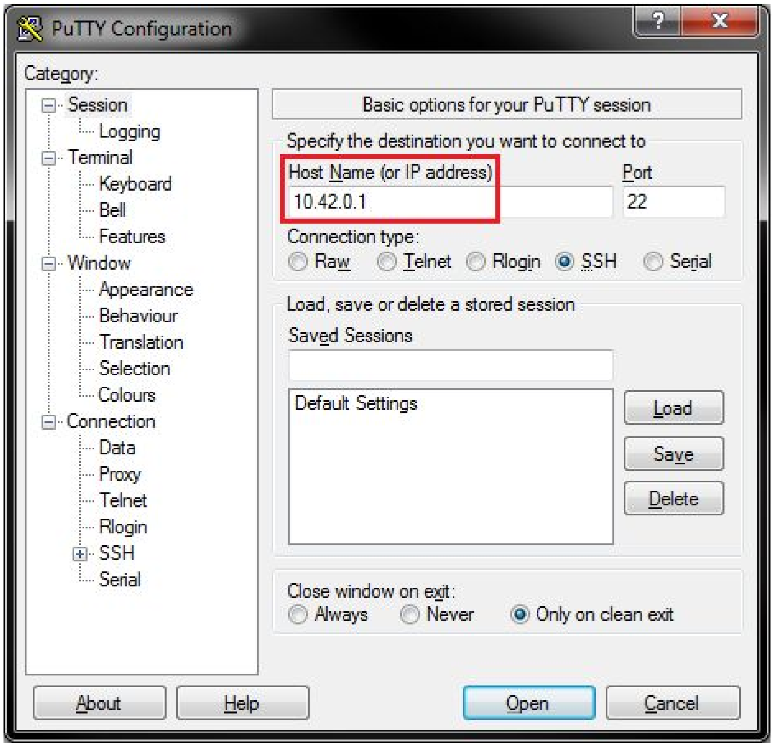
SBG8300-4C25-5G

SBG8300Guest-4C25-5G

SBG8300-4C25-5G_5GEXT

Open Network and Sharing Center

- Open SSH client (eg. PuTTY on Windows) and open connection at IP address **10.42.0.1**:



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 10.42.0.1

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

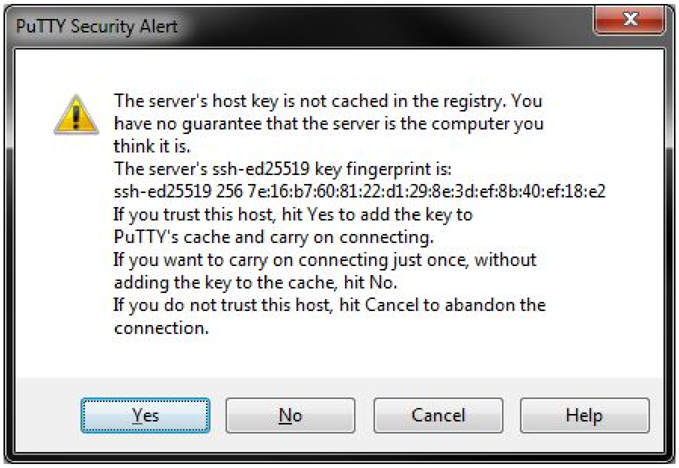
Load, save or delete a stored session

Saved Sessions

Default Settings

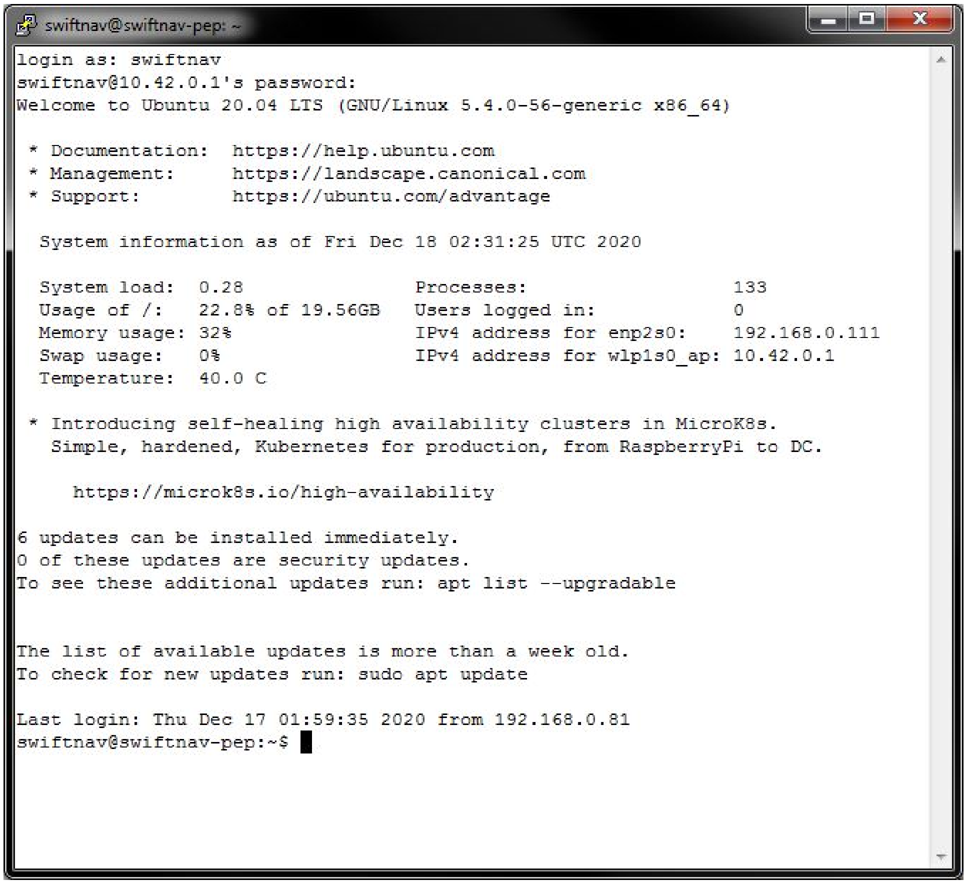
Close window on exit: ○ Always ○ Never ● Only on clean exit

- Accept server's key (this will show only once):



The server's host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.
The server's ssh-ed25519 key fingerprint is:
ssh-ed25519 256 7e:16:b7:60:81:22:d1:29:8e:3d:ef:8b:40:ef:18:e2
If you trust this host, hit Yes to add the key to PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without adding the key to the cache, hit No.
If you do not trust this host, hit Cancel to abandon the connection.

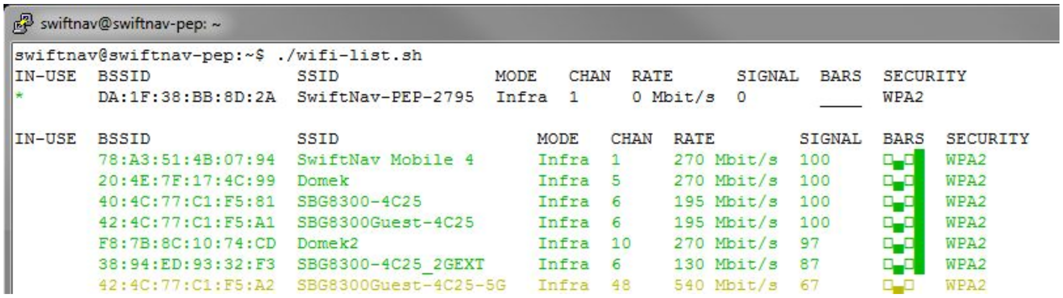
- Login as `swiftnav` and enter password (`swiftnav` by default):



```
login as: swiftnav
swiftnav@10.42.0.1's password:
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.4.0-56-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Fri Dec 18 02:31:25 UTC 2020

  System load:  0.28               Processes:                133
  Usage of /:   22.8% of 19.56GB   Users logged in:          0
  Memory usage: 32%                IPv4 address for enp2s0:  192.168.0.111
  Swap usage:   0%                 IPv4 address for wlp1s0_ap: 10.42.0.1
  Temperature:  40.0 C

 * Introducing self-healing high availability clusters in MicroK8s.
   Simple, hardened, Kubernetes for production, from RaspberryPi to DC.

     https://microk8s.io/high-availability

6 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Thu Dec 17 01:59:35 2020 from 192.168.0.81
swiftnav@swiftnav-pep:~$ █
```

## Connecting PEP to the Wi-Fi access point for Internet access

- Use `wifi-list.sh` script to list nearby networks:

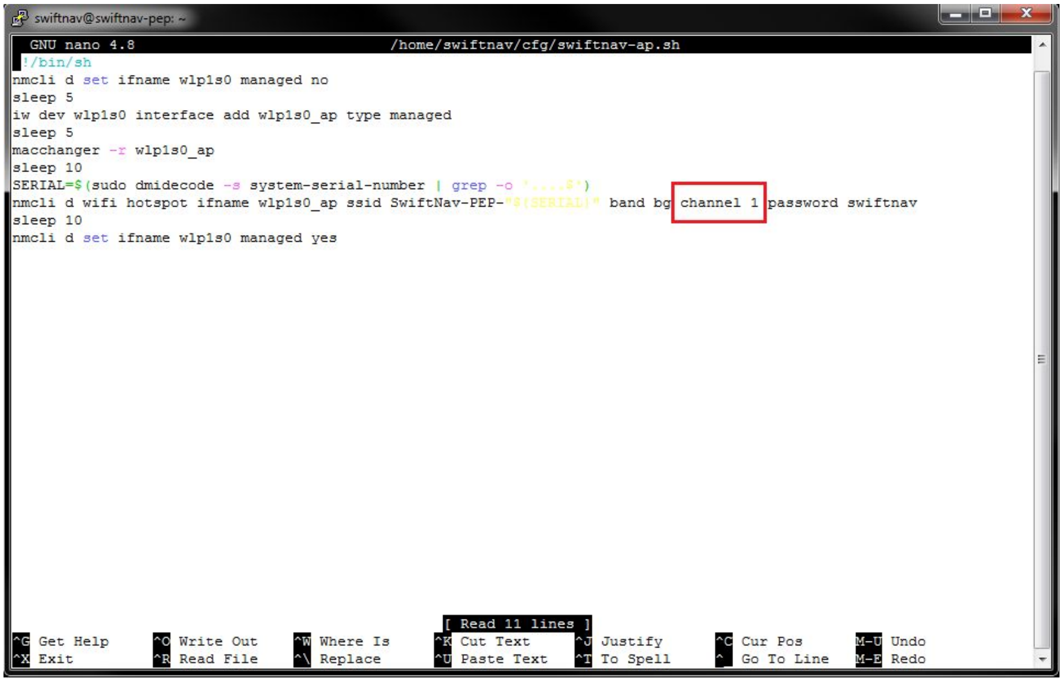


```
swiftnav@swiftnav-pep:~$ ./wifi-list.sh
IN-USE  BSSID              SSID              MODE   CHAN  RATE      SIGNAL  BARS  SECURITY
*       DA:1F:38:BB:8D:2A  SwiftNav-PEP-2795  Infra  1      0 Mbit/s  0       ____  WPA2

IN-USE  BSSID              SSID                    MODE   CHAN  RATE        SIGNAL  BARS  SECURITY
        78:A3:51:4B:07:94  SwiftNav Mobile 4       Infra  1     270 Mbit/s  100     ▂▄▆█  WPA2
        20:4E:7F:17:4C:99  Domek                   Infra  5     270 Mbit/s  100     ▂▄▆█  WPA2
        40:4C:77:C1:F5:81  SBG8300-4C25            Infra  6     195 Mbit/s  100     ▂▄▆█  WPA2
        42:4C:77:C1:F5:A1  SBG8300Guest-4C25       Infra  6     195 Mbit/s  100     ▂▄▆█  WPA2
        F8:7B:8C:10:74:CD  Domek2                  Infra  10    270 Mbit/s  97      ▂▄▆█  WPA2
        38:94:ED:93:32:F3  SBG8300-4C25_2GEXT      Infra  6     130 Mbit/s  87      ▂▄▆█  WPA2
        42:4C:77:C1:F5:A2  SBG8300Guest-4C25-5G    Infra  48    540 Mbit/s  67      ▂▄▆_  WPA2
```

- Check for the network to connect (SSID) and note the channel it uses. The Wi-Fi channel must be the same to the one PEP uses for its access point.

  If the PEP's channel is the same as the cell modem's channel, skip this step.

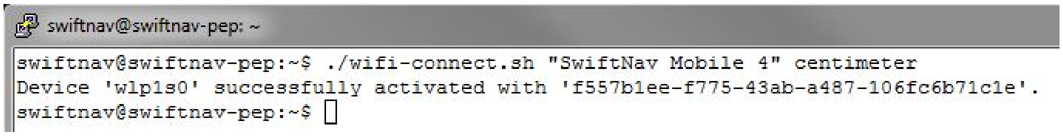
  If the cell modem channel is different from the PEP channel (by default 1) then use `wifi-edit.sh` script to open PEP Wi-Fi configuration file and change the PEP's channel number to match the cell modem:



```
  GNU nano 4.8                      /home/swiftnav/cfg/swiftnav-ap.sh
#!/bin/sh
nmcli d set ifname wlp1s0 managed no
sleep 5
iw dev wlp1s0 interface add wlp1s0_ap type managed
sleep 5
macchanger -r wlp1s0_ap
sleep 10
SERIAL=$(sudo dmidecode -s system-serial-number | grep -o '....$')
nmcli d wifi hotspot ifname wlp1s0_ap ssid SwiftNav-PEP-"${SERIAL}" band bg channel 1 password swiftnav
sleep 10
nmcli d set ifname wlp1s0 managed yes

                                        [ Read 11 lines ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     M-U Undo
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell    ^_ Go To Line  M-E Redo
```

After changing the channel, restart PEP (power cycle) for changes to take effect.

- Use `wifi-connect.sh` script to connect to the cell modem access point. The script uses two parameters: SSID and password. If any of them contain spaces use double quotes.

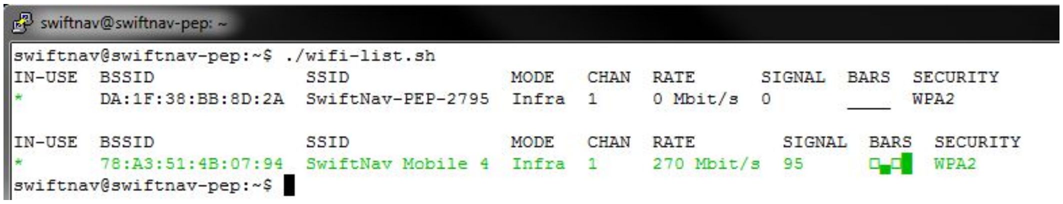


```
swiftnav@swiftnav-pep:~$ ./wifi-connect.sh "SwiftNav Mobile 4" centimeter
Device 'wlp1s0' successfully activated with 'f557b1ee-f775-43ab-a487-106fc6b71c1e'.
swiftnav@swiftnav-pep:~$ 
```

Command can also be used directly:

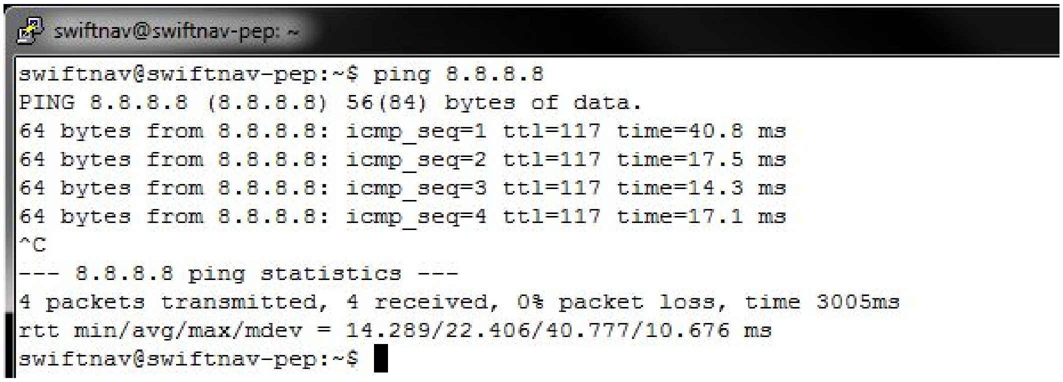
```
sudo nmcli d wifi connect <SSID> password <PASSWORD> ifname wlp1s0
```

- Run `wifi-list.sh` again to check connection:



```
swiftnav@swiftnav-pep:~$ ./wifi-list.sh
IN-USE  BSSID              SSID               MODE   CHAN  RATE       SIGNAL  BARS  SECURITY
*       DA:1F:38:BB:8D:2A  SwiftNav-PEP-2795  Infra  1     0 Mbit/s   0       ____  WPA2

IN-USE  BSSID              SSID               MODE   CHAN  RATE        SIGNAL  BARS  SECURITY
*       78:A3:51:4B:07:94  SwiftNav Mobile 4  Infra  1     270 Mbit/s  95      ▂▄▆█  WPA2
swiftnav@swiftnav-pep:~$ 
```

- A simple test to verify Internet connection is to ping server at IP address 8.8.8.8:



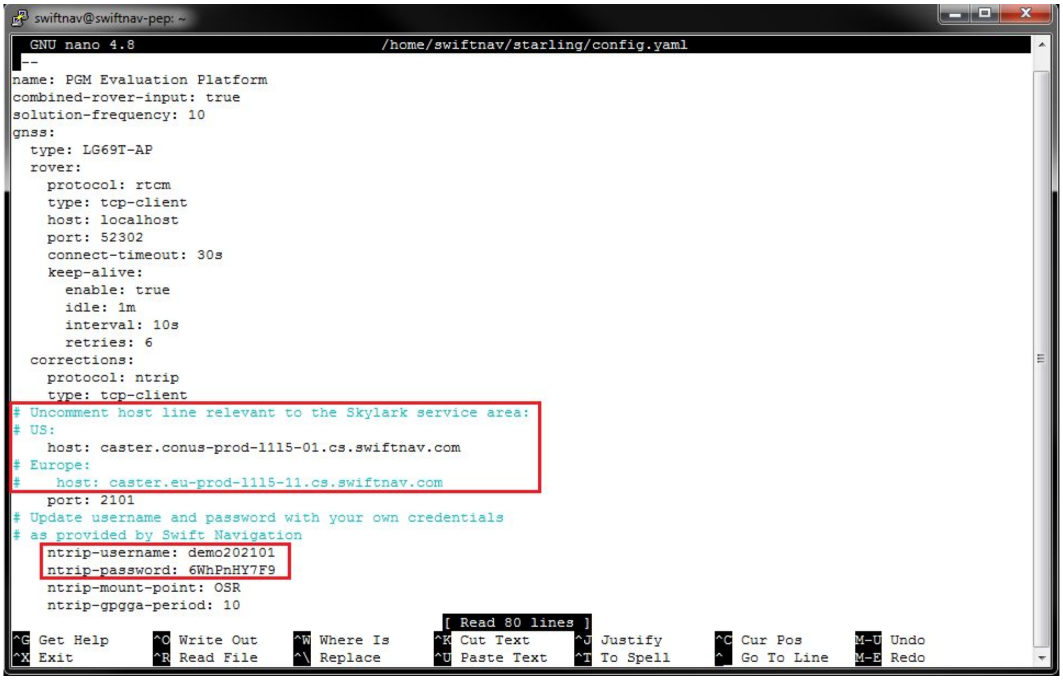
```
swiftnav@swiftnav-pep:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=117 time=40.8 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=117 time=17.5 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=117 time=14.3 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=117 time=17.1 ms
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 14.289/22.406/40.777/10.676 ms
swiftnav@swiftnav-pep:~$ 
```

## Configuring and running Starling

Starling GNSS position engine requires a simple configuration before using it. Configuration is stored in `config.yaml` file in the `starling/` directory. Following items need to be configured:

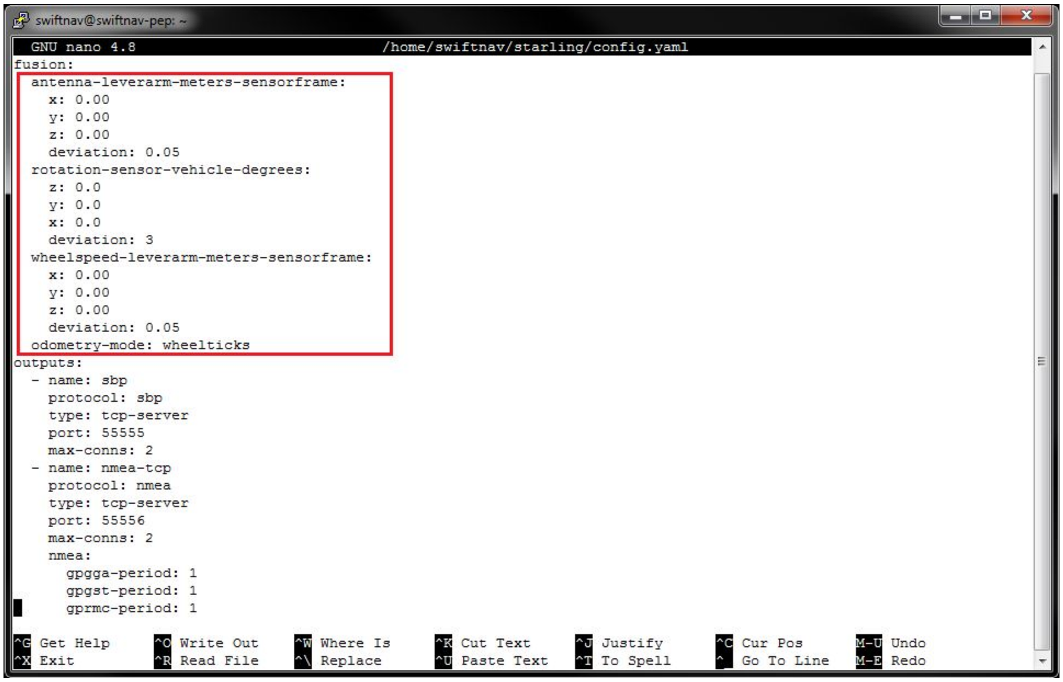
- Corrections service
- Inertial Fusion settings

- Use `skylark-edit.sh` to open the Starling configuration file for editing.
- Select Skylark host and update NTRIP credentials as provided by Swift:



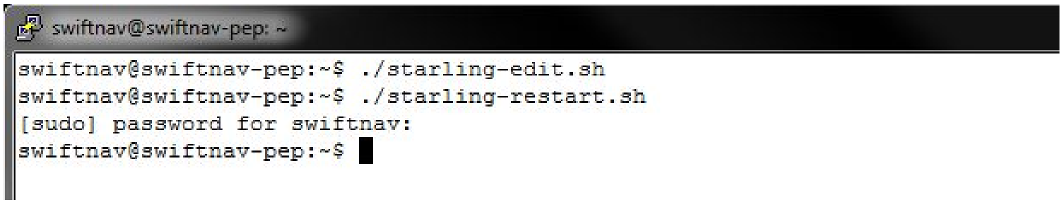
```
  GNU nano 4.8                        /home/swiftnav/starling/config.yaml
--
name: PGM Evaluation Platform
combined-rover-input: true
solution-frequency: 10
gnss:
  type: LG69T-AP
  rover:
    protocol: rtcm
    type: tcp-client
    host: localhost
    port: 52302
    connect-timeout: 30s
    keep-alive:
      enable: true
      idle: 1m
      interval: 10s
      retries: 6
  corrections:
    protocol: ntrip
    type: tcp-client
# Uncomment host line relevant to the Skylark service area:
# US:
    host: caster.conus-prod-1115-01.cs.swiftnav.com
# Europe:
#     host: caster.eu-prod-1115-11.cs.swiftnav.com
    port: 2101
# Update username and password with your own credentials
# as provided by Swift Navigation
    ntrip-username: demo202101
    ntrip-password: 6WhPnHY7F9
    ntrip-mount-point: OSR
    ntrip-gpgga-period: 10
                                                 [ Read 80 lines ]
^G Get Help     ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos      M-U Undo
^X Exit         ^R Read File    ^\ Replace      ^U Paste Text   ^T To Spell     ^_ Go To Line   M-E Redo
```

- Update device orientation and lever arm. See Appendix A for typical orientations.



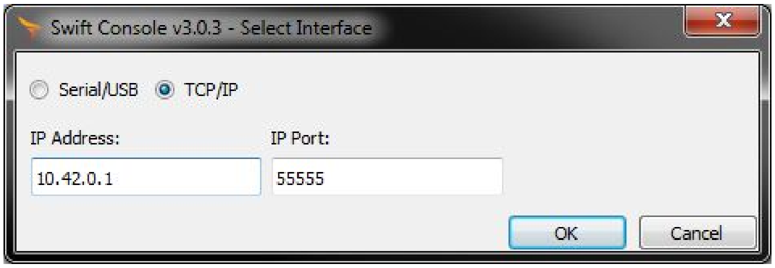
```
  GNU nano 4.8                        /home/swiftnav/starling/config.yaml
fusion:
  antenna-leverarm-meters-sensorframe:
    x: 0.00
    y: 0.00
    z: 0.00
    deviation: 0.05
  rotation-sensor-vehicle-degrees:
    z: 0.0
    y: 0.0
    x: 0.0
    deviation: 3
  wheelspeed-leverarm-meters-sensorframe:
    x: 0.00
    y: 0.00
    z: 0.00
    deviation: 0.05
  odometry-mode: wheelticks
outputs:
  - name: sbp
    protocol: sbp
    type: tcp-server
    port: 55555
    max-conns: 2
  - name: nmea-tcp
    protocol: nmea
    type: tcp-server
    port: 55556
    max-conns: 2
    nmea:
      gpgga-period: 1
      gpgst-period: 1
      gprmc-period: 1

^G Get Help     ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos      M-U Undo
^X Exit         ^R Read File    ^\ Replace      ^U Paste Text   ^T To Spell     ^_ Go To Line   M-E Redo
```

- Save file, close editor and restart Starling service for changes to take effect:



```
swiftnav@swiftnav-pep:~$ ./starling-edit.sh
swiftnav@swiftnav-pep:~$ ./starling-restart.sh
[sudo] password for swiftnav:
swiftnav@swiftnav-pep:~$
```

- Open Swift Console to check GNSS operation and to make logs:

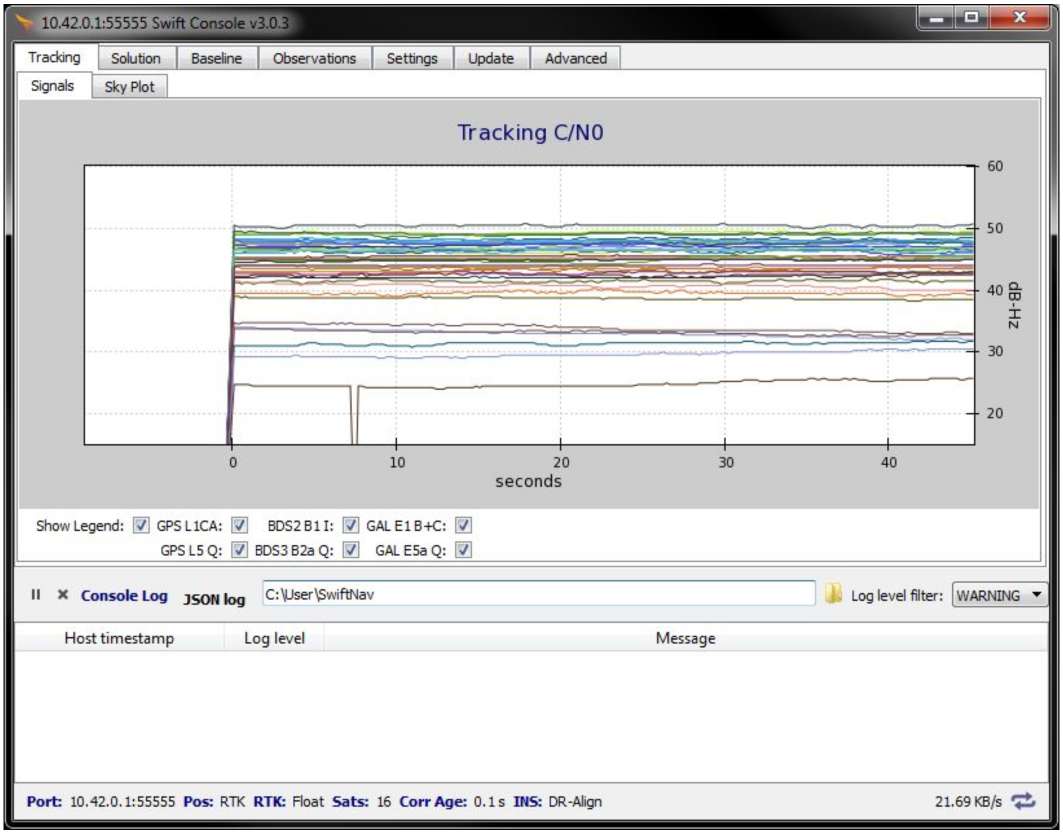


○ Serial/USB ● TCP/IP

IP Address: 10.42.0.1

IP Port: 55555

Swift Console main window:



Show Legend: ☑ GPS L1CA: ☑ BDS2 B1 I: ☑ GAL E1 B+C: ☑
GPS L5 Q: ☑ BDS3 B2a Q: ☑ GAL E5a Q: ☑

C:\User\SwiftNav Log level filter: WARNING

**Port:** 10.42.0.1:55555 **Pos:** RTK **RTK:** Float **Sats:** 16 **Corr Age:** 0.1 s **INS:** DR-Align 21.69 KB/s

- To make a log click **JSON Log** button on Swift Console. To stop logging click the same button again.

# Appendix A - Orientation Settings

Use the worksheet below to determine corresponding euler angles for the Starling orientation settings (`rotation-sensor-vehicle-degrees`). Angles are in degrees.

| | | | |
|---|---|---|---|
| **X: Forward**<br>**Y: Left**<br>Z: 0.0<br>Y: 0.0<br>X: 180.0 | **X: Right**<br>**Y: Forward**<br>Z: 90.0<br>Y: 0.0<br>X: 180.0 | **X: Backward**<br>**Y: Right**<br>Z: 180.0<br>Y: 0.0<br>X: 180.0 | **X: Left**<br>**Y: Backward**<br>Z: -90.0<br>Y: 0.0<br>X: 180.0 |
| **X: Forward**<br>**Y: Right**<br>Z: 0.0<br>Y: 0.0<br>X: 0.0 | **X: Right**<br>**Y: Backward**<br>Z: -90.0<br>Y: 0.0<br>X: 0.0 | **X: Backward**<br>**Y: Left**<br>Z: 180.0<br>Y: 0.0<br>X: 0.0 | **X: Left**<br>**Y: Forward**<br>Z: 90.0<br>Y: 0.0<br>X: 0.0 |
| **X: Forward**<br>**Y: Up**<br>Z: 0.0<br>Y: 0.0<br>X: 90.0 | **X: Right**<br>**Y: Up**<br>Z: 0.0<br>Y: 90.0<br>X: 90.0 | **X: Backward**<br>**Y: Up**<br>Z: 180.0<br>Y: 0.0<br>X: -90.0 | **X: Left**<br>**Y: Up**<br>Z: 0.0<br>Y: -90.0<br>X: 90.0 |
| **X: Forward**<br>**Y: Down**<br>Z: 0.0<br>Y: 0.0<br>X: -90.0 | **X: Right**<br>**Y: Down**<br>Z: 0.0<br>Y: -90.0<br>X: -90.0 | **X: Backward**<br>**Y: Down**<br>Z: 180.0<br>Y: 0.0<br>X: 90.0 | **X: Left**<br>**Y: Down**<br>Z: 0.0<br>Y: 90.0<br>X: -90.0 |
| **X: Up**<br>**Y: Left**<br>Z: 0.0<br>Y: 90.0<br>x:-180.0 | **X: Up**<br>**Y: Forward**<br>Z: 90.0<br>Y: 0.0<br>X: -90.0 | **X: Up**<br>**Y: Right**<br>Z: 0.0<br>Y: -90.0<br>X: 0.0 | **X: Up**<br>**Y: Backward**<br>Z: -90.0<br>Y: 0.0<br>X: 90.0 |
| **X: Down**<br>**Y: Left**<br>Z: 0.0<br>Y: -90.0<br>X: 180.0 | **X: Down**<br>**Y: Forward**<br>Z: 90.0<br>Y: 0.0<br>X: 90.0 | **X: Down**<br>**Y: Right**<br>Z: 0.0<br>Y: 90.0<br>X: 0.0 | **X: Down**<br>**Y: Backward**<br>Z: -90.0<br>Y: 0.0<br>X: -90.0 |

# Appendix B - Default Settings

**Linux:**

Server name: `swiftnav-pep`
Username: `swiftnav`
Password: `swiftnav`

**Wi-Fi Access Point:**

Band: 2.4 GHz
RF channel: 1
SSID: **SwiftNav-PEP-xxxx** where xxxx is the last 4 digits of the device serial number
Password: `swiftnav`
IP: 10.42.0.1

**Wi-Fi Client:**

Not configured.

**Bluetooth:**

Not configured.

**Ethernet Port:**

Network configuration: DHCP

**RS232 Serial Port:**

Baud rate: 115200 bps, 8N1.
Flow control: Disabled

**Starling:**

10 Hz SBP output on TCP server port 55555
10 Hz NMEA output on TCP server port 55556
10 Hz NMEA output on RS232 serial port

**Skylark:**

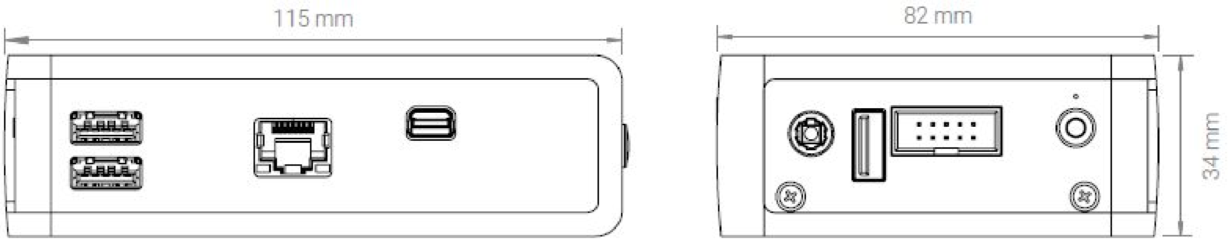
Region: North America
Credentials: Factory with short expiration

**Inertial Fusion:**

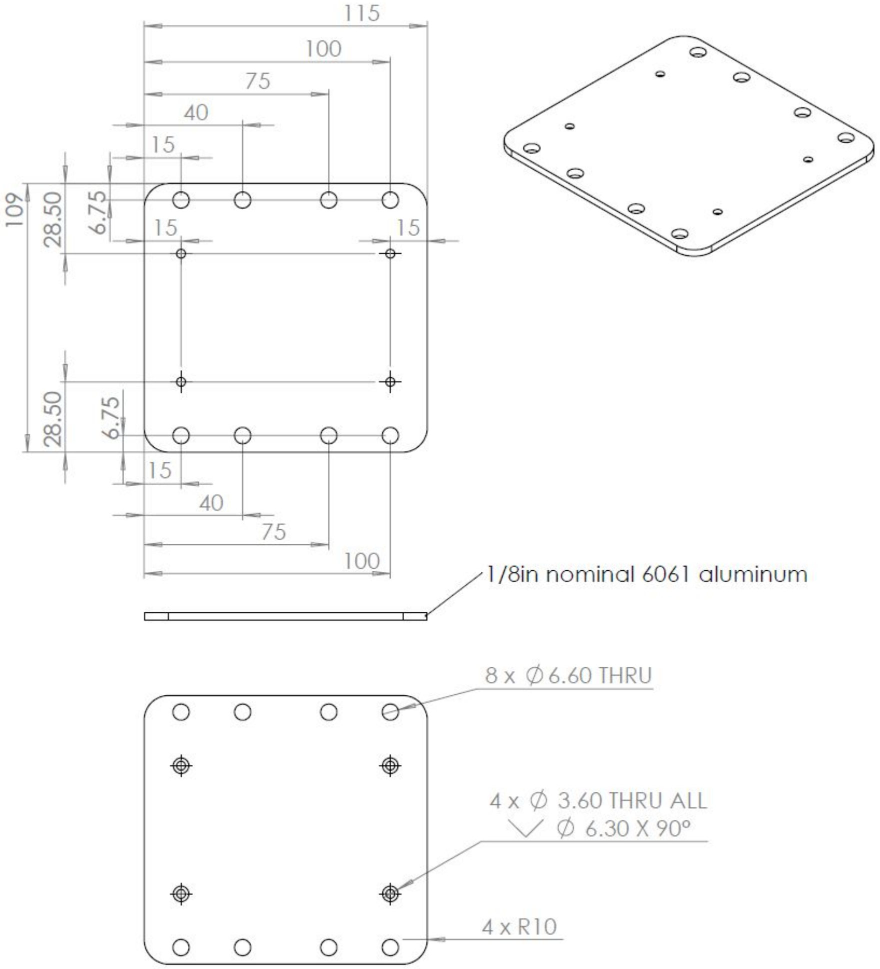
Not configured.

# Appendix C - Dimensions

**PEP Device:**



**Mounting Bracket:**



Dimensions are in mm.

# Appendix D - Vehicle Signals

Evaluation kit includes a M8 connector to pigtails cable for optional vehicle signals.

| M8 6 Pin Connector | Cable Color | Signal |
|---|---|---|
| Pin 1 | Black | Ground |
| Pin 2 | Red | Reverse |
| Pin 3 | Green | Vehicle Speed Signal (VSS) |
| Pin 4 | Yellow | CAN Lo |
| Pin 5 | Brown | CAN Hi |
| Pin 6 | Orange | Pulse Per Second (PPS) |

Refer to PGM Data Sheet for detailed signal description.

Plug connector type: Hirose HR10A-7P-6S(73).

# Appendix E - Control Scripts Content

Following shell scripts are provided for convenience for common operations over SSH:

`eth-edit.sh` - Opens Ethernet configuration file for editing

```
#!/bin/sh
sudo nano /etc/netplan/00-installer-config.yaml
```

`starling-edit.sh` - Opens Starling configuration file for editing

```
#!/bin/sh
nano /home/swiftnav/starling/config.yaml
```

`starling-restart.sh` - Restarts Starling service (required after changing configuration)

```
#!/bin/sh
sudo systemctl restart starling
```

`wifi-connect.sh` - Connects to a Wi-Fi access point. Requires SSID and password params

```
#!/bin/sh
#Usage: ./wifi-connect.sh <SSID> <PASSWORD>
sudo nmcli d wifi connect "$1" password "$2" ifname wlp1s0
```

`wifi-delete.sh` - Deletes (forgets) Wi-Fi network. Requires SSID parameter.

```
#!/bin/sh
#Usage: ./wifi-delete.sh <SSID>
sudo nmcli connection delete "$1"
```

`wifi-disconnect.sh` - Disconnects from Wi-Fi access point

```
#!/bin/sh
sudo nmcli d disconnect wlp1s0
```

`wifi-edit.sh` - Opens Wi-Fi access point configuration file

```
#!/bin/sh
nano /home/swiftnav/cfg/swiftnav-ap.sh
```

`wifi-list.sh` - Scans and lists nearby Wi-Fi networks

```
#!/bin/sh
nmcli d wifi list
```

# Appendix F - Additional Resources

**Onlogic Device**

www.onlogic.com/cl200g-11/
www.onlogic.com/computers/industrial/fanless/cl200-series/

**Swift Console**

support.swiftnav.com > General > Downloads > Swift Console

**Windows Tools**

PuTTY - SSH Client
www.putty.org

WinSCP - File Manager
winscp.net

**macOS Tools**

Transmit - File Manager
panic.com/transmit/

Cyberduck - File Manager
cyberduck.io/

**Linux Tools**

FileZilla - File Manager
filezilla-project.org/