navdata.net

Release 1.0

The navdata.net Community

Mar 17, 2018

Contents

CHAPTER 1

About navdata.net

1.1 Purpose

Develop a combination of easily accessible and cost efficient hard and software to allow community based precision geolocation. Compared to existing projects, we emphasize on creating a public networked service rather than an individual setup. We consider the project successful if we aggregate an equivalent or better service than existing NTRIP providers. While processing GNSS data, the related time information shall be made available in the form of public NTP servers.

1.2 Users

The project shall be valuable for the public at large serving diverse purposes. Improving the location precision for mobile phones, vehicles, farmers machinery, drone pilots or other location and navigation purposes.

1.3 Current goal

To create a firmware image combining:

1.3.1 Hardware

- Raspberry Pi 3B
- ublox M8T GPS / GLONASS / Galileo receiver

1.3.2 Software

- Yocto Project Poky
- RTKLIB (edition by RTKEXPLORER)
- NTP server

into a ready to run RTK base station streaming correction data to a PylonGPS caster.

1.4 Resources

1.4.1 Documentation

Accessible on Read the Docs and maintained at https://github.com/navdata-net/documentation

1.4.2 Installer

Maintained on github at https://github.com/navdata-net/installer

Used to develop the navdata.net configuration and for installations not using the navdata.net images for reference hardware. The resources developed in the installer repository are reused in the yocto layer repository.

1.4.3 Yocto workdir

Maintained on github at https://github.com/navdata-net/workdir

Used to simplify working with yocto, develop the navdata.net yocto image configuration and make it easy to compile and develop the navdata.net image.

1.4.4 Yocto layer

Maintained on github at https://github.com/navdata-net/meta-navdatanet

Used to develop the recipes for software navdata.net needs to add to yocto.

CHAPTER 2

Hardware for navdata.net reference basestation

This is the set of hardware currently used for development:

- NEO-M8T TIME & RAW receiver
- USB breakout board
- Raspberry PI 3 B All-In-Bundle
- NAVILOCK 60559
- DELOCK 88444
- Adapter SMA - TNC

2.1 Configuring and using GNSS receivers for navadata.net

2.1.1 u-blox

By using rtklibexplorers edition of RTKLIB we get enhanced support for u-blox receivers. On rtklibexplorers website we can find an article about Configuring the GPS receiver for use with RTKLIB.

In summary:

```
set UBX:CFG:PRT:Baudrate to 115200
disable all NMEA messages
save configuration in UBX:CFG:CFG:Save configuration
```

CHAPTER 3

Configuration and structure of navdata.net

3.1 navdata.net system

3.1.1 Overview

navdata.net combines several open source software components into an easy to use satellite navigation network that offers improved location precision to the general public as a community based service.

navdata.net provides ready to run system images for specific low cost hardware and general configuration information to allow supporters to contribute data by operating community basestations at low cost.

3.1.2 High level components

User

A user of navdata.net uses a local device, equipped with or connected to a satellite navigation receiver (eg. GPS device). The users intention is to use data from navdata.net to improve their location precision and reliability.

The navdata.net data is retrieved through a navdata.net client and handed over to evaluation by RTKlib. RTKlib combines it with the local receivers data to compute improved precision location information which is provided to client software (differential GPS).

Broadcaster and catalog

The users navdata.net client connects to navdata.net broadcasters to identify and subscribe to the basestation closest to the user (best precision). The broadcaster sends subscribed navigation data to the client.

Basestation

A Basestation is directly connected to a -fixed location- satellite navigation antenna via a receiver that supports delivery of raw signal data. The basestation evaluates the data and sends it to a navdata.net Broadcaster.

3.2 Basestation

3.2.1 Overview

The reference basestation is based on a Raspberry Pi 3 and a u-blox M8T receiver. The basestation is configured as an NTP server, RTKlib rtkrcv for location and a combination of RTKlib str2str and PylonGPS transceiver for data streaming.

3.2.2 Hardware

The Raspberry Pi 3 is connected to the M8T receiver via USB for data and GPIO pin 4 for the 1 Hz PPS signal.

3.2.3 NTP server

The NTP server is configured as a client to 4 pool.ntp.org servers of the country of the basestations public IP. The M8T PPS signal is provided via the kernel based PPS driver and used by NTPD for precision timekeeping. NTPD is configured as an NTP server for public reuse.

3.2.4 rtkrcv

RTKlibs rtkrcv is the direct user of the M8T receiver via its USB based serial tty. rtkrcv continuously computes the basestations location from the satellite data. It is configured to provide several network ports on the local IP:

```
:3130 - telnet console (-p option)
:3134 - monitoring port (-m option)
```

3.2.5 str2str

str2str is used to shield rtkrcv from user acces and convert the u-blox receiver data format into standardised RTCMv3 format.

```
:3131 - passthrough receiver data / raw (u-blox format)
:3132 - passthrough receiver data / RTCM3
:3133 - TCM3 antenna location message 1006
:3141 - integrated RTCM 3 location and raw data
```

CHAPTER 4

Developing with navdata.net

4.1 Poky

We base our image on the Poky distribution developed by the Yocto project. To avoid issues between the developers working environment and the requirements of cross compiling, we make use of Yoctos docker based build tool crops/poky.

4.2 source code location

This docker image uses a directory of its host computer as the storage for its build process. This folder is accessed at '/workdir' inside the build container. For ease of use and validity of the same paths from host and container, we store our code at that same location at '/workdir'.

CHAPTER 5

Preparing your development environment

5.1 OS setup for navdata.net development

As the underlying OS any recent linux distribution with support for docker should do. Expect up to 50GB storage space to be utilized in folder /workdir.

5.1.1 Ubuntu Desktop

Below commands are execute by user root. Based on default Desktop installation, add required packages:

```
apt install docker
```

We will run all development, compilation and image creation as user yocto:

```
adduser yocto
```

Development files will be on directory /workdir, this makes paths aligned with the yocto build container:

```
mkdir /workdir
chown yocto:yocto /workdir
```

5.2 Development environment setup

5.2.1 Clone / update workdir repository

Switch to yocto user as prepared in <os_setup>, clone your copy:

```
cd /
git clone https://github.com/navdata-net/workdir.git
```

Inside workdir, the navdata.net setup adds several additional git repositories. When you are working with navdata.net, prefer to execute your git commands from the /workdir or meta-navdatanet folder respectively to avoid any ambiguity reagrding which repository your action is meant to execute on.

Clone the poky and poky layer repositories using navdata.net script:

```
cd /workdir
./fetchrepos.sh
```

You can run the fetchrepos.sh script any time later to update / pull your copy of the repositories.

5.2.2 Compile navdata.net basestation image

To compile the navdata.net basestation image, you need to start the yocto build container and initiate bitbake inside it:

```
cd /workdir
./rundocker.sh
cd /workdir/dev
./bake.sh
```

Any ditbake commands should be run from within the docke build container. You can verify operating from the container by checking your username, which should be “pokyuser”.

CHAPTER 6

Working with navdata.net

6.1 Navdata.net development environment components

navdata.net workdir provides you with the following content.

6.1.1 Folder structure

folder dev & prod

For simplicity in the initial phases, we keep two copies of poky in separate folders instead of using git branches. This simplifies working with the build directory of bitbake inside those trees. Inside each of these copies we add additional meta layer repositories at the dev and prod directory level (poky root). Therefore the navdata.net tools are in the **'workdir repository'_** and the **'meta-navdatanet repository'_** is located inside the poky root folders /dev and /prod.

fetchrepos.sh

Retrieve and/or update all external source code, namely poky and additional poky layers into the /workdir folder:

```
cd /workdir
./fetchrepos.sh
```

rundocker.sh

Start Yocto provided build environment:

```
cd /workdir
./rundocker.sh
```

bake.sh

Execute a build of the navdata.net base station image from within docker console started above:

```
cd /workdir/dev
./bake.sh
```

CHAPTER 7

Indices and tables

- genindex
- modindex
- search